國 立 交 通 大 學

電機與控制工程學系

碩士論文

智慧型直接記憶體存取器設計

Design of Smart DMA Controller

研究生：　　蘇育緯

指導教授：　　林進燈 教授

中華民國九十四年七月

智慧型直接記憶體存取器設計

Design of Smart DMA Controller

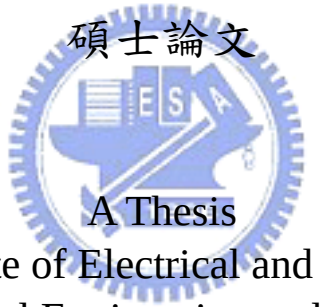
研 究 生：蘇育緯　　Student: Yu-Wei Su

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

碩士論文



A Thesis

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 2005

Hsinchu, Taiwan, Republic of China

中華民國九十四年七月

智慧型直接記憶體存取器設計

研究生：蘇育緯　　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

摘要

數位訊號處理在現代科技生活上是非常重要的研究領域，舉凡通訊、娛樂到日常生活所需，都脫離不了數位訊號處理的範疇。在消費性電子產品的高度需求下，使用數位訊號處理器除了著重運算效能外，對成本的要求更為嚴苛。高階數位訊號處理器的成本相當高，若需降低成本而採用低階處理器去獲得相同的效能，勢必需要更高的時脈與更高的功率消耗。為了使一般通用型處理器達到數位訊號處理器的效能，使得通用型處理器的效能更為強大，本論文提出一個智慧型直接記憶體存取（DMA）控制器，以輔助處理器提升效能及傳輸效率。智慧型 DMA 控制器設計以傳統 DMA 傳輸模式設計加上支援四種定址模式，能夠有效選取傳輸資料區塊，降低傳輸的頻寬及處理器的負擔。它特別的設計特色是：(1) 擁有內建乘加運算器搭配定址模式，可支援雙通道資料記憶體向量運算，協助處理器處理大量且具有規則與繁雜的數位訊號；(2) 支援周邊輸出入匯流排，使得周邊擴充更有彈性；(3) 內建乘加運算器僅僅增加 10%的硬體成本，卻能使得處理器的效能大幅躍進。本論文設計一個智慧型 DMA 控制器，並整合於已開發的通用處理器核心上，成為一顆等同數位訊號處理器（DSP-like）的晶片。此晶片採用 UMC 0.18μm 製程，以 Cell-based 方式設計，晶片面積約 2.3x2.3 mm^2，預估最大操作頻率在 100MHz。

Design of the Smart DMA Controller

Student: Yu-Wei Su Advisor: Chin-Teng Lin

Department of Electrical and Control Engineering

National Chiao-Tung University

Abstract

In recent years, digital signal processing plays an important role in our life, such as communication, entertainment and daily necessities. Under the high requirement of consumer electronics, expecting high performance, the cost requirement for a digital signal processor (DSP) is quite severe. Because the cost of a DSP is very high, if a general-purpose processor is used to reach the efficiency like a DSP, it will claim the higher clock rate and power consumption. In order to reach DSP-like performance of the processor, this thesis investigates a novel smart DMA controller to assist the processor to improve the performance and the transmission efficiency. Based on the traditional DMA, the smart DMA supports four kinds of addressing and transmission types so that it can select the region of valid data to reduce the bandwidth of transmission and the load of the processor. The features of smart DMA design are as: (1) it has a built-in multiplication-and-accumulation (MAC) which processes mass and regular data computation. Moreover, it has two channel controllers which can access two memories and perform vector operations at the same time; (2) it supports the peripheral I/O bus and it is flexible to expand I/O devices; (3) the cost of built-in MAC only increases 10%,

but it can greatly improve the performance of the processor. This thesis represents the design of smart DMA integrated into the developed processor core, and then it becomes a DSP-like processor chip. The chip has been integrated in the total area of 2.3×2.3mm^2 by using UMC 0.18μm CMOS technology and fabricated via the National Chip Implementation Center (CIC). 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，使我獲益良多。

在實驗室裡，仁峰學長給予我最直接的教導，不管大小疑難雜症，常常去請教仁峰學長。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、晴慧、立偉、盈彰、經翔、峻谷、家昇等，感謝大家在研究上的互相扶持及鼓勵。

最後要感謝交往十年的女友鈺涵的支持，讓我能專心於學術上的研究，渡過所有難關，謝謝！

目　錄

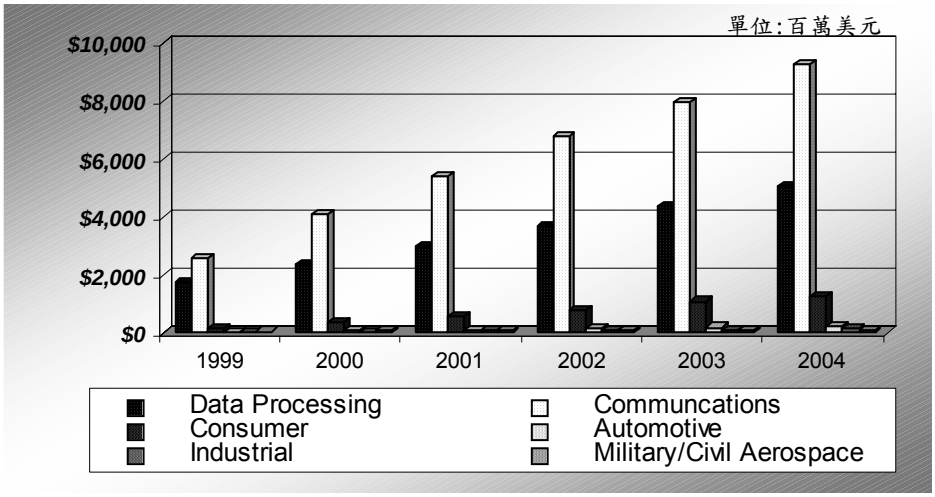
圖 目 錄

表 目 錄

第一章 緒論

1.1 簡介

數位訊號處理器(Digital Signal Processor, DSP)的應用領域很廣，但實際上沒有一個處理器能完全滿足所有的或絕大多數應用需要，設計工程師在選擇處理器時需要根據性能、成本、整合度、開發的難易程度以及功率消耗等因素進行綜合考慮。DSP 元件按設計要求可以分為兩類：第一類應用領域為廉價的、大規模嵌入式應用系統，如手機、磁碟驅動（DSP 用作伺服電機控制）以及可攜式數位音頻播放器等。在這些應用中，價格和整合度是最重要的考慮因素。對於可攜式電池供電的設備，功率消耗也是一個關鍵的因素；另一類是需要用複雜運算法對大量數據進行處理的應用，例如 3D 聲音處理和影像、語音的辨識等多媒體需求，也需要用 DSP 元件及其相關的輔助運算器。此類運算法要求苛刻、產品量大且相當複雜，所以在選擇處理器時會盡量選擇性能最佳、易於開發並支援多處理器的 DSP 元件。圖 1-1 列出近幾年 DSP 在市場上的需求趨勢，圖表上反應出通訊產品、電腦相關產品及數位消費性電子類每年估計的成長幅度相當大。



資料來源：Dataquest, 2001/01，工研院經資中心 ITIS 計畫整理 2000/03

圖 1-1：DSP 產業趨勢圖

消費性電子產品有個最大的特點在於兼顧成本及效能，造成目前定點數 16-bit DSP 的產值仍高達七成以上，低成本成為消費性電子產品很重要的一個關鍵。若為了低成本而採用低階處理器，將無法發揮其能力，主因在於數位訊號處理有相當多特別的架構及定址法[1]，如環狀定址（Circular Addressing）、位元反置定址模式（Bit-reversed Addressing）、零負擔（Zero overhead）迴圈、單一指令週期乘加運算等，若不支援其定址法，處理器要花較多的指令周期達到相同的效能，如此一來，勢必要使用更高的時脈，亦即更高的功率消耗。

在數位訊號處理的領域內，部分研究在設計增強數位訊號處理的特殊 IP，如向量內積或轉換函數的 Macrocell 等[2]。若使這些 IP 與處理器做整合，其大量連續資料傳輸及周邊裝置的管理仍須透過 DMA（Direct Memory Access）來處理。在多媒體應用中，常常會進行多重數據傳輸，所以匯流排結構必須支援 DSP 核心及 DMA 存取內、外部記憶體空間。隨著數據傳輸率和性能要求的升高，『系統性能調節』控制將變得非常重要，例如 DMA 控制器可以最佳化到每一個時脈周期傳輸一個數據。當在同一方向上有多個數據傳輸時，將能有處理大量繁雜的工作。DMA 配合處理器使用，必須能夠定址 I/O 裝置。常見有兩個方法：

1、記憶體映射（memory-mapped）I/O： 部分位址空間用來指定 I/O 裝置，對這些位址寫入或讀取，會被解譯成對 I/O 裝置下達命令。當處理器將位址和資料放到匯流排上，裝置控制器會辨別操作命令並記錄資料，當成一個命令傳到 I/O 裝置。

2、特殊的 I/O 指令： 針對個別 I/O 裝置設計相應的指令，處理器根據指令解碼，直接對 I/O 裝置下達命令。

處理器與 I/O 裝置的通訊有兩種方式，輪詢(Polling)及中斷(Interrupt)。輪詢是將 I/O 的資訊放至狀態暫存器，處理器周期性的去檢查狀態位元來決定下一個的 I/O 操作時間，並可以透過狀態暫存器獲得 I/O 的資訊，缺點是浪費大量的處理器時間。而中斷可告知處理器 I/O 裝置需要服務，當中斷發生，處理器必須確認是哪一個裝置產生的中斷，不同裝置會有不同的優先權，並根據優先權做

不同的處理。I/O 裝置和處理器的通訊只有在傳輸完成及錯誤發生才需使用中斷，使用 DMA 的傳輸，首先需設定裝置的位址、傳輸資料記憶體的來源正目的位址及傳輸的資料數，處理器由此建立 DMA 通道。接著 DMA 開始在裝置上執行命令，並取得匯流排控制權。當資料取得後，若在匯流排上傳輸超過一次才能傳完，DMA 會產生下次傳輸的記憶體位址，不需要處理器就能完成整個傳輸。許多 DMA 控制器都包含有記憶體，在傳輸延遲或等待取得匯流排控制權之前，扮演緩衝器的角色。一旦 DMA 傳輸完成，中斷發生，處理器會檢查 DMA 或記憶體，看看是否完成所有的工作。

傳統 DMA 僅有連續資料傳輸及透過 LLI（Linked List Item）不連續資料傳輸[6]，並且支援四種傳輸模式，記憶體到記憶體、記憶體到周邊、周邊到記憶體、周邊到周邊。針對目前 DMA 控制器的研究，僅著重在傳輸效率的提升和通訊應用層面[3]-[5]。為了降低數位訊號處理器開發成本和複雜度，因此，本論文提出一個智慧型 DMA 控制器（Smart DMA Controller），利用 DMA 有控管大量資料的特性及結合 DMA 資料傳輸與數位訊號處理能力的 IP，有效率地輔助一般訊號處理器做解決大量資料排序與大量乘加（MAC）運算功能，同時運用處理器和智慧型 DMA 的結合，達成具有 DSP 的工作能力。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。一般的 DMA 僅支援連續資料傳輸，因此在處理數位訊號資料時顯得沒有效率，若只需要移動資料的一半（如: downsampling 動作），仍必須將所有資料搬入記憶體中來處理，既沒有效率又佔用匯流排的頻寬。本論文提出的智慧型 DMA 改善原始 DMA 傳輸設計，並提供四種定址方式有效處理資料搬移問題：

- 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）

透過以上四種定址法的組合，達到選取有效資料，排除無效的資料，降低頻寬的使用，並提高資料傳輸的效率及降低處理器的負擔。

2. 輔助 DSP 運算的 Co-processor：

智慧型 DMA 控制器內建了一組乘加運算器（multiply-and-accumulate, MAC），搭配上述四種定址法，使得只能處理資料傳輸的 DMA 提升到運算的層次，如：DCT、FIR、DWT…等運算[7]。

- DCT：Mirror + Index-based + Increasing Addressing
- DWT：Increasing + Decreasing Addressing
- FIR：Increasing + Decreasing Addressing

3. 增加少量硬體成本達到 DSP 效能：

智慧型 DMA 支援雙通道資料記憶體快速向量運算，擁有直接存取兩個記憶體的能力，且在 I/O 匯流排上支援 APB 匯流排的標準[8]，使用者能夠在周邊匯流排上加掛相容 APB 介面的周邊裝置，擁有記憶體到記憶體、記憶體到周邊、周邊到記憶體、周邊到周邊四種傳輸模式。如表 1-1，加上 MAC 運算單元後，僅增加 10%的 Gate Count，成本相當低。

ITEM	Gate count
Smart DMAC	31.5k
DMAC	28.5k
MAC	3.0k

表 1-1：智慧型 DMA 與 DMA 面積比較

1.2 論文架構

本篇論文中，第二章介紹智慧型 DMA 控制器的設計，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 控制器和 VLIW 處理器的整合方式，及相關軟體發展及如何應用。第四章詳述實驗結果，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，在第五章做總論。

第二章 智慧型 DMA 控制器設計

本章介紹智慧型 DMA 控制器的設計，包含智慧型 DMA 的功能、架構設計及控制暫存器的說明及設定。

2.1 智慧型 DMA 控制器功能

2.1.1 傳輸模式

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。

1. 記憶體到記憶體（Memory to Memory）：

記憶體介面可同時支援兩個內部記憶體，在此模式下，可在兩記憶體間互相傳輸資料，也可在同一個記憶體內傳輸。當來源和目的設定為不同的記憶體，智慧型 DMA 同時處理讀取和寫出；當來源和目的設定為相同的記憶體，智慧型 DMA 先處理讀取動作，待緩衝器滿了或讀取的記憶體忙碌，再自動切換到寫出的動作。圖 2-1(a)是記憶體到記憶體的傳輸路徑，虛線箭頭表示所有匯流排的傳輸路徑，實線箭頭表示通道 0 的記憶體到記憶體傳輸路徑。

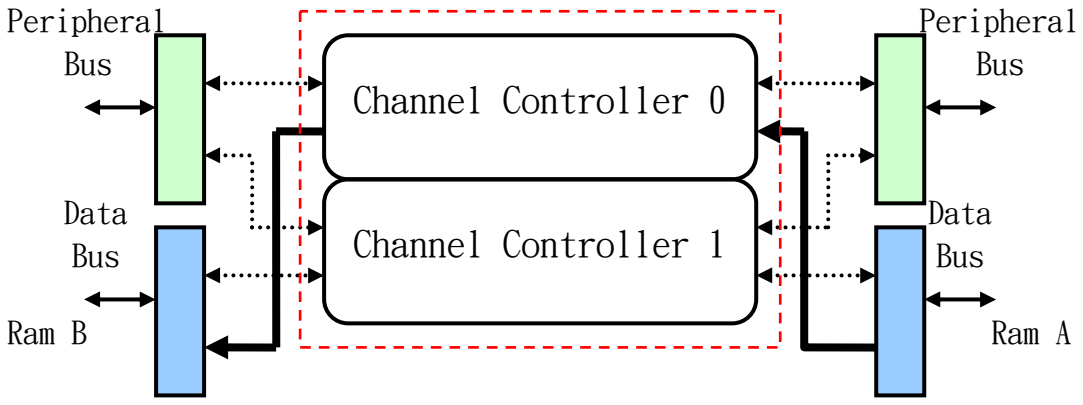


圖 2-1(a)：DMA 傳輸路徑（記憶體到記憶體）

2. 記憶體到周邊（Memory to Peripheral）：

可在記憶體中規畫一個區塊，透過智慧型 DMA 可將記憶體區塊內的資料傳輸到周邊。比如將處理過的音源資料傳輸到周邊 I²S。圖 2-1(b)為記憶體到周邊的傳輸路徑，虛線箭頭表示所有匯流排的傳輸路徑，實線箭頭表示通道 0 的記憶體到周邊傳輸路徑。

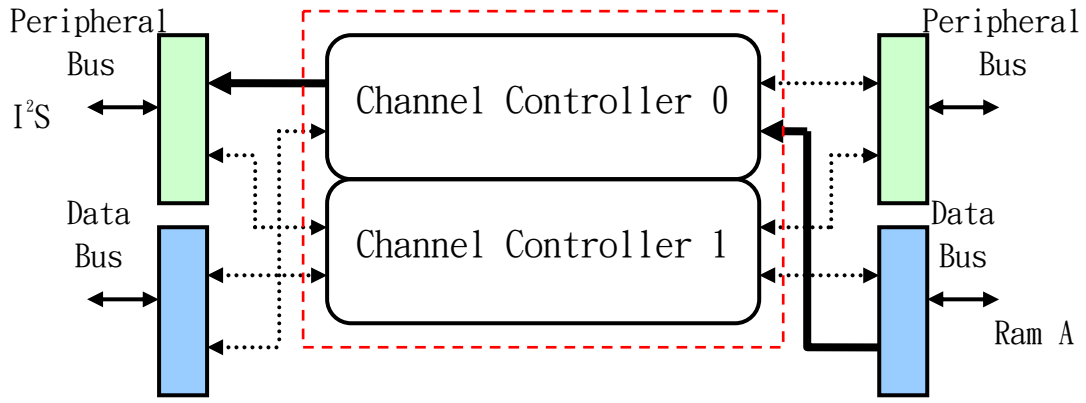


圖 2-1(b)：DMA 傳輸路徑（記憶體到周邊）

3. 周邊到記憶體（Peripheral to Memory）：

周邊裝置可透過智慧型 DMA 將資料傳回自訂的記憶體區塊內，處理器再取內部記憶體資料去處理。圖 2-1(c)為周邊到記憶體傳輸的路徑，智慧型 DMA 將音源序列資料由 I²S 存入指定內部記憶體內。虛線箭頭表示所有匯流排的傳輸路徑，實線箭頭表示通道 0 的周邊到記憶體傳輸路徑。

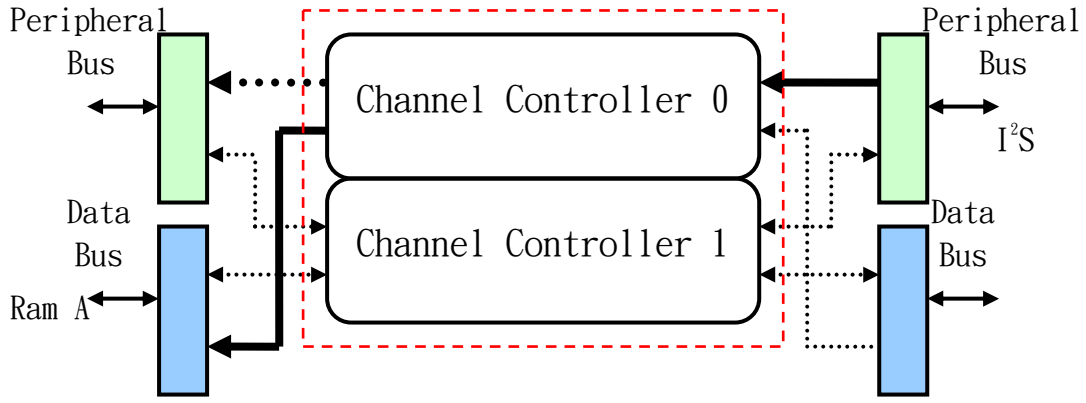


圖 2-1(c)：DMA 傳輸路徑（周邊到記憶體）

4. 周邊到周邊（Peripheral to Peripheral）：

當周邊擁有直接處理資料的能力，或是資料只要在周邊間互傳，可經由智慧型 DMA 設定周邊到周邊的傳輸，還可設定不中斷傳輸，處理器完全不需要去處理。圖 2-1(d)為周邊到周邊傳輸的路徑，虛線箭頭表示所有匯流排的傳輸路徑，實線箭頭表示通道 0 的周邊到周邊傳輸路徑。

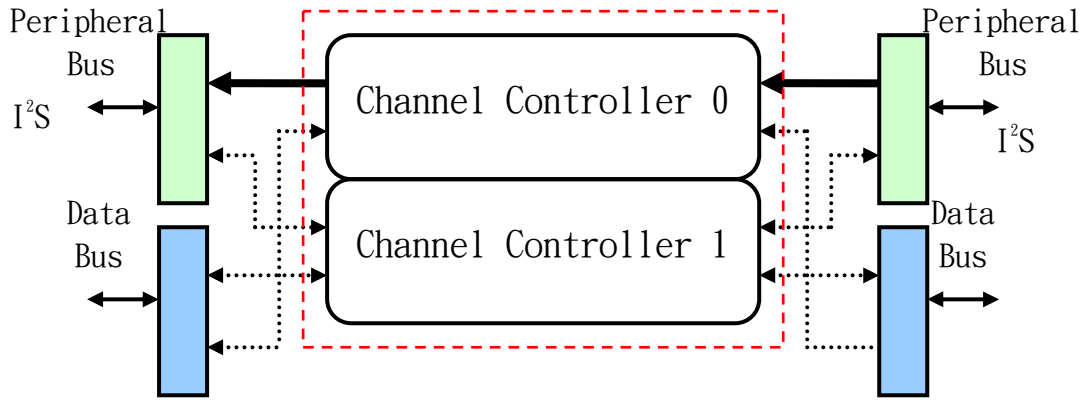


圖 2-1(d)：DMA 傳輸路徑（周邊到周邊）

2.1.2 定址模式

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，分別說明如下：

1. **遞增/遞減定址法（Increasing/Decreasing Addressing）**

最基本的定址方法，傳輸時可選擇遞增或遞減的定址方式，四種傳模式皆適用，圖 2-2 為循序遞增/遞減定址法傳輸方式，資料由記憶體 A 複製到記憶體 B，讀取時採用遞增定址，寫出時採用遞減定址。

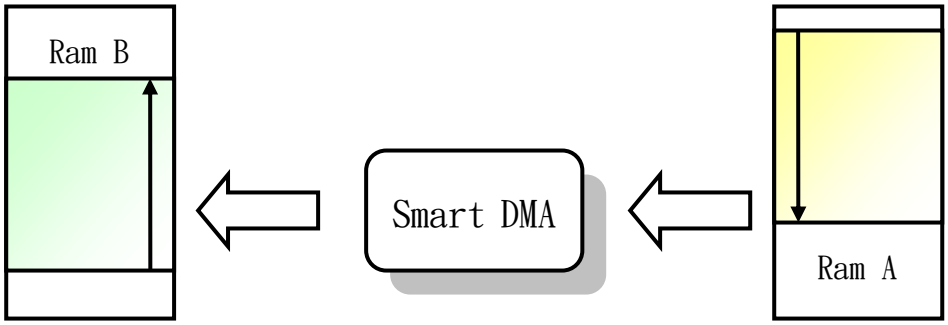


圖 2-2：遞增/遞減定址法傳輸

遞增定址法搭配乘加運算器如圖 2-3，可計算大量資料的向量內積（Inner product），資料來源可以是記憶體或周邊，也可在用一個記憶體內工作，分攤處理器的運算負擔。

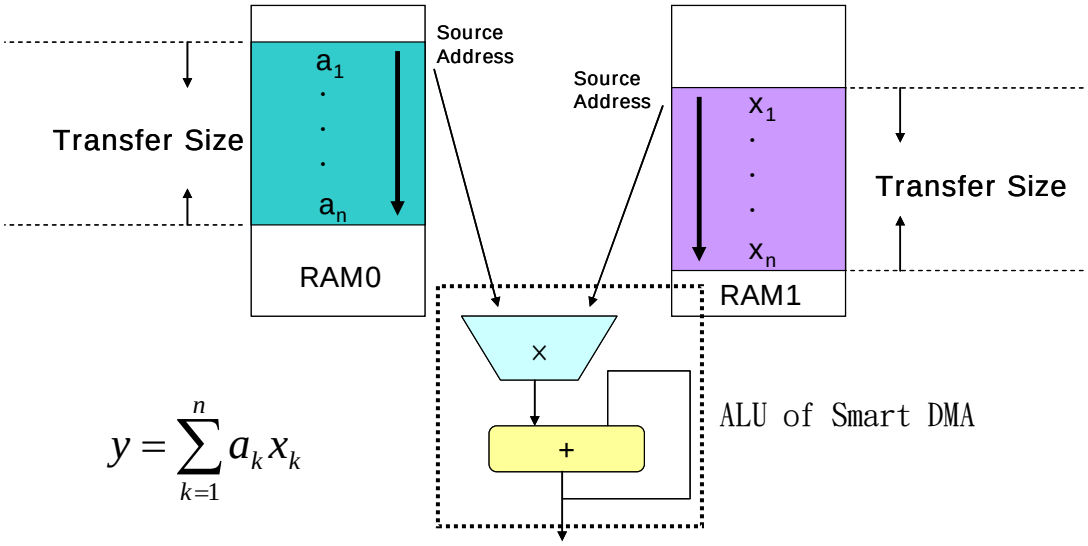


圖 2-3：智慧型 DMA 向量內積（Inner Product）運算

遞增定址法加遞減定址法，搭配乘加運算器，如圖 2-4 可計算數位訊號處理常見的摺積（Convolution）運算，資料來源可以是記憶體或周邊，也可在用一個記憶體內工作，分攤處理器的運算負擔。

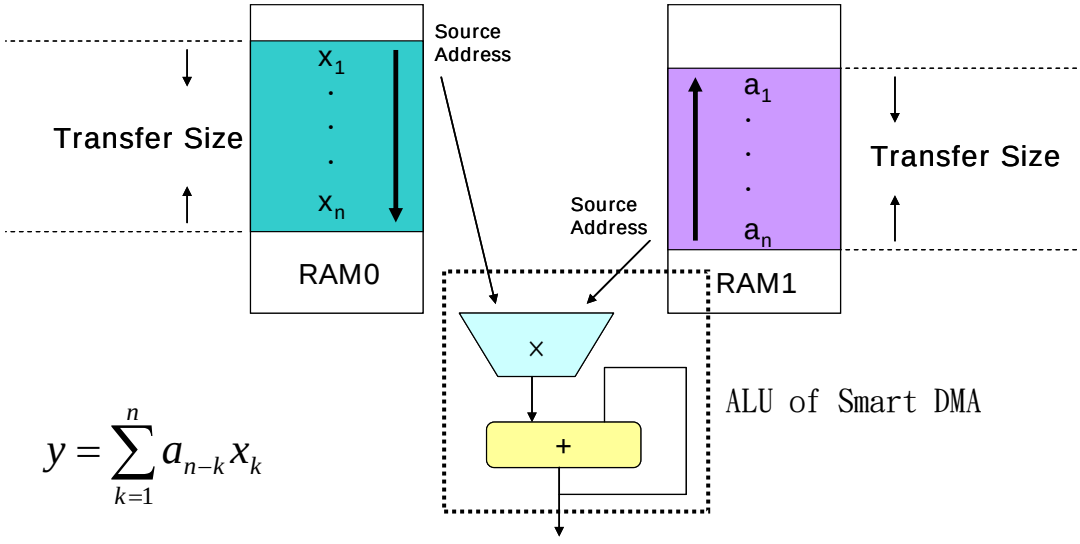


圖 2-4：智慧型 DMA 摺積（Convolution）運算

2. 環狀定址法（Circular Addressing）

環狀定址使用在資料傳輸時，使用者自訂一個資料區塊，當位址超過區塊邊界時，以環狀的方式，回到區塊起點位址，如此循環，直到資料傳輸完畢。此定址方式可選擇遞增或遞減定址，達到使用者需求。若搭配乘加運算器，可計算數位訊號處理常見的環狀褶積，如圖 2-5 所示，左邊 X_1 為一循序的資料區塊，右邊 X_2 為一環狀的資料區塊，智慧型 DMA 讀取兩區塊的資料後，做乘累加的動作，如此能達到環狀褶積的功能。

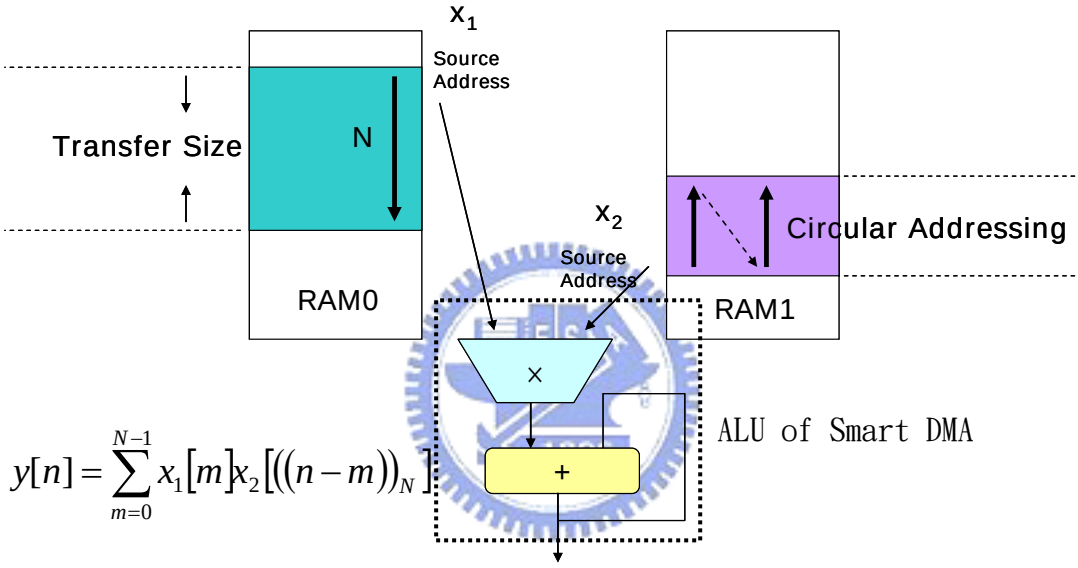


圖 2-5：智慧型 DMA 環狀摺積（Circular Convolution）運算

3. 鏡射定址法（Mirror Addressing）

鏡射定址法使用在資料傳輸時，使用者自訂一個資料區塊，當位址超過區塊邊界時，位址以鏡射的方式，往回遞增或遞減，如此循環，直到資料傳輸完畢。此鏡射定址方式可選擇遞增或遞減定址，達到使用者需求。若搭配乘加運算器，可處理數位訊號處理上餘弦轉換所遇到的鏡射定址方式，如圖 2-6 所示，左邊 X_1 為一循序的資料區塊，右邊 X_2 為一環狀的資料區塊，智慧型 DMA 讀取兩區塊的資料後，做乘累加的動作，如此能達到鏡射定址運算的功能。

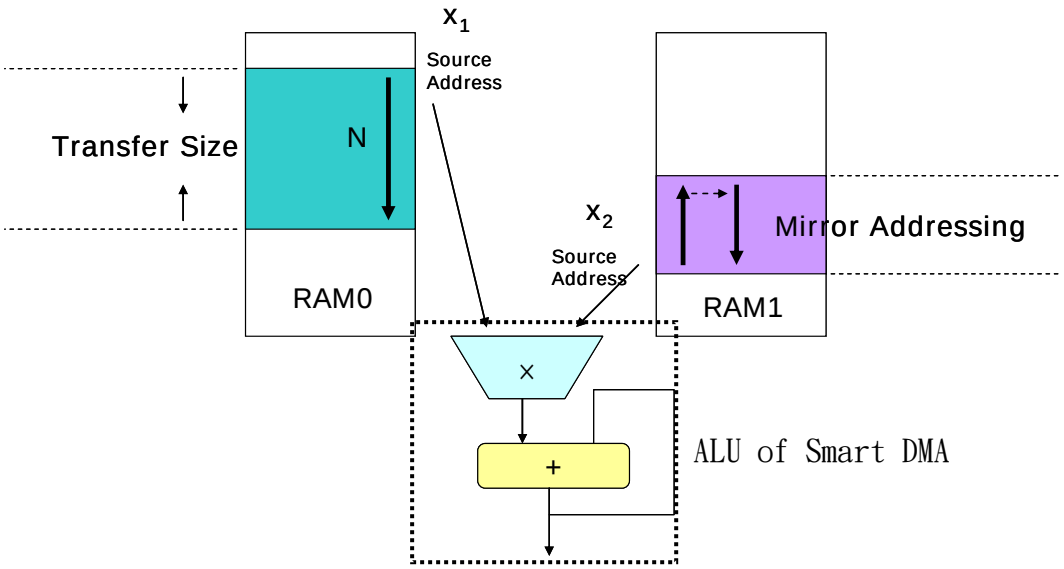
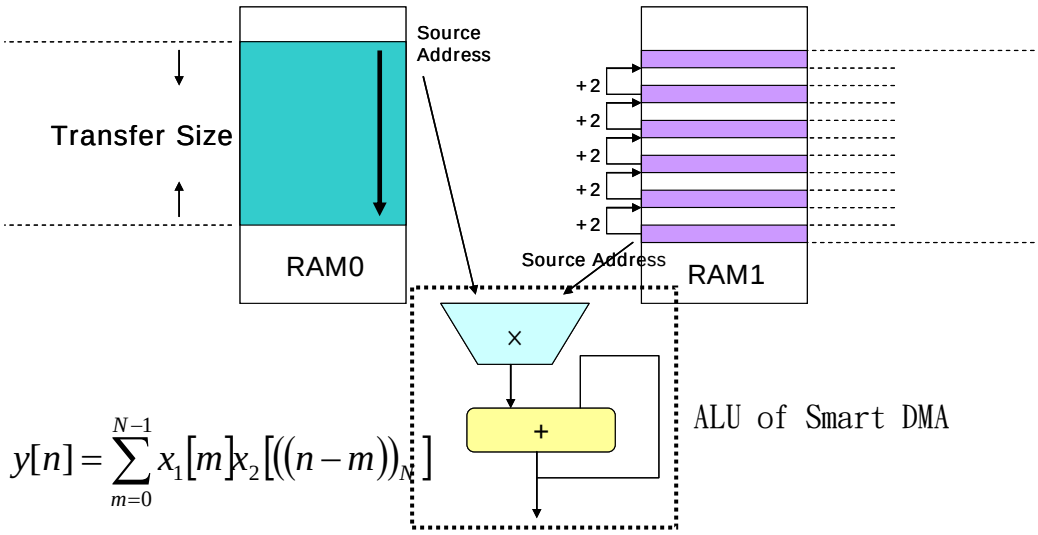


圖 2-6：智慧型 DMA 鏡射定址（Mirror Addressing）

4. 索引定址法（Index-based Addressing）

索引定址法使用在資料傳輸時，使用者自訂位址遞增或遞減的間隔，選取有效的資料，降低資料傳輸的頻寬。可設定零間隔，也就是每次抓取相同的位址。搭配乘加運算器，可處理數位訊號處理上的許多轉換（Transfer Function）、濾波器（Filter）等，如離散餘弦轉換。如圖 2-7 所示，左邊為循序資料區塊，右邊區塊以每次遞減 2 的方式讀取，智慧型 DMA 讀取兩邊資料後做乘加運算，如此達到索引定址運算的能力。



$$y[n]=\sum_{m=0}^{N-1}x_1[m]x_2[((n-m))_N]$$

圖 2-7：索引定址（Index-based Addressing）

■ 定址法組合

綜合以上四種定址法可分為三類，D-type、B-type 及 I-type，每一類選一個定址法使用，共有 2x3x1=6 種模式。索引定址法可選擇不同間隔的定址，其變化更是多樣化。

Direction (D-type)	Block (B-type)	Index (I-type)
Increasing Decreasing	Circular Mirror Normal	Index-based

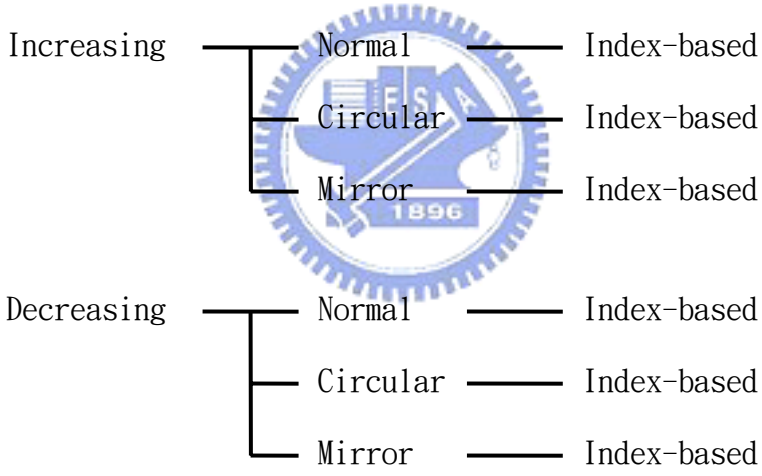


表 2-1：定址法組合表

2.2 智慧型 DMA 控制器架構

一般的 DMA 與智慧型 DMA 架構的差別在於後者在通道控制器對定址法的支援、內建 MAC 運算器、因應不同功能的暫存器組、不同匯流排的支援和直接支援雙記憶體的介面等。智慧型 DMA 整體架構設計如圖 2-8，包含通道控制器、優先權仲裁器、暫存器組、乘加運算器、中斷控制器及記憶體介面，其說明如下：

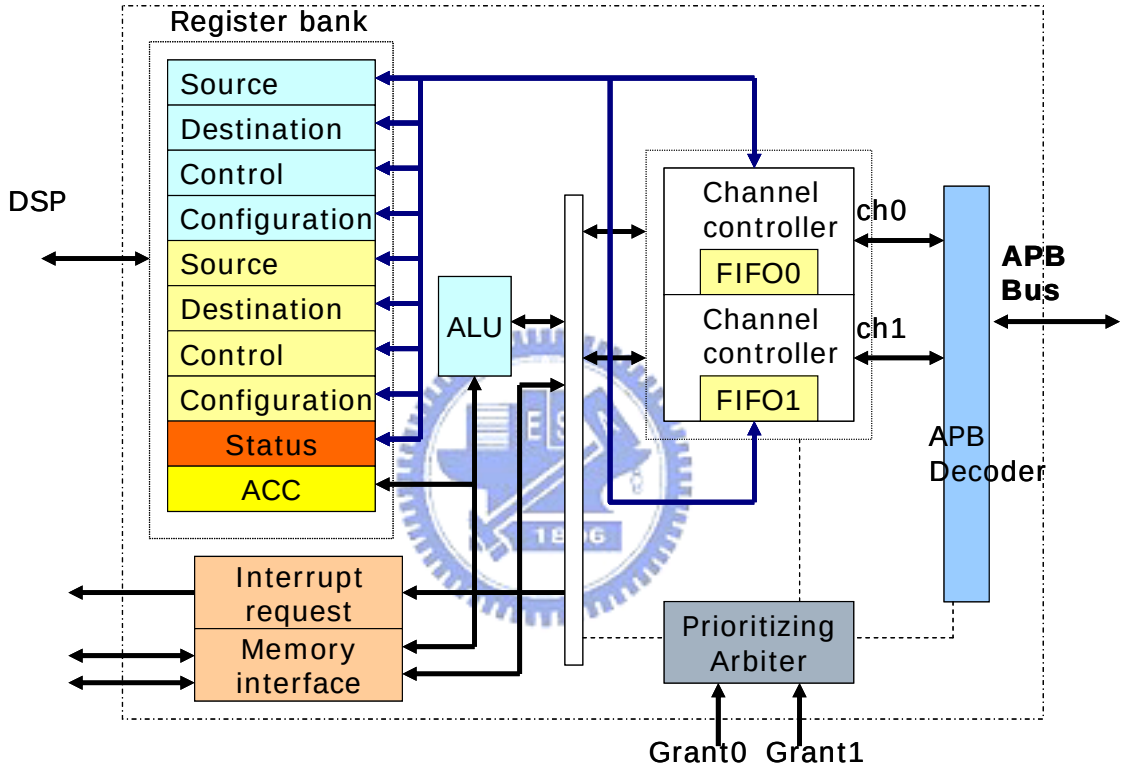


圖 2-8：智慧型 DMA 的架構

2.2.1 通道控制器（Channel Controller）

通道控制器為 DMA 裡重要的核心，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。

通道控制器內含一組讀取控制器、一組寫出控制器，兩組控制器各自獨立。

當通道控制器被開啟，讀取控制器判斷緩衝器（FIFO）內是否已滿，來決定是否將資料讀入；寫出控制器則判斷緩衝器（FIFO）內是否有資料，來決定是否將資料寫出。暫存器組直接和通道控制器溝通，暫存器組可決定智慧型 DMA 的所有功能。讀寫控制器採用有限狀態機（Finite State Machine, FSM）實現，通道控制器架構圖如圖 2-9。

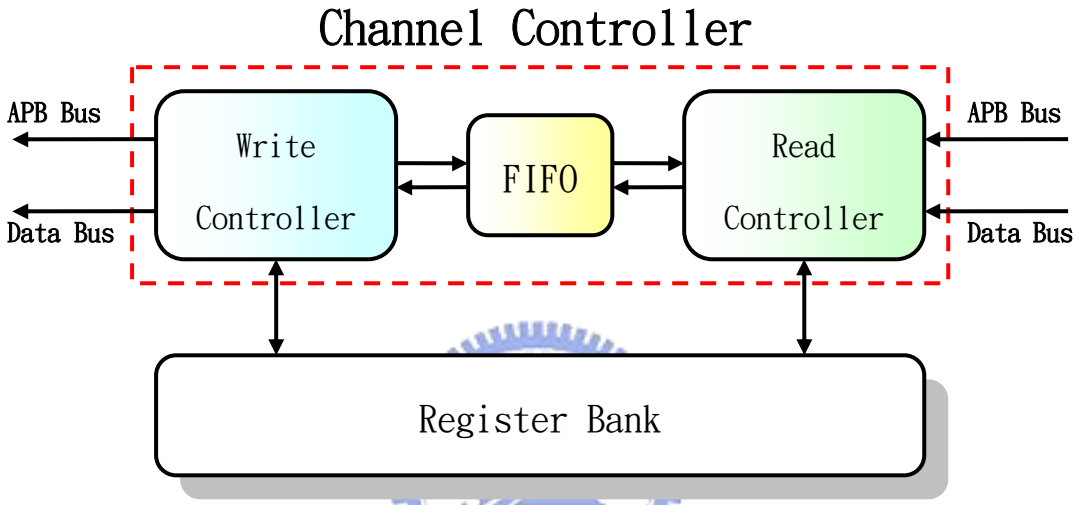


圖 2-9：通道控制器架構圖

讀/寫控制器設計狀態流程如圖 2-10 所示，讀/寫各有兩種狀態切換。使用智慧型 DMA 之前，讀/寫控制器狀態為 Disable。在啟動後，會依設定的傳輸模式進入不同的模式（Memory/Peripheral）。先進入到 Idle 模式，等待取得讀/寫控制的權利，接著進入 Setup 模式，等待來源或目的的資料有效、緩衝器同時有效時，進入 Enable 模式，進行傳輸。若資料未傳完，狀態進入 Setup 模式，等待下一筆資料的傳輸。若資料已傳完，狀態會回到原始的 Disable 模式，並發出中斷（Interrupt）的通知，完成本次的工作。

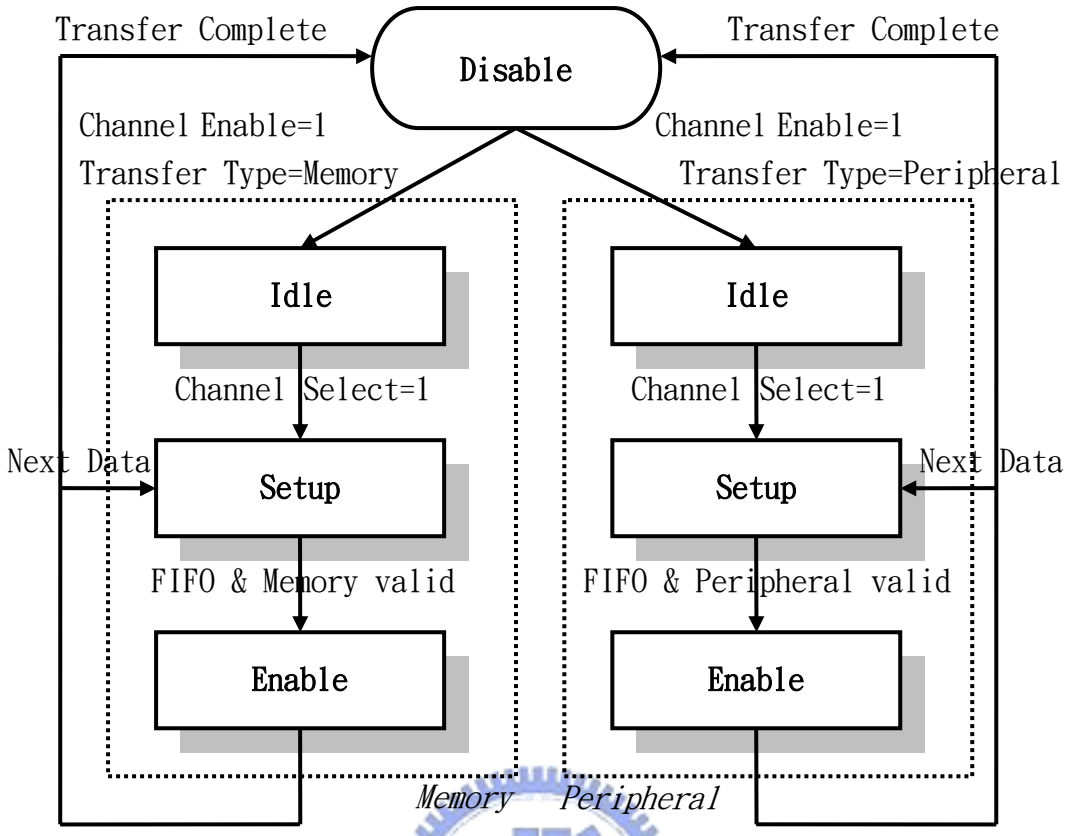


圖 2-10：讀/寫控制器設計流程圖

當利用兩通道執行 MAC 運算時，如圖 2-11，若一通道讀取資料有延遲，另一通道的寫出控制器將等待資料到齊後，再執行乘加動作。

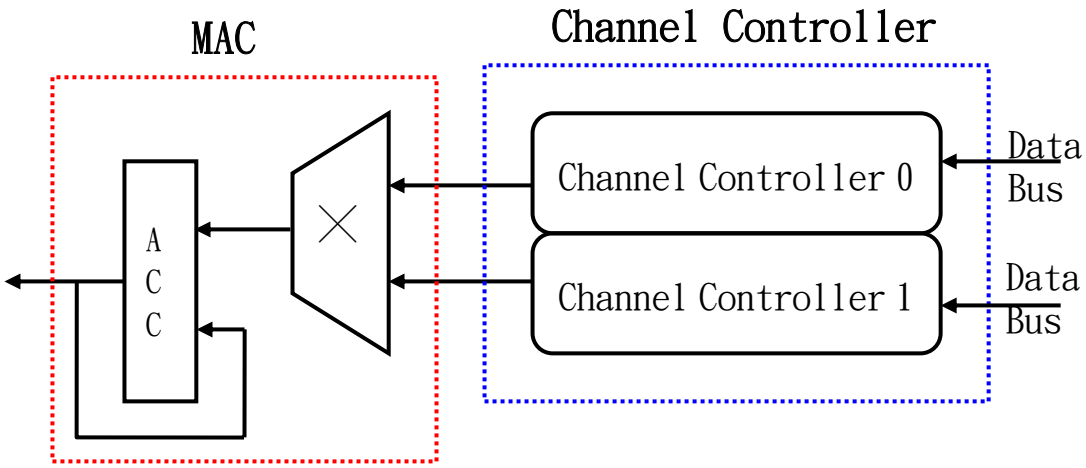


圖 2-11：使用通道控制器配合 MAC 運算

2.2.2 先進先出緩衝器（FIFO – First In First Out）

在通道控制器中需要緩衝器將資料暫存，使用同步先進先出（FIFO）。主要分為讀取控制、寫出控制及先進先出狀態，其控制訊號如表 2-2 所示，三組控制訊號分別獨立，以設計通道控制器的讀/寫控制器。

讀取控制	
data_in	欲丟入緩衝器的資料。
Push	CLK 正緣時，Push=1 時，將資料丟進緩衝器。
寫出控制	
data_out	取出的緩衝器資料。
Pop	CLK 正緣時，Pop=1 時，將資料從緩衝器取出。
先進先出狀態	
Full	緩衝器資料已滿。
Empty	緩衝器內無資料。
Half	緩衝器內資料已達半數以上。

表 2-2：先進先出控制訊號

如圖 2-12 所示，緩衝器內有兩個指標 Pt_PUSH 及 Pt_POP，分別指向緩衝器的 Push 與 Pop 的位址指標。當 Push 動作發生時，資料會被放入緩衝器，指標 Pt_PUSH 往下移，當指標超過緩衝器結尾，會回到緩衝器的起點；當 Pop 動作發生時，資料從緩衝器內丟出，指標 Pt_Pop 往下移，當指標超過緩衝器結尾，會回到緩衝器的起點。

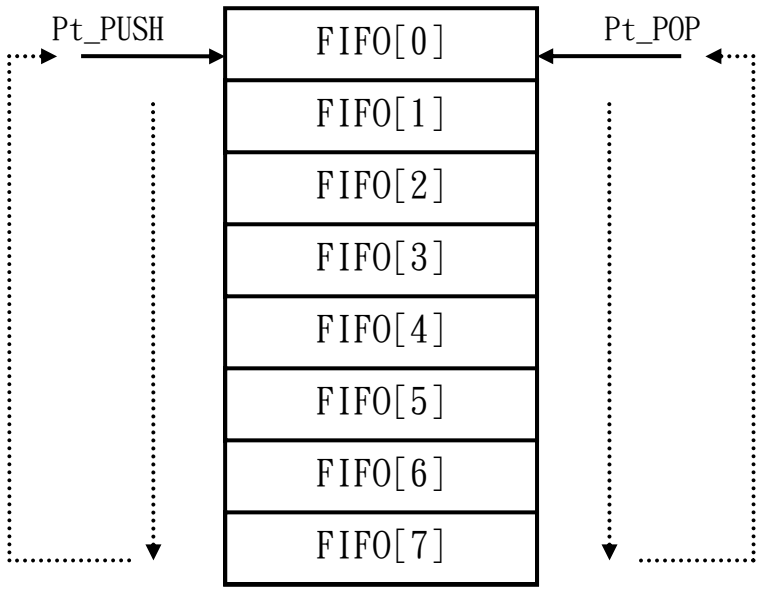


圖 2-12：緩衝器（FIFO）內部指標

緩衝器內建計數器（counter），用來計數目前已存資料數，並可由計數器得知目前緩衝器的狀況，如圖 2-13 所示：

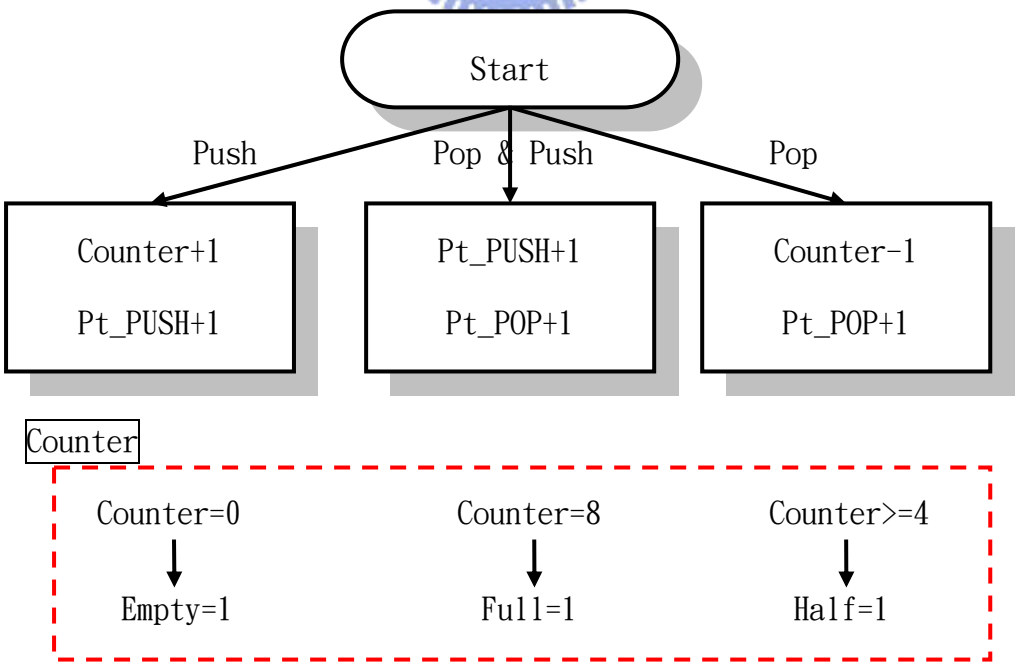


圖 2-13：緩衝器內指標、計數器示意圖

2.2.3 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，必須設計一個優先權仲裁器如圖 2-14，協調衝突的情況，決定優先的順序。

智慧型 DMA 內建兩組通道控制器 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。另有一特殊情況須注意：低優先權的通道正在傳輸時，須等待它傳完當筆資料，才將匯流排控制權交回，以防止資料的遺失。通常記憶體的傳輸資料量大且不間斷，建議在優先權低的通道上設定記憶體的傳輸，以免造成周邊傳輸慢速資料時，無法取得匯流排的控制權，影響周邊傳輸資料的正確性。

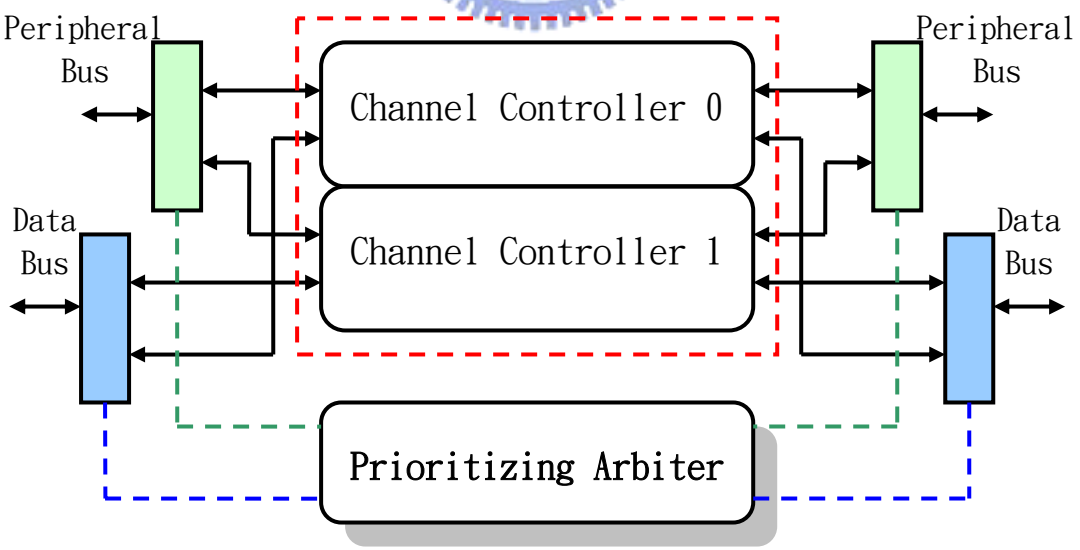


圖 2-14：優先權仲裁器（Prioritizing Arbiter）控制兩組匯流排

2.2.4 暫存器組（Register Bank）

智慧型 DMA 內部擁有十組 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-maping）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 2-15(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。

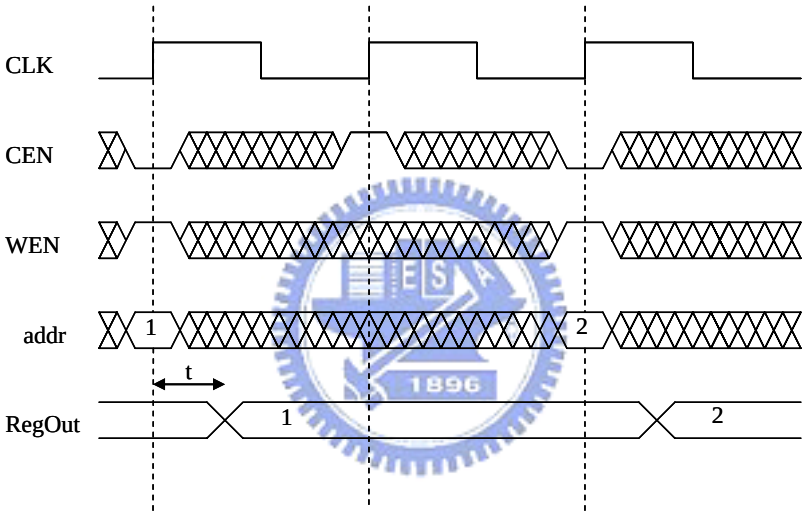


圖 2-15(a)：智慧型 DMA 暫存器組的讀取時序圖

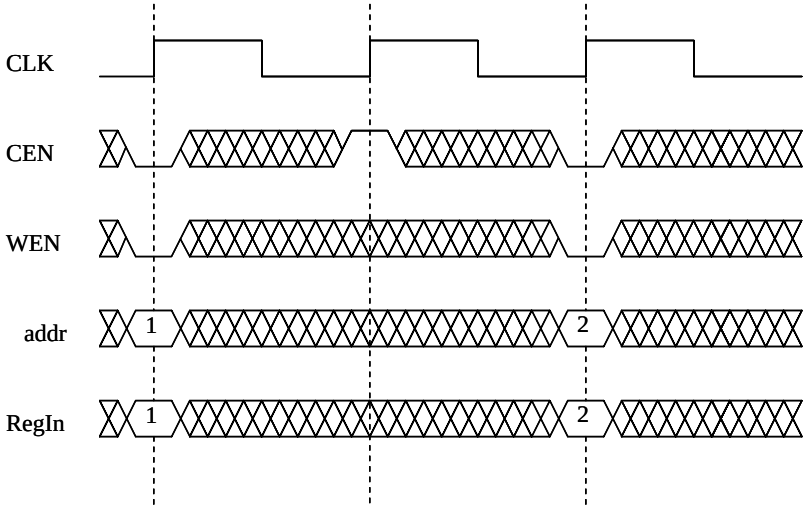


圖 2-15(b)：智慧型 DMA 暫存器組的寫入時序圖

暫存器組的周邊連接如圖 2-16 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和 MAC 運算器。

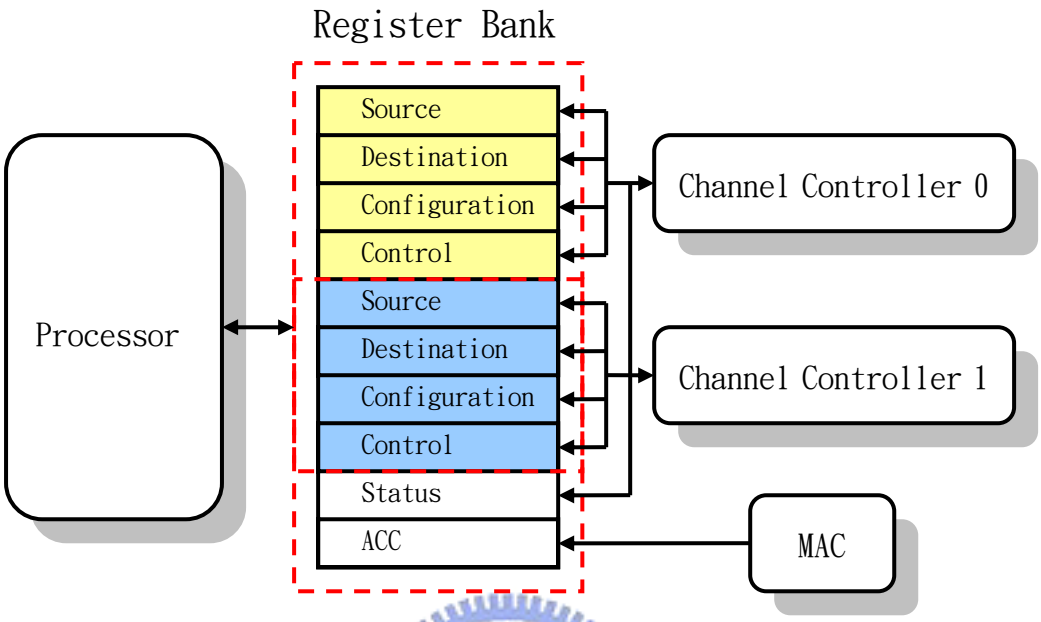


圖 2-16：暫存器組示意圖

2.2.5 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 2-17 所示。

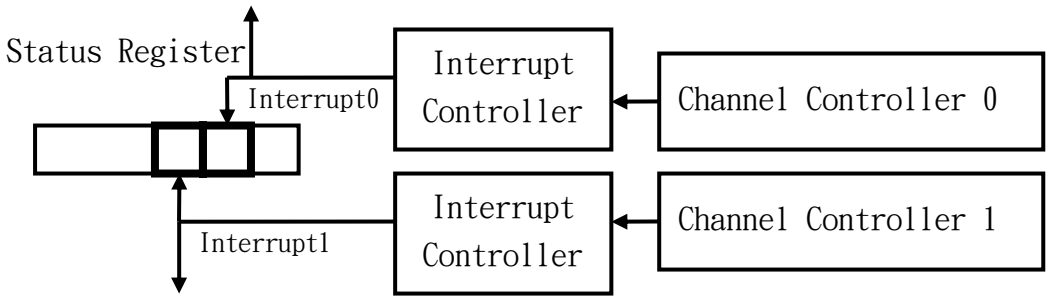


圖 2-17：中斷控制器連接狀態暫存器

2.2.6 乘加運算器（MAC）

在智慧型 DMA 裡內建乘加器（MAC），使得智慧型 DMA 處理大量資料時，也可處理大量繁複的數位訊號處理。由於乘法器在實現時，會造成整體速度上的拖累，因此，採用 2 級快速乘法器（2-stage Multiplier）做為乘加運算器的核心。乘法器為 16x16 bits 的有號數（Signed）快速乘法器，輸出為 32-bit 的有號數。累加器寬度增加到 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。運算的結果存在累加器裡，同樣是 40-bit，但取出後的結果為 32-bit。由於在計算時，用到兩通道控制器，資料有可能不會同時獲得，因此設計了一個 Enable 的訊號，用來同步乘法的結果。

2.2.7 記憶體介面（Memory Interface）

智慧型 DMA 直接支援兩個 2KB（512x32-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 2-18，記憶體讀取/寫入使用相同的記憶體介面。

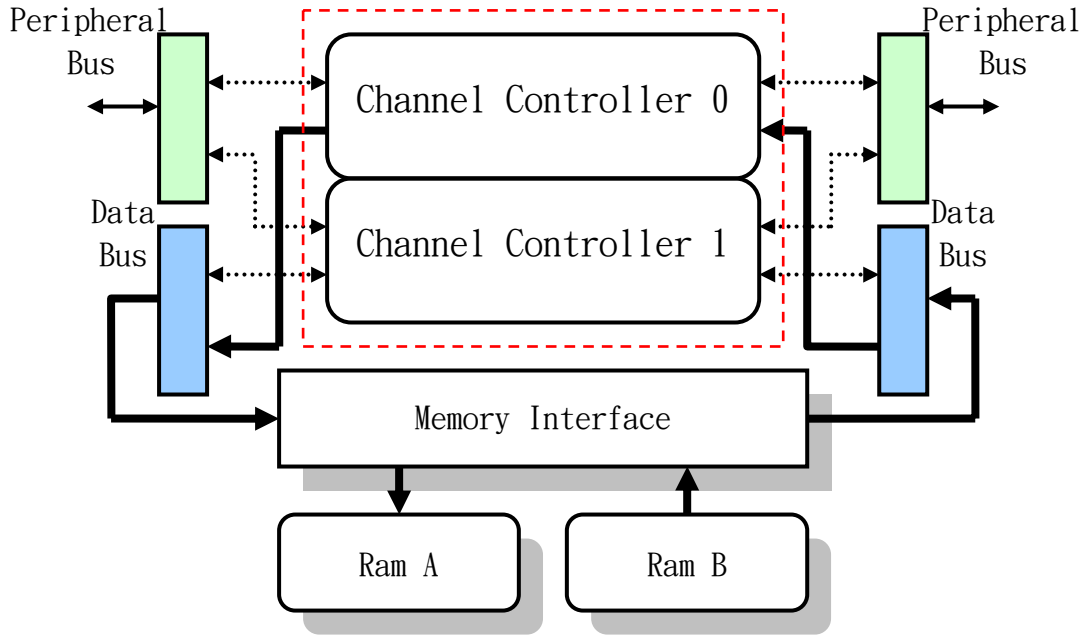


圖 2-18：透過記憶體介面的資料傳輸

通道控制器可透過暫存器的設定（見 2.3 節）選擇要存取的記憶體。因此，記憶體介面必須符合圖 2-19(a)(b)的存取時序。

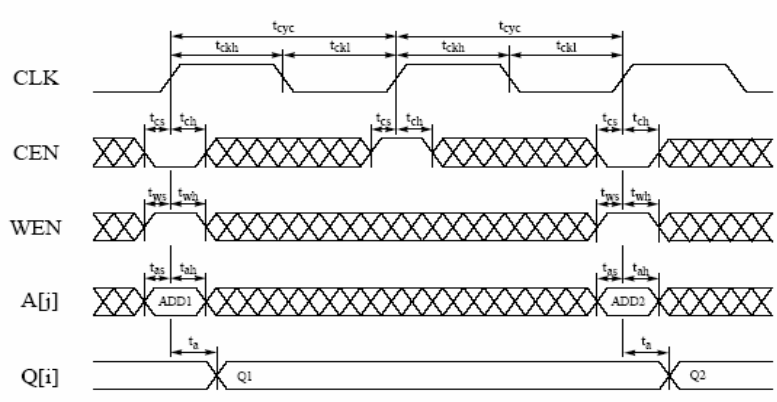


圖 2-19(a)：記憶體讀取時序圖

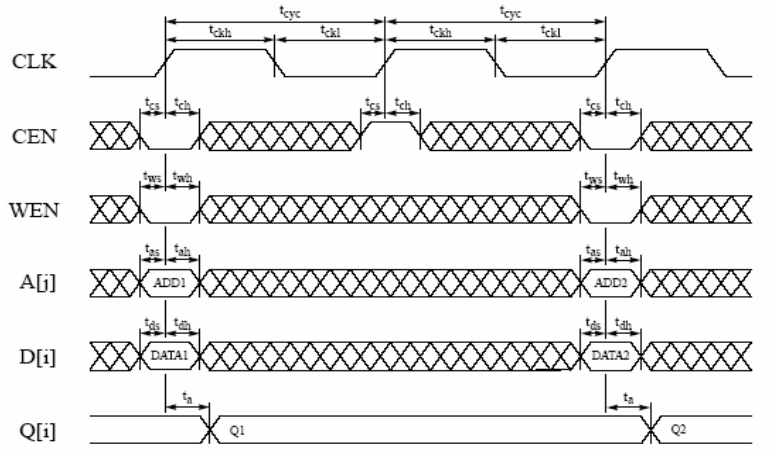


圖 2-19(b)：記憶體寫出時序圖

2.2.8 周邊匯流排（APB – Advanced Peripheral Bus）

APB 匯流排為 ARM 的 AMBA（Advanced Microcontroller Bus Architecture）[8]架構下的一部分，其受控端介面訊號如圖 2-20 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（I^2S），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。

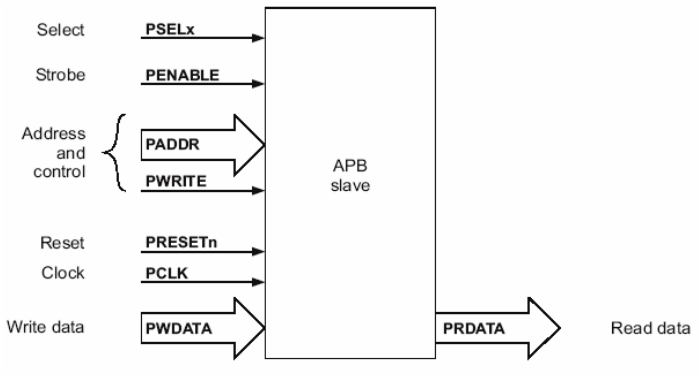


圖 2-20：APB 匯流排受控端介面

APB 匯流排時序如圖 2-21，一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。

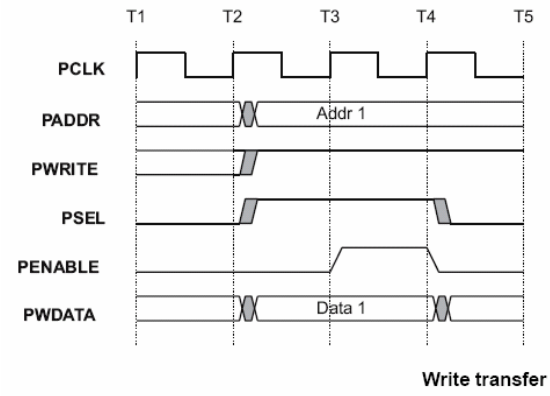


圖 2-21(a)：APB 匯流排寫入時序圖

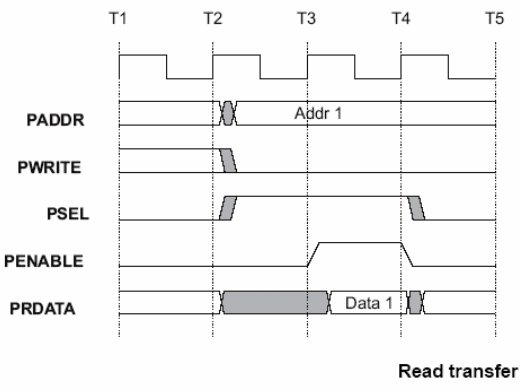


圖 2-21(b)：APB 匯流排讀取時序圖

2.3 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。

2.3.1 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 2-3，位元排列說明如下：

- Source Circular：決定環狀定址（0）或鏡射定址（1），若不需上述兩種定址，只要將環狀或鏡射定址的區塊設為零即可。
- Source Base：決定索引定址要遞增或遞減的位址數，亦可設為零，表示只存取同一位址。
- Source Offset：如圖 2-22 所示，決定環狀或鏡射定址的第一筆資料離區塊起點的數目。

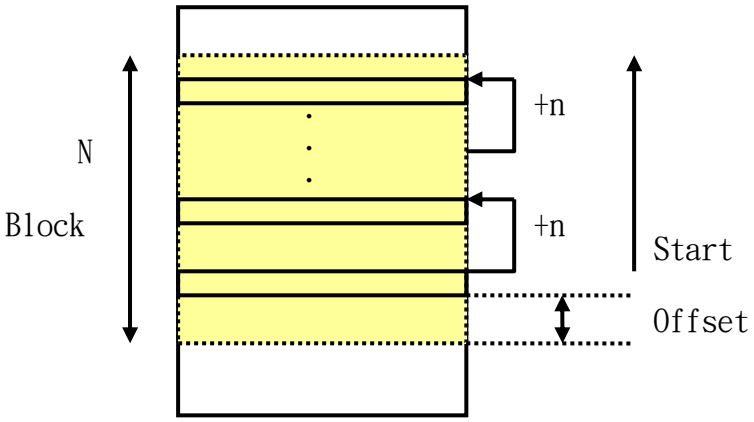


圖 2-22：Source Offset 示意圖

- Source Device：決定要存取的記憶體。若為周邊則須符合 APB 匯流排的定址方式。

- **Source Address**：決定存取的位址。當存取記憶體時，因記憶體容量為 512x32，位址線只有 9-bit 有效。

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

表 2-3：來源暫存器（Source Register）

2.3.2 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 2-4 所示。

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

表 2-4：目的暫存器（Destination Register）

2.3.3 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 2-5，位元排列如下：

- **Source/Destination Region**：來源/目的之環狀或鏡射定址區塊大小。控制暫存器的低位元為控制設定：
- **Source Increase**：來源位址遞增（Increase=1）。
- **Source Decrease**：來源位址遞減（Increase=0 & Decrease=1）。
- **Destination Increase**：目的位址遞增（Increase=1）。
- **Destination Decrease**：目的位址遞減（Increase=0 & Decrease=1）。
- **Source/ Destination Width**：寬度（保留）。
- **Transfer Size**：傳送資料筆數。

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

表 2-5：控制暫存器（Control Register）

2.3.4 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 2-6，位元排列說明如下：

- Halt：通道讀取工作暫停，並將 FIFO 內資料寫出後暫停。
- Interrupt Enable：中斷致能。
- Source Peripheral：來源周邊，最多八組周邊裝置。
- Destination Peripheral：目的周邊，最多八組周邊裝置。
- Transfer Type：傳輸模式（0：RMA/ 1：I/O）（來源-目的）。
- Sequence Transfer：連續傳輸，傳輸資料無限大（ST=1）。
- Function：特殊功能（乘加：001）。
- ACC Clear：累積器清除（AC=1）。
- Channel Enable：通道致能。

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

表 2-6：配置暫存器（Configuration Register）

2.3.5 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 2-7，位元排列說明如下：

- Interrupt：中斷（Low-Active）。
- FIFO Full：可監控緩衝器的狀態。
- FIFO Empty：可監控緩衝器的狀態。
- FIFO Half：可監控緩衝器的狀態。
- Channel Select：仲裁器決定通道是否工作。
- Error：錯誤狀態（累加器溢位發生）。

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）

表 2-7：狀態暫存器（Status Register）

2.3.6 累積暫存器（ACC Register）

累積器對外為一組 32-bit 可讀寫暫存器，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 2-8。

31:0	Accumulator	累加器

表 2-8：累加器（Accumulator）

2.3.7 智慧型 DMA 使用流程

智慧型 DMA 的設定方式如圖 2-23，將來源暫存器、目的暫存器及控制暫存器設定後，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。

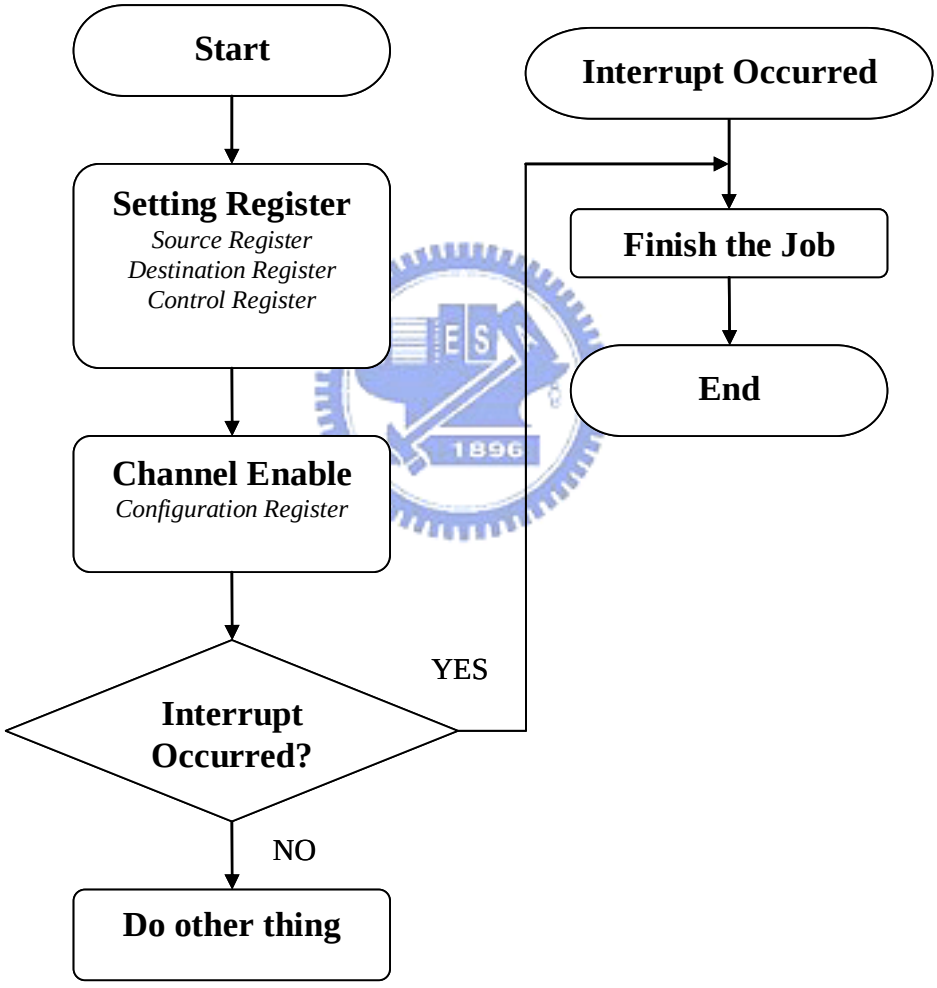


圖 2-23：智慧型 DMA 的使用流程

第三章 智慧型 DMA 控制器與 VLIW 處理器的整合

智慧型 DMA 控制器不能單獨存在運作，需搭配一個處理器成為一個具有 DSP 能力的處理器。因此，本論文也發展出一個 64 位元超長指令集架構（VLIW）處理器核心，與智慧型 DMA 控制器整合，以呈現完整的應用。以下章節將說明此超長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。

3.1 VLIW 處理器

本論文設計的 VLIW 訊號處理器擁有七級管線架構的設計[9][10]，為一個 Multi-core 和 Multi-Ram 的處理器 IP[11][12]，Multi-core 可以是 DSP 運算單元、或微控制單元。當一個指令透過程式計數器（Program counter）抓取進入處理器內部後，會先進入預先解碼（Pre-decoder），在這一級將一個 64bit 的 VLIW 長指令做拆解，成為兩個 32bit 的指令，並決定兩個 32bit 指令分別應該進入哪一個 core 中執行。當指令進入某一個核心時，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[13]，周邊裝置(或 IP)可利用標準的 APB 規格掛上匯流排，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[14][15]，架構如圖 3-1。

3.1.1 VLIW 處理器核心

VLIW 架構訊號處理器擁有七級管線架構，七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：

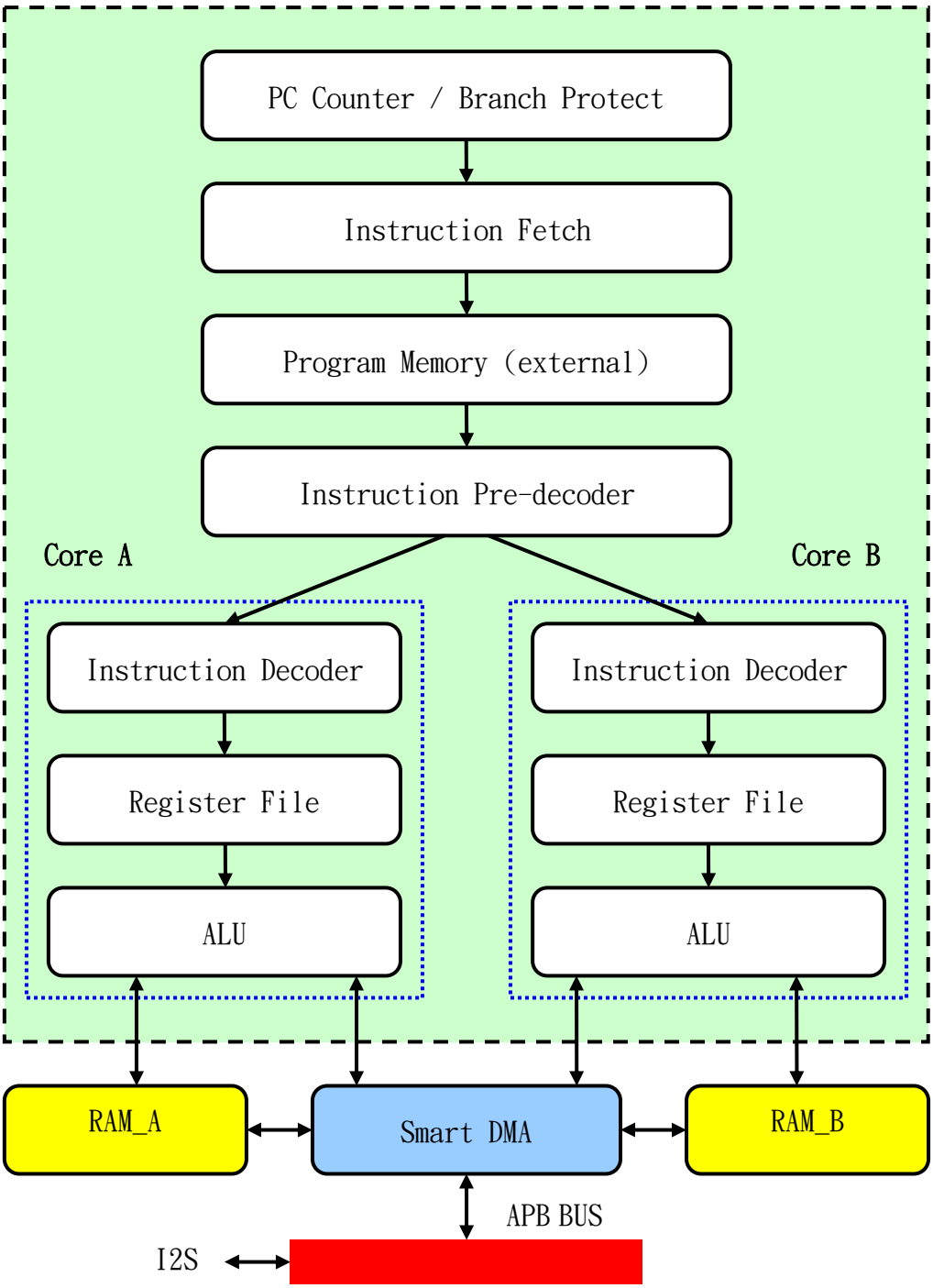


圖 3-1：VLIW 架構多媒體訊號處理器的組成結構

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. 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。
3. 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 VLIW 指令為 64-bit。
4. Instruction Pre-decoder：這一級的目的是把一個 64-bit 的 VLIW 長指令，切割成兩個 32-bit 的指令分別交給下面的兩個處理器核心做運算。由於整個設計都是朝 IP 化的原則，所以當增加新的 IP 處理核心，尤其是新增不同類的處理器核心時，本級電路會判斷不同的指令運算適合什麼樣的核心去處理。指令碼也在此級被抓取，判斷怎樣的指令要交給哪一個處理核心執行需要指令碼去進行判斷，指令碼的提前抓取也很方便在下一級的指令解碼器中做作解碼的動作。
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。由於此處理器整合智慧型 DMA 模組的設計，因此新增一些智慧型 DMA 控制指令。還需

提供智慧型 DMA 一組存取控制暫存器的位址訊號線，以決定智慧型 DMA 的控制參數要寫到哪一個智慧型 DMA 控制暫存器內。

6. **Register File**：每個處理核心有 32 個通用暫存器。處理核心 A 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[16]，包含以下五種說明：

1. **Bit Reverse**:針對 FFT 運算所增加的 memory 定址模式,例如位址(01101)可被轉成(10110)的位置儲存。
2. 一個指令週期完成乘加運算。
3. **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。
4. 良好的 **Data Forwarding** 機制。
5. **Condition Branch**：預測採用 Prediction-untaken 方法設計。

3.1.2 VLIW 處理器指令集

VLIW 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：**

Instruction	Opcode	Example	Mode
MOVRC	000001	MOV rd,data	Direct
MOVRR	000010	MOV rd,rs	Reg-Reg
MOVRM	000011	MOV rd,address	Direct
MOVMR	000100	MOV address,rs	Direct
MOVMRR	000101	MOV @rs2,rs	Indirect
MOVRRM	000110	MOV rd,@rs	Indirect
MOVARR	100010	MOV rd(a),rs(b)	Reg-Reg
MOVB	101111	MOVB rd,base(rs)	Displacement
MOVI	110000	MOVI rd,rs1(rs2)	Index
MOVREVRM	101010	MOV rd,address	Bit Reverse
MOVREVMR	101011	MOV address,rs	Bit Reverse
MOVREVMRR	101100	MOV @rs2,rs	Bit Reverse
MOVREVRRM	101101	MOV rd,@rs	Bit Reverse

表 3-1：資料搬移指令列表

VLIW 處理器共提供 Direct、Reg to Reg、Indirect、Displacement (base add)、Index、 Bit reverse 六大類的定址模式，其中 MOVARR 可以互相提取不同處理器核心的通用暫存器的內容。

■ 算數與邏輯運算指令 ：

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

表 3-2：算數邏輯運算指令列表

■ 跳躍指令 ：

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

表 3-3：跳躍指令列表

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：**

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

表 3-4：其他指令列表

SET 指令是當 VLIW 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 ：

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

表 3-5：智慧型 DMAC 控制指令列表

3.2 音源介面（I²S-IC Sound Bus）

因應多媒體應用上音源介面的需求，設計 I²S 音源介面[17]，提供串列音源資料的輸入與輸出。I²S 的設計包含三個訊號線：SCK、WS 及 SD，說明如下：

- SCK（Serial Clock）：每個位元傳送的時脈，通常和取樣頻率相同。
- WS（Word Select）：左右聲道切換的訊號（WS=0，左聲道；WS=1，右聲道），若以每聲道資料 32-bit 來計算，WS 周期是 SCK 的 64 倍。
- SD（Serial Data）：串列資料，接收端在 SCK 正緣收資料，而傳送端在 SCK 負緣送資料。

I²S 為單向資料傳遞設計，一組傳送端、一組接收端，在傳送端部分，以受控者（Slave）角色傳送資料，SCK、WS 由外部接收端發送，因此取樣率由外部接收端決定。如圖 3-2：

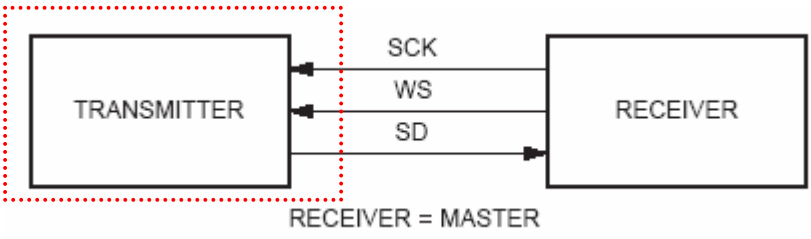


圖 3-2：I²S 傳送端

在接收端部分，以受控者（Slave）角色接收資料，SCK、WS 由外部傳送端發送，因此取樣率由外部傳送端決定。圖 3-3：

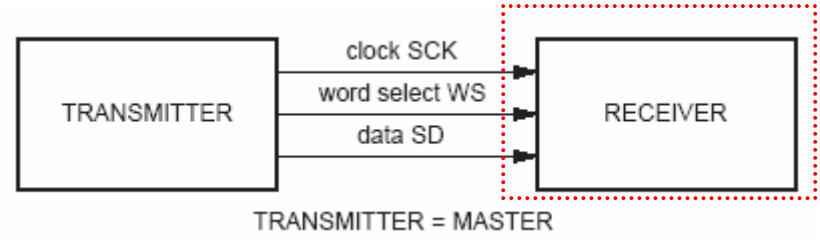


圖 3-3：I²S 接收端

傳送及接收需符合 I²S 的時序圖，主要使用位移暫存器實現 I²S 介面。

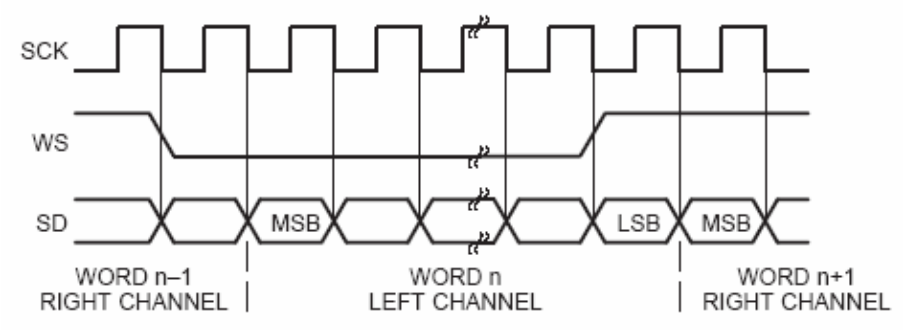


圖 3-4：I2S 時序圖

I²S 對內符合 APB 匯流排的規格。在匯流排上，視為兩個暫存器，只要將資料丟入暫存器，傳送或接收的電路立即啟動。

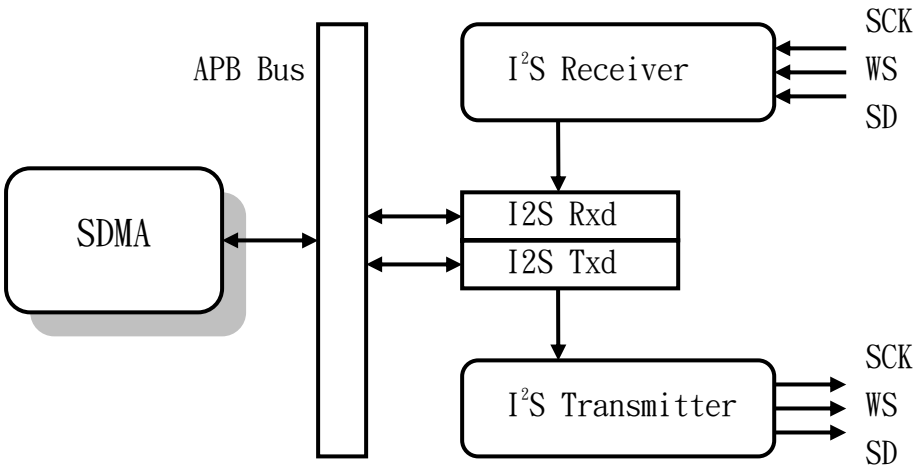


圖 3-5：APB 匯流排上的 I²S

3.3 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[13](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。

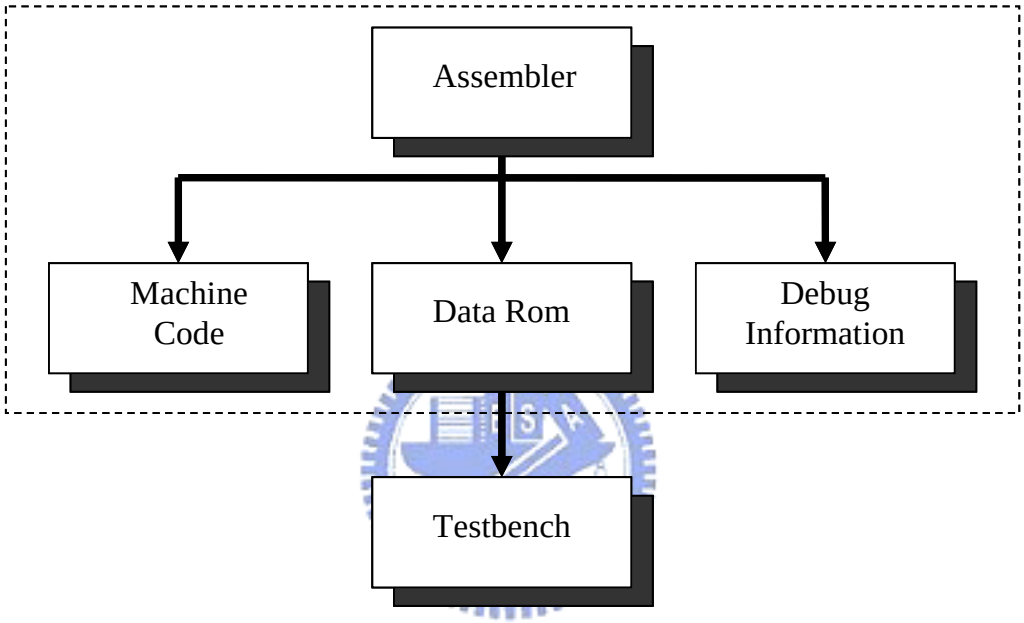


圖 3-6：組譯器（Assembler）的用途

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：產生程式記憶體的資料，提供模擬及後續測試必備的資料。

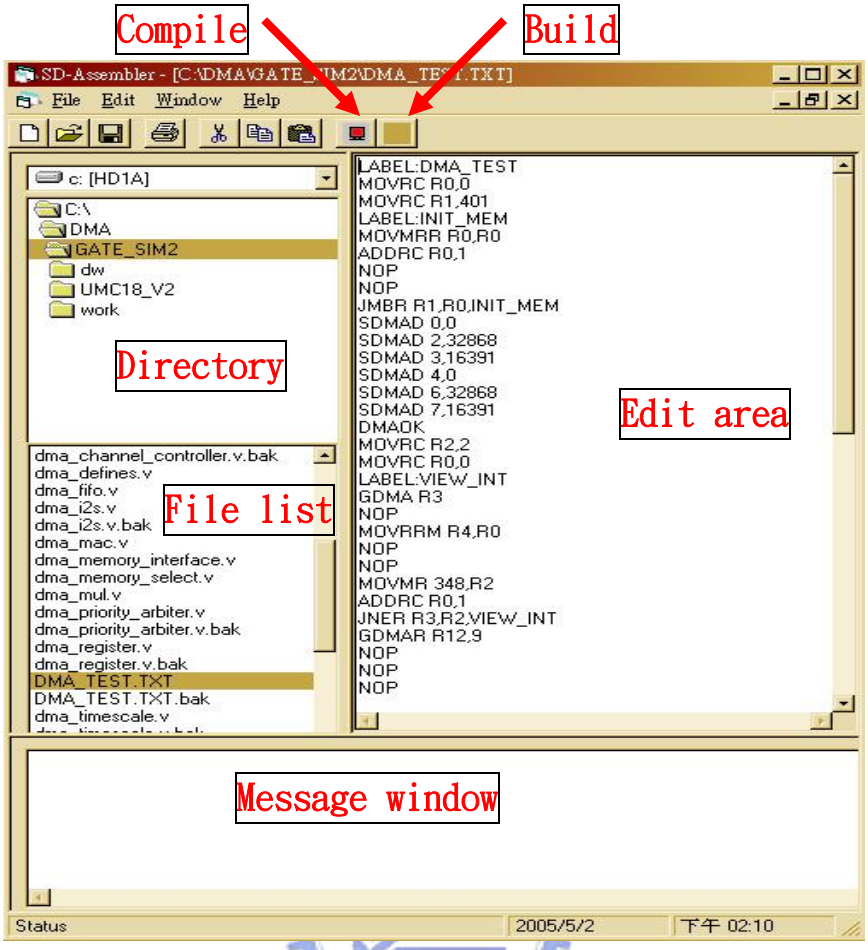


圖 3-7：圖形化介面組譯器（Assembler）

VLIW 處理器使用超長指令集的設計核心，在編寫程式及組譯時須注意：

- 平行處理：平時以處理核心 A 為主，若需使用處理核心 B，則需在兩指令間加上分隔符號『||』，若不需有處理核心 A，則須指令 B 前加上『NOP』指令，如下圖：

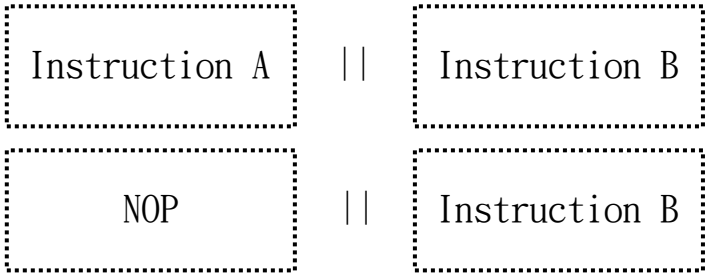


圖 3-8：平行處理指令的使用

3.4 整合

本論文將智慧型 DMA 和 VLIW 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 3-9，VLIW 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。

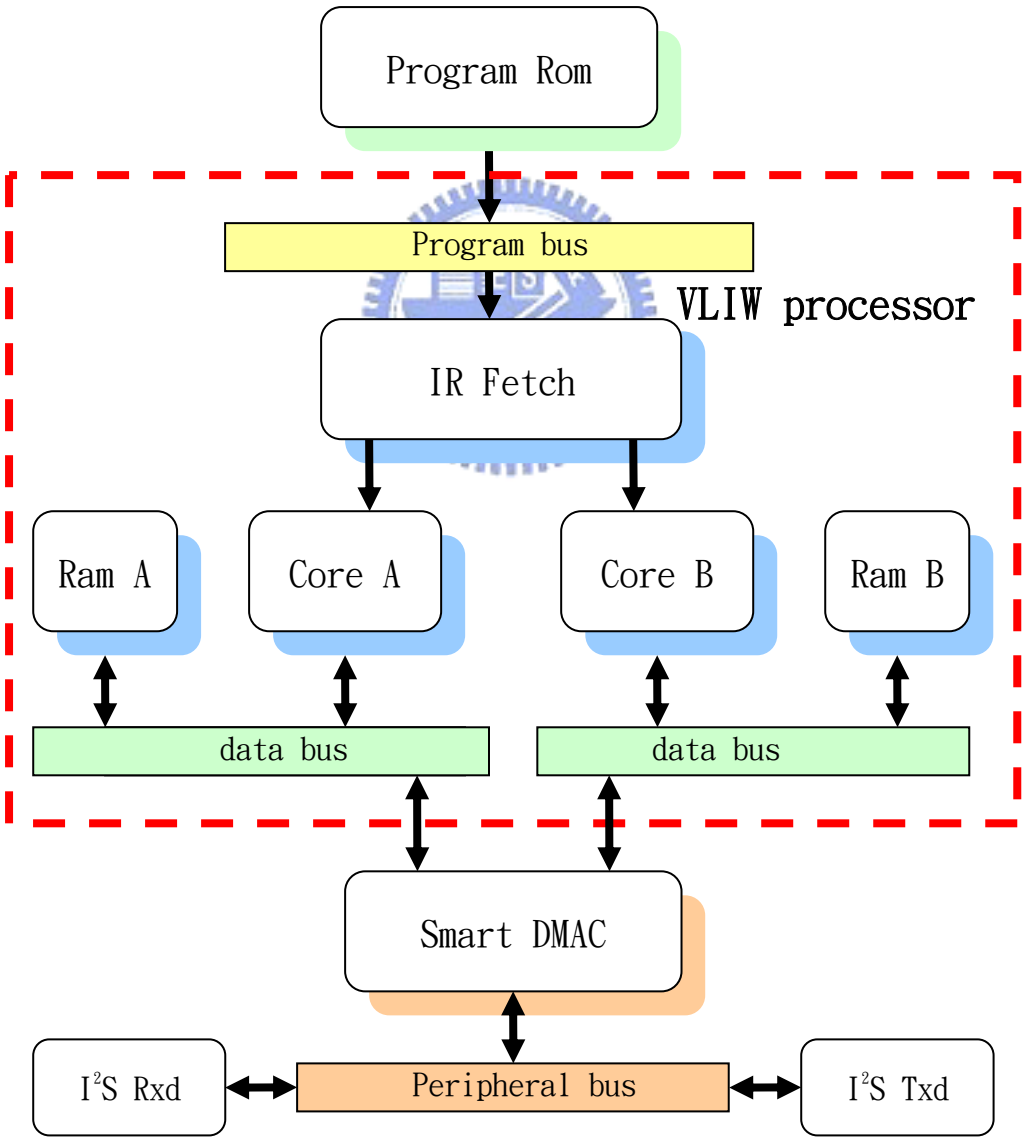


圖 3-9：VLIW 處理器與智慧型 DMA 共用匯流排

整個系統共有四條分開的匯流排，智慧型 DMA 與處理器共用二條資料匯流排，透過資料匯流排讓處理器與智慧型 DMA 共用兩個資料記憶體。資料匯流排的管制與衝突，由 VLIW 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。

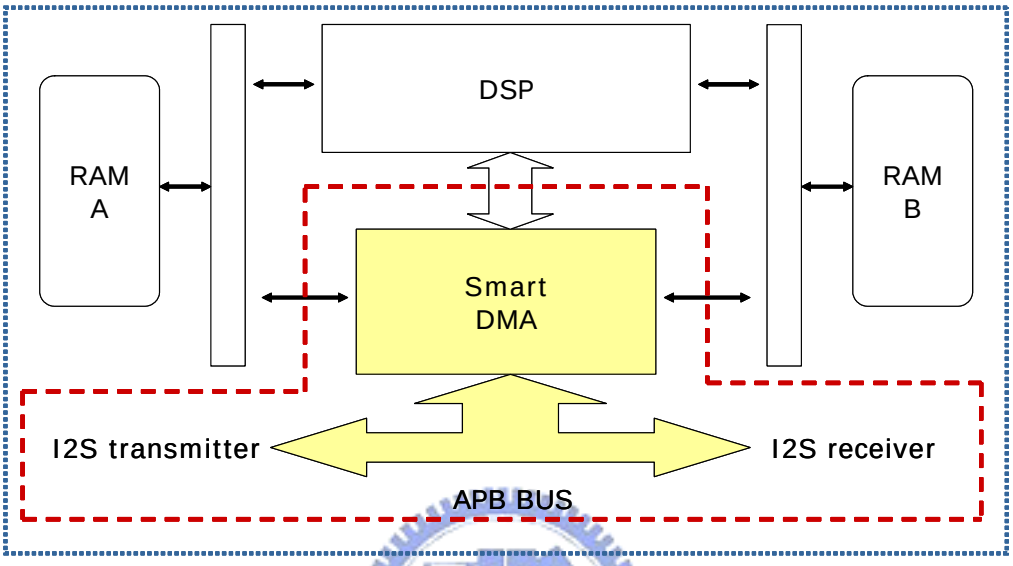


圖 3-10：VLIW 處理器與智慧型 DMA 的整合

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：

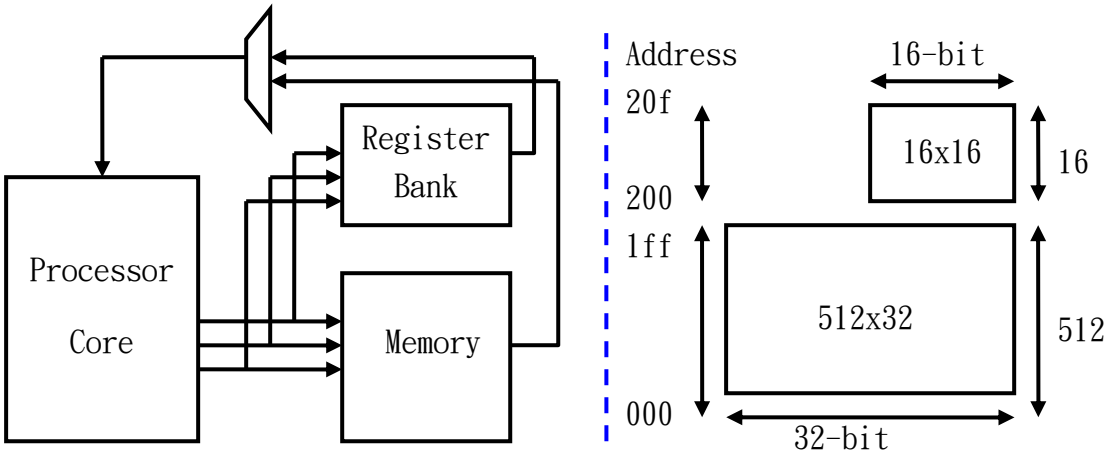


圖 3-11：VLIW 處理器存取智慧型 DMA 暫存器-記憶體對映

第四章 驗證結果與晶片實現

本章詳述智慧型 DMA 整合在 VLIW 處理器中，其功能上的驗證方法及結果，，並提及晶片實現的部分，最後比較處理器結合智慧型 DMA 的功能及效能。

4.1 設計驗證與結果

在組譯器上寫一系列測試程式，用以驗證 VLIW 處理器加上智慧型 DMA 的功能和效能，主要有下列項目：資料搬移、向量內積、褶積、餘弦轉換、離散小波轉換，詳述如下：

4.1.1 資料傳輸（Data Transmission）

資料搬移選擇四種型態，主要驗證 VLIW 處理器的架構下記憶體間的傳輸正確性。

1. **不同記憶體間的傳輸**：如圖 4-1，驗證在兩個不同記憶體之間的傳輸是否正確。

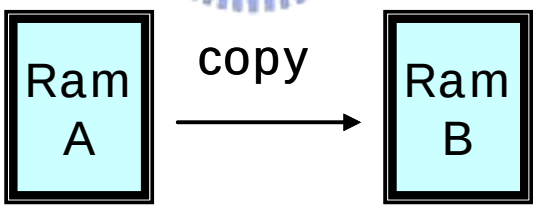


圖 4-1：不同記憶體間的傳輸

2. **同一記憶體裡的傳輸**：如圖 4-2，驗證在同一個記憶體裡的傳輸是否正確。

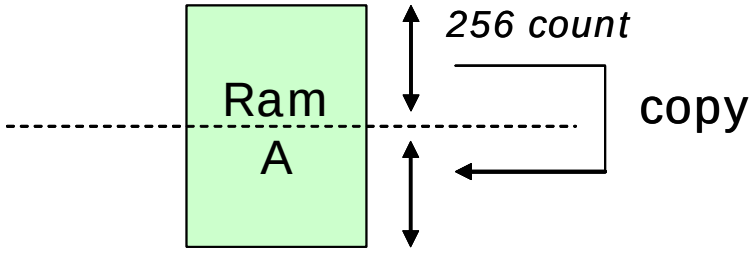


圖 4-2：同一記憶體裡的傳輸

3. **不同記憶體傳輸的環狀定址：**如圖 4-3，以環狀定址方式抓取來源資料，再存入目的記憶體，驗證結果是否正確。

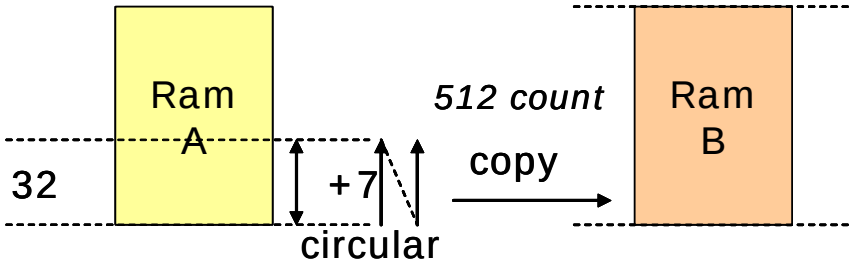


圖 4-3：不同記憶體傳輸的環狀定址

4. **不同記憶體傳輸的鏡射定址：**如圖 4-4，以鏡射定址方式抓取來源資料，再存入目的記憶體，驗證結果是否正確。

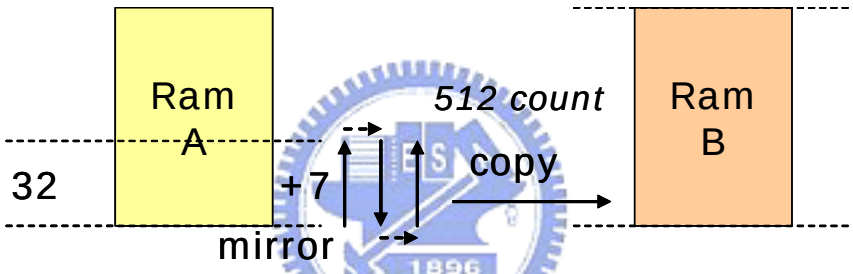


圖 4-4：不同記憶體傳輸的鏡射定址

4.1.2 向量內積（Inner Product）

如圖 4-5，每個位址存放的資料和其位址相同，以此計算兩個記憶體相對應值的乘加運算，共 512 筆的向量內積：

$$\sum_{i=1}^{511} MemA[i] \times MemB[i] = 44608256$$

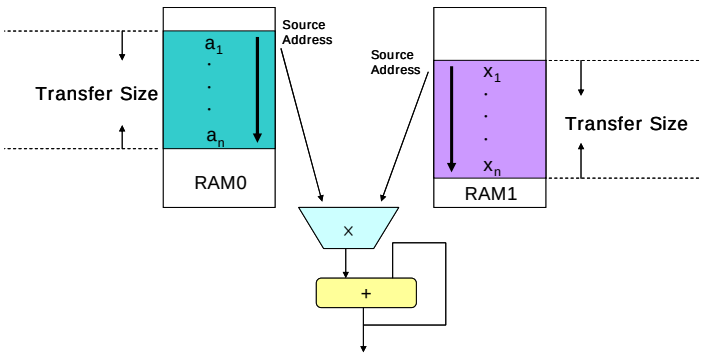


圖 4-5：向量內積驗證示意圖

圖 4-6：向量內積 Post-layout Simulation 結果

結果如圖 4-6，資料運算正確。

4.1.3 褶積（Convolution）

如圖 4-7，每個位址存放的資料和其位址相同，在兩個記憶體間做褶積的運算，其中一個記憶體位址遞增，另一個記憶體位址遞減。結果如圖 4-8，資料運算正確。

$$\sum_{i=0}^{511} MemA[i] \times MemB[(511-i)] = 22238720$$

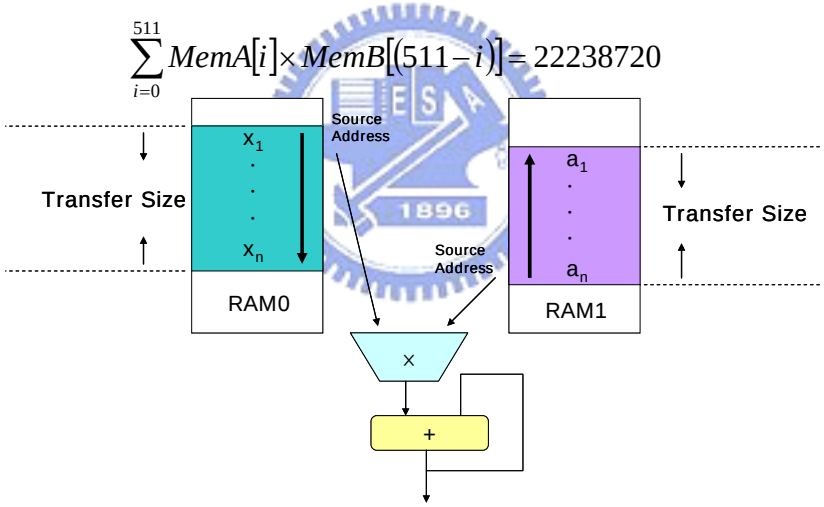


圖 4-7：褶積驗證示意圖

圖 4-8：褶積 Post-layout Simulation 結果

4.1.4 餘弦轉換（Discrete Cosine Transform）

採用定點數平移 13-bit 做 36 點 DCT-2 運算，結果如圖 4-9，共有 36 筆資料輸出。其數學式如下：

1-D 36-point DCT-2

$$X[k]=\sqrt{\frac{2}{N}}\beta[k]\sum_{n=0}^{N-1}x[n]\cos(\frac{\pi k(2n+1)}{2N}),0\le k\le N-1 \qquad \beta[k]=\begin{cases}1/\sqrt{2}\ .....k=0\\ 1.....1\le k\le N-1\end{cases}$$

輸入 X：

```
Columns 1 through 18

   1    1    1    1    2    2    2    2    3    3    3    3    4    4    4    4    5    5

Columns 19 through 36

   5    5    6    6    6    6    7    7    7    7   18   18   18   18 -100 -100  100  100
```

[71]	71	5917
[70]	70	-1704
[69]	69	7092
[68]	68	-14485
[67]	67	33029
[66]	66	-61415
[65]	65	103246
[64]	64	-154031
[63]	63	209000
[62]	62	-260075
[61]	61	298022
[60]	60	-317656
[59]	59	312576
[58]	58	-285519
[57]	57	234471
[56]	56	-170299
[55]	55	83082
[54]	54	0
[53]	53	-90493
[52]	52	202875
[51]	51	-305511
[50]	50	407942
[49]	49	-490476
[48]	48	550088
[47]	47	-572744
[46]	46	557738
[45]	45	-504600
[44]	44	422795
[43]	43	-327352
[42]	42	229330
[41]	41	-149119
[40]	40	82295
[39]	39	-54265
[38]	38	19501
[37]	37	-135562
[36]	36	251160

圖 4-9：DCT-2 Post-layout Simulation 的運算結果

如圖 4-10，使用 MATLAB 軟體同樣去運算 DCT-2，和採定點數的智慧型 DMA 運算模擬結果差距非常小。

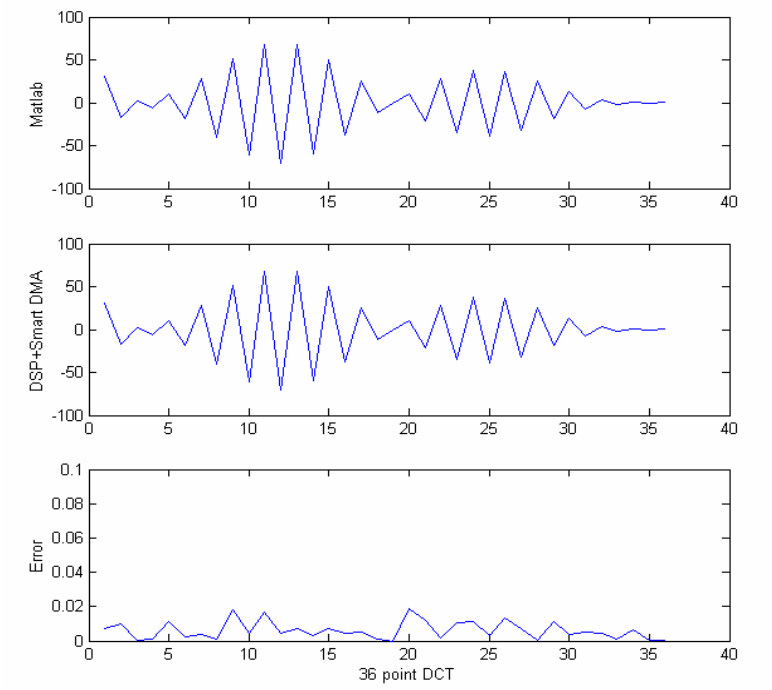
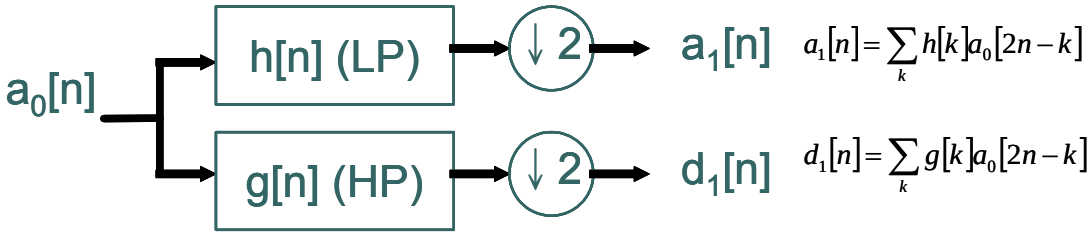


圖 4-10：DCT-2 模擬結果與 MATLAB 運算結果比較

4.1.5 離散小波轉換（Discrete Wavelet Transform）

離散小波轉換（DWT）主要由兩個 FIR 濾波器組成，一個為高通濾波器，一個為低通濾波器，如下圖：



$$a_1[n]=\sum_k h[k]a_0[2n-k]$$

$$d_1[n]=\sum_k g[k]a_0[2n-k]$$

圖 4-11：離散小波轉換示意圖

濾波器係數採用 JPEG2000 標準的 Daubechies 9/7-tap filter，其有較好的效能及訊號響應。藉由離散小波轉換的模擬結果，驗證智慧型 DMA 能夠處理數位訊號處理常見的數學式。如圖 4-12，為小波轉換裡的高通濾波器及低通濾波器。圖 4-13 為離散小波轉換的結果。

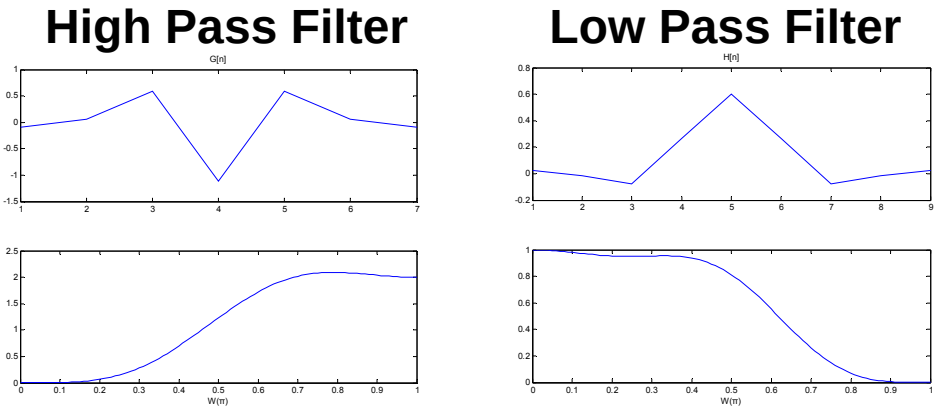


圖 4-12：離散小波轉換所採用之濾波器

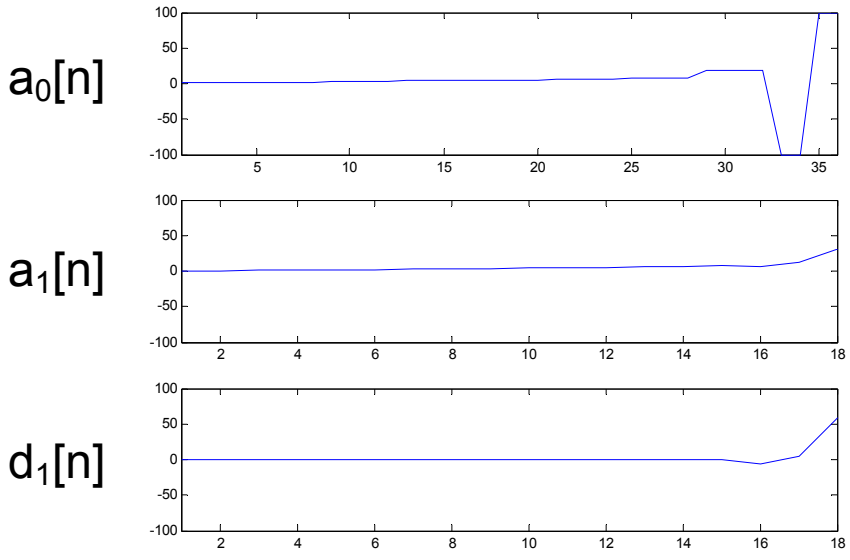


圖 4-13：離散小波轉換模擬結果

4.1.6 周邊匯流排（APB Bus）

APB 匯流排主要用做周邊裝置的傳輸、擴充，由於在多媒體處理上，非常需要音源介面來傳輸音源資料，因此，驗證方式：音源透過音源介面，經過匯流排，智慧型 DMA 再將資料丟到音源介面的輸出端，如下圖：

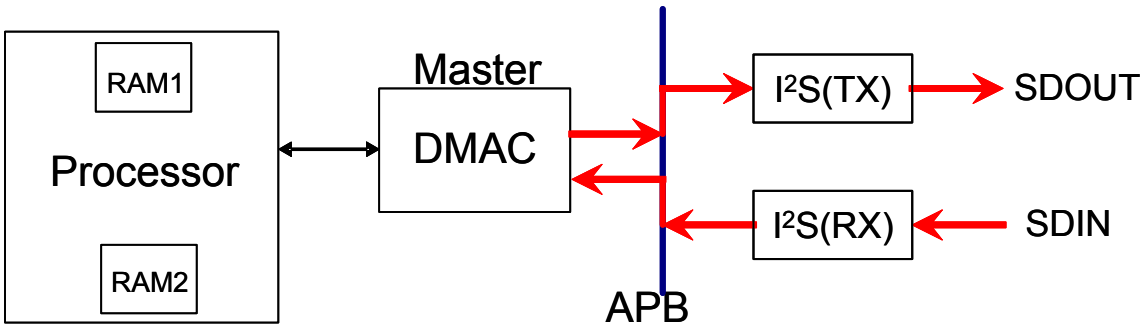


圖 4-14：APB 匯流排的驗證

透過音源介面（I^2S）的 Bypass 傳輸，可驗證 APB 匯流排及音源介面的正確性。當周邊的 I/O 擁有直接處理資料的能力，音源也能藉由此方式傳入。SDOUT 在開始傳送資料後，相隔一個 LRCLK 的間隔，輸出和 SDIN 的資料。

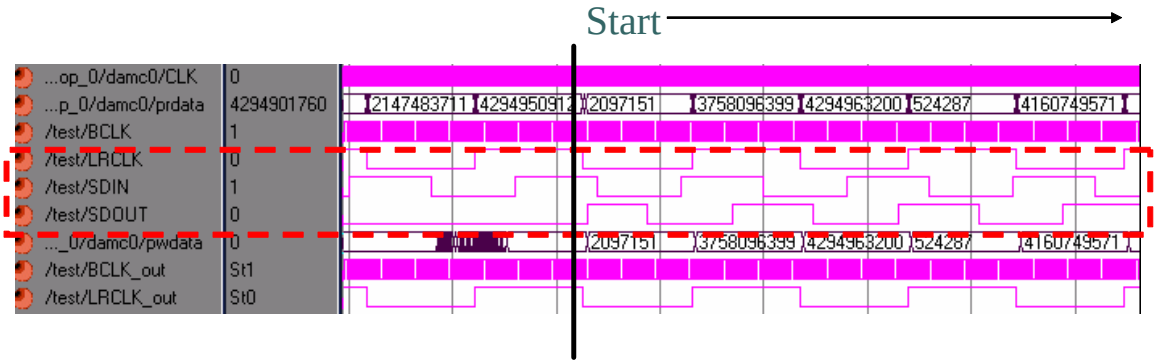


圖 4-15：音源介面 Bypass 傳輸的模擬

4.2 晶片製作

4.2.1 設計流程

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

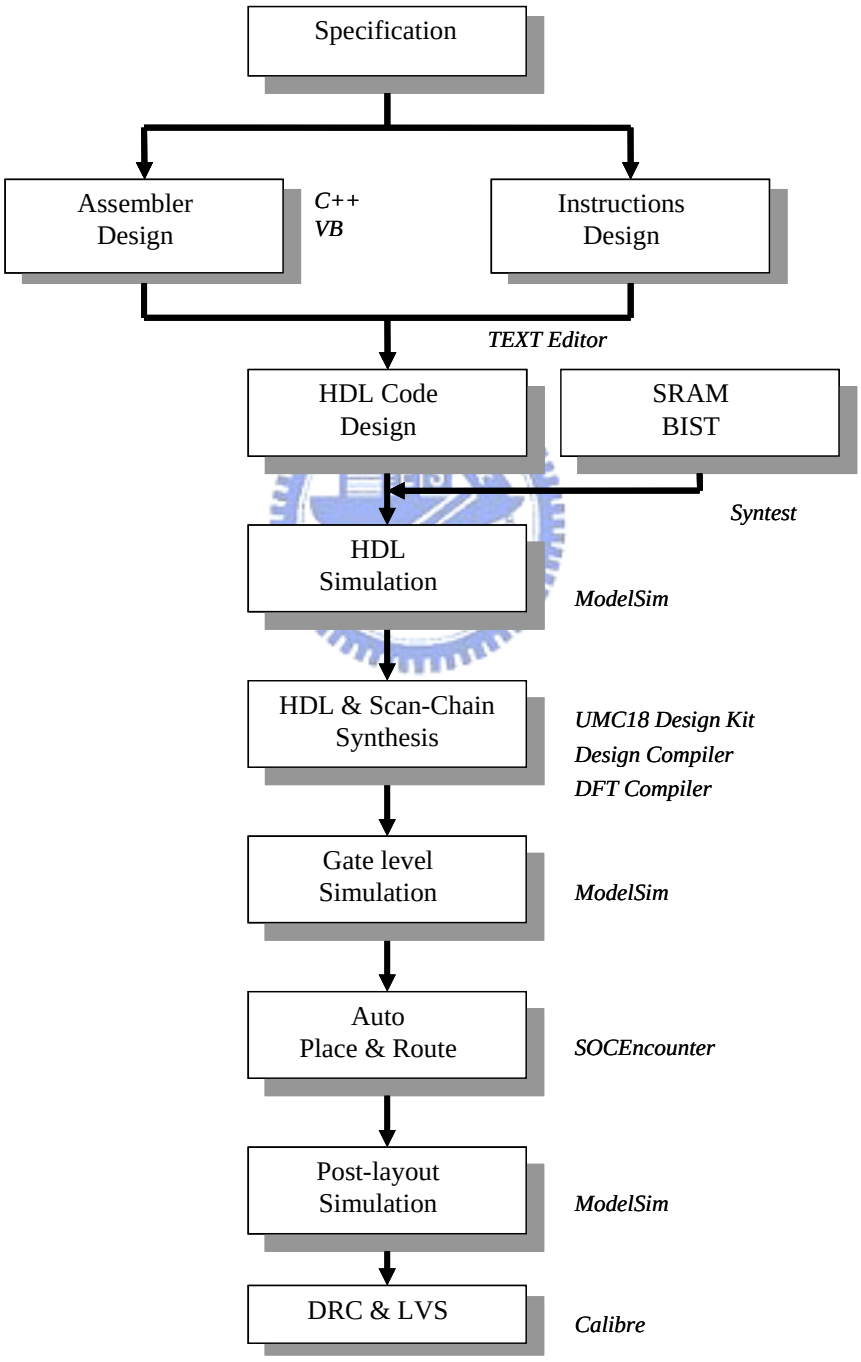


圖 4-16：晶片設計流程

4.2.2 合成結果

初期功能的設計，以 Verilog 硬體描述語言實作[18]，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[19]，Library 使用 UMC 0.18um 製程。經由合成軟體 Design Compiler 合成後，其結果如下：

ITEM	Area(mm^2)	Timing	Total fault	Fault coverage
Processor+SDMA	1181371	9.78 ns	299330	98.45 %

表 4-1：合成後的資訊

4.2.3 佈局與封裝

採用 Cadence 公司出的 SOC Encounter 佈局軟體[20]，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 ：128 CQFP

Chip Size ：2.376× 2.369 mm^2

Gate Count ：604K gate count

Power Dissipation ：~400mW

Max. Frequency ：128.5 MHz （7.781 ns）

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 4-17，為佈局及腳位圖，上方擺 VLIW 處理器核心 A 及連接至處理器核心 A 的記憶體，下方為 VLIW 處理器核心 B 及連接至處理器核心 B 的記憶體，中間放智慧型 DMA 的模組。如圖 4-18 為晶片打線圖，以 128-CQFP 的方式包裝。

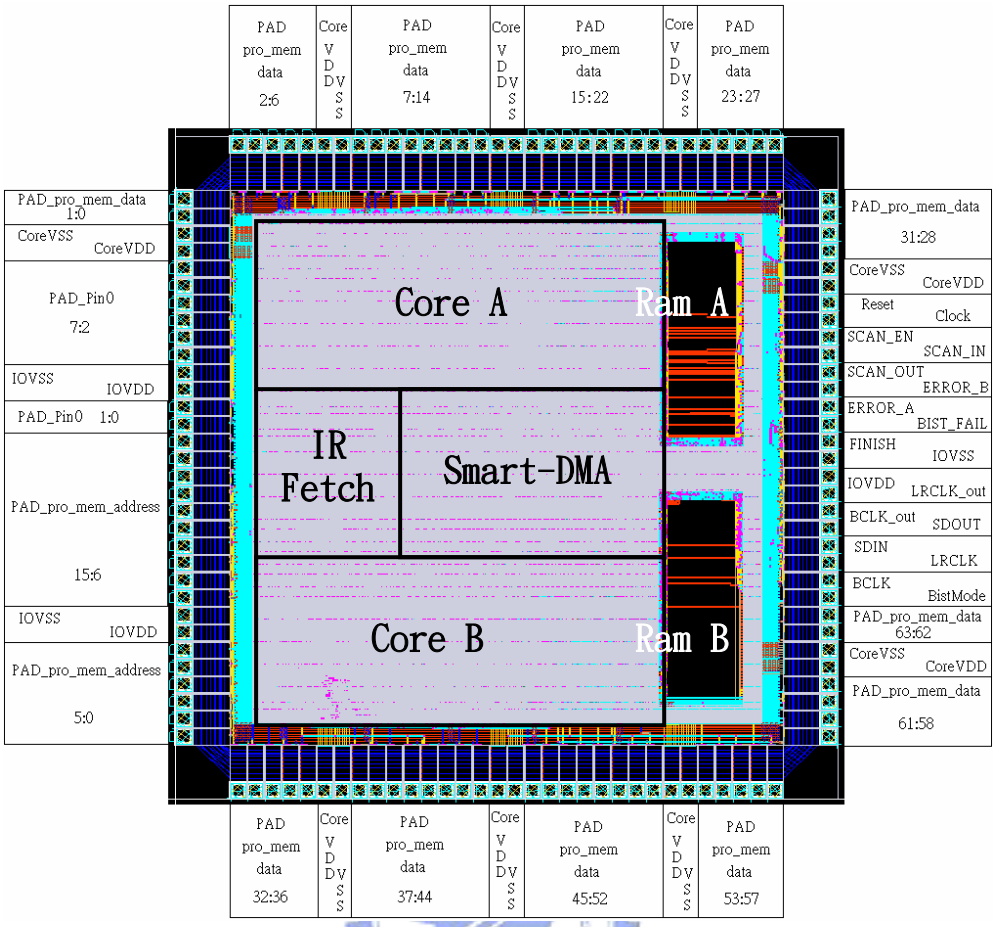


圖 4-17：佈局及腳位圖

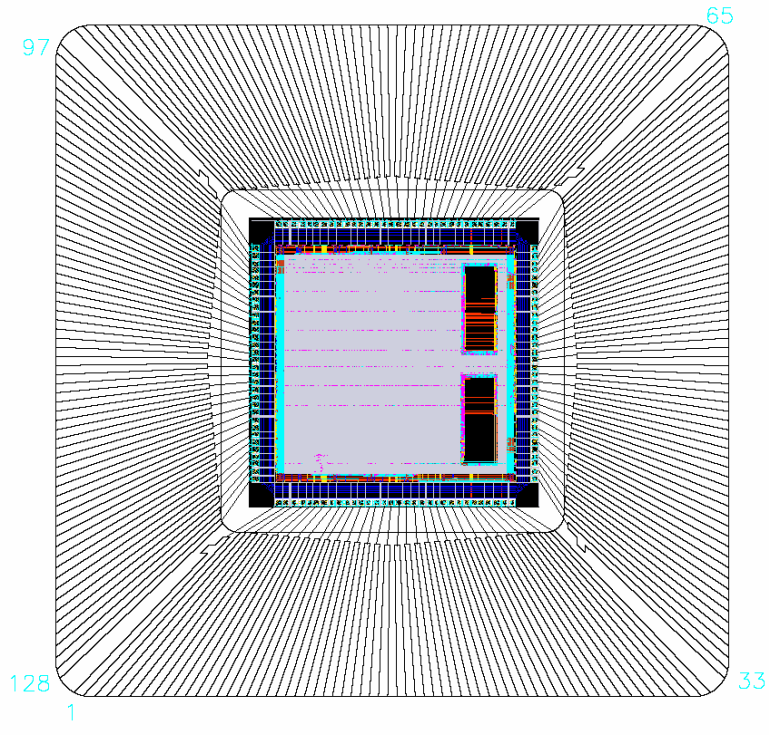


圖 4-18：128 CQFP 打線圖

而在後續的佈局驗證部分，DRC（Design Rule Check）在手動修改錯誤後，已正確無誤。LVS（Layout VS Schematic）共比對 48055 個點，也驗證無誤。

DRC	PASS（48 DRC error fixed）
LVS	PASS（48055 point）

表 4-2：DRC/LVS 驗證

晶片設計規格如表 4-3。

Technology	Description
Process	UMC 0.18μm 1P6M Mixed Signal
Architecture	VLIW 7-stage pipeline
Synthesis	Synopsys Design Compiler
Gate Count	604K
Embedded Memory	RAM0 (2KB)，RAM1 (2KB)
Die size	2.3 × 2.3 mm^2
Supply	1.8V/3.3V ± 10%
Input Delay Time	Max 0.714ns/ Min 0.543ns
Power consumption	400mW
Expectant Speed	100MHz
Pre-Sim Speed	102.3MHz
Post-Sim Speed	128.5MHz
DMAC Design	
DMA Channel	2
DMA Request	8
DMA Gate Count	33K
Transfer Type	Memory-to-Memory Memory-to-Peripheral Peripheral-to-Memory Peripheral-to-Peripheral
Compliance with APB bus	Support 8 device
Hardware DMA channels priority	High: Ch0 / Low:Ch1
DMA Special Function	Inner Product Convolution Multiple Addressing Built-in I2S interface

表 4-3：晶片設計規格

4.2.4 測試考量

針對晶片測試考慮兩大類，一為測試晶片在製造過程產生的錯誤，一為晶片功能上的測試。前者考慮的是記憶體自我測試的機制，及整體電路的 Scan-Chain 測試機制，說明如下：

由於目前製程愈來愈先進，單位面積可容納邏輯閘數目不斷增加，為防止在晶片製造過程中發生錯誤造成晶片損毀，必須加入可測試性電路，使得在早期即可得知晶片製造是否有問題。電路裡的測試電路，由 Synopsys 公司出的 DFT Compiler 產生。使用 DFT compiler 在設計中加上一條 Scan-Chain，測試資料 983 筆，如圖 4-19，整體 Fault coverage 達到 98.45%。

Uncollapsed Stuck Fault Summary Report

fault class	code	#faults
Detected	DT	294615
detected_by_simulation	DS	(260297)
detected_by_implication	DI	(34318)
Possibly detected	PT	154
atpg_untestable-pos_detected	AP	(138)
not_analyzed-pos_detected	NP	(16)
Undetectable	UD	902
undetectable-tied	UT	(69)
undetectable-redundant	UR	(833)
ATPG untestable	AU	3338
atpg_untestable-not_detected	AN	(3338)
Not detected	ND	321
not-observed	NO	(321)
total faults		299330
test coverage		98.75%
fault coverage		98.45%

圖 4-19：測試涵蓋率（Fault coverage）

在記憶體部分，如圖 4-20 使用 SyntestTM 的 Srambist 軟體在記憶體中加入自我測試機制，採用 Moving Inversion (13 N March) 演算法，來測試 SRAM 是否有損壞的地方。每一個記憶體有各自獨立的 BIST controller，並將 BistMode

訊號接在一起，因此在開啟 BIST 測試模式時，將同時開啟兩個記憶體的測試模式，測試結束時，Finish 也將同時結束。一旦測試有錯誤，BistFail 訊號將被設為 HIGH。

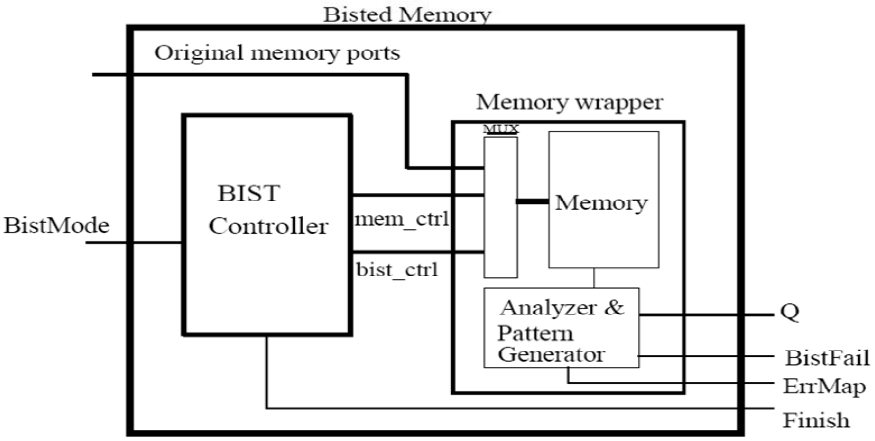


圖 4-20：加上自我測試電路（BIST）的記憶體模組

除了前項的測試用途外，為了晶片執行功能的正確性，提供了資料搬移、向量內積、褶積、一維 36 點餘弦轉換、順向小波轉換及 APB 匯流傳輸排測試六組。在 Post layout 上所驗證的 Pattern（如 4.1 節），並轉成在 IMS 機台可以使用的 Format，以驗證其晶片執行功能與 Post layout 後的正確性，透過 IMS 測試機台，可以在輸出的 Pad 上觀察到結果。

4.3 效能比較

市面上有種類相當多的 DMA 控制器[21][22]，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。本論文提出的智慧型 DMA 控制器，具有資料傳輸、多種傳輸模式，並擁有多樣定址能力，結合乘加運算單元，更具有輔助運算的能力。

	NCTU	FARADAY	GLOBAL UNICHIP	
	ECE	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 Circular Mirror Index-Based	Chain Transfer	Incrementing Non-incrementing Wrap-incrementing	
Special Function	Multiplication & Accumulator	None	None	

表 4-4：與市面上 DMAC 做比較

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。

4.3.1 資料傳輸效能

比較四種不同傳輸狀況，使用處理器搭配 DMA 與處理器搭配智慧型 DMA 的結果。如表 4-5，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。

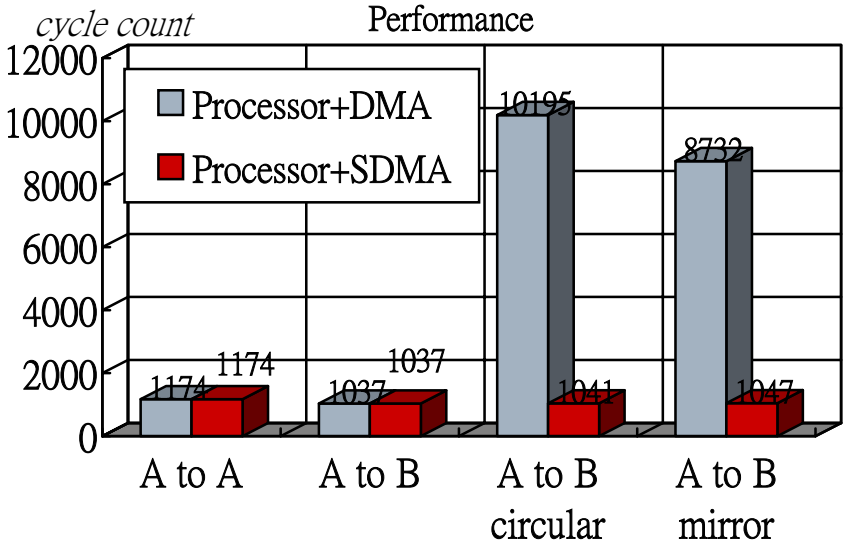


表 4-5：資料傳輸效能表

4.3.2 資料運算效能

如表 4-6，採用處理器加智慧型 DMA 的輔助運算，比起處理器使用 DMA 的資料搬移後再做計算，以下的四種運算情況下，速度有明顯地提升。

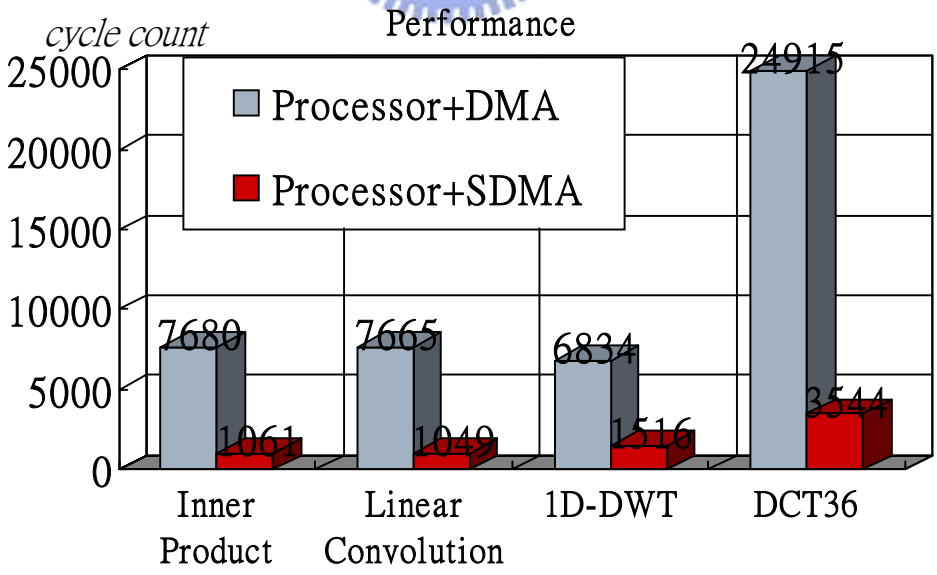


表 4-6：資料運算效能表

4.3.3 其他處理器比較

如表 4-7，以本論文設計的 VLIW 處理器，比起其他研究的數位訊號處理器（DSP）[23]，複雜度、成本都相當低。

Processor	Olofsson ISSCC2002	Agarwala ISSCC2002	TI C55x	**Ours**
S-P -Com MAC	1/4 cycle	1/4 cycle	2 cycle	**1 cycle**
S-P –Real MAC	1/8 cycle	1/8 cycle	1/2 cycle	**1/2 cycle**
Tech	.13um 8M	.13um 6M	-	**.18um 6M**
Area(mm^2)	10x10	8.5x8.5	-	**2.3x2.3**
Power(mw)	1000	718	-	**400**

表 4-7：與其他 DSP 的規格比較表

以原先設計的 VLIW 處理器來說，其效能遠遠不如 DSP，而在加上智慧型 DMA 後，其效能已追上一般的數位訊號處理器[24]，而成本遠低於數位訊號處理器。如表 4-8。

ITEM	50-taps FIR, 100 samples	50-taps complex FIR, 100 samples
NCTU ECE	11200 (cycles)	24000 (cycles)
A 32-b RISC/DSP Microprocessor	12200 (cycles)	22000 (cycles)
TI's C60	~ 5000 (cycles)	~ 20000 (cycles)

表 4-8：與 DSP 運算效能比較表

第五章 結論

智慧型 DMA 在多媒體系統上的發展，可以朝向整合 VLIW 處理器結合智慧型 DMA 成為一個完整系統，配合軟體開發環境，足夠在此平台上發展完整的多媒體應用，並針對多媒體上音訊、視訊的應用做開發，使其能呈現完整的多媒體應用能力。應用本論文的 VLIW 處理器達到多媒體處理的效能，配合智慧型 DMA 的架構，完成如圖 5-1，MP3 Encoder 應用的例子。虛線框起來的部分，可交由智慧型 DMA 來輔助運算，此時處理器可分攤其他方塊的工作，使得原先無法單獨處理 MP3 Encoder 工作的處理器，能夠 Real-Time 執行 MP3 Encoder 的功能，提升到 DSP 等級的能力。

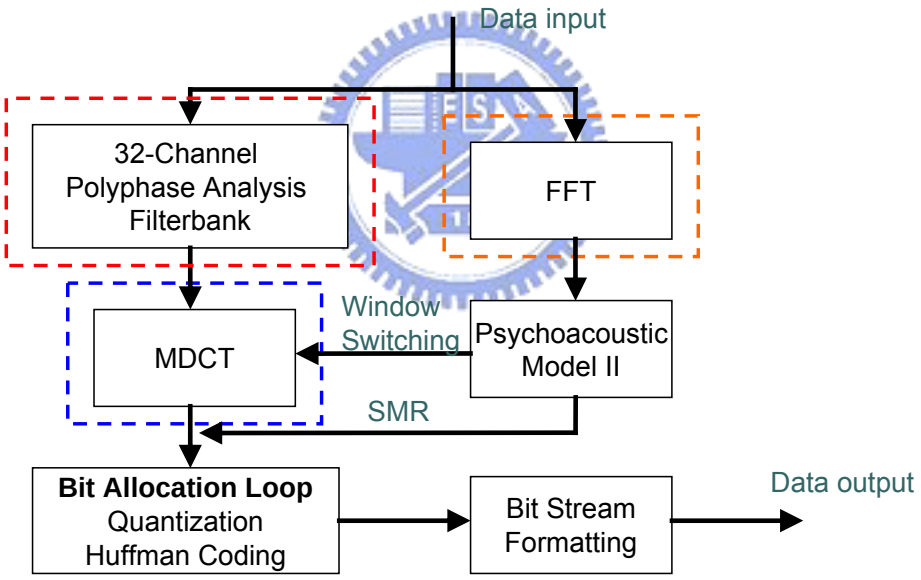


圖 5-1：MP3 編碼流程

根據圖 5-1，主要運算的方塊為 32 組濾波器、改良餘弦轉換及快速傅利葉轉換，本論文的功能驗證裡，已包含濾波器及餘弦轉換的實現，因此，上述功能可打包成一個 Function，直接透過智慧型 DMA 來處理，另外 FFT 的實現也可以利用智慧型 DMA 協助資料的排列，而處理器負責流程控制及其他方塊的實現。下圖為 MP3 Encoder 系統示意圖，音源由 A/D 透過 I2S 介面進入

CHIP 內，進到內部記憶體，經過 MP3 Eencoder 演算法的程式處理後，送到智慧型 DMA 透過 I/O Bus 存入外部的儲存體 CF card，最後可以將 CF 卡的 MP3 音源資料檔讀出用 Media Player 撥放。

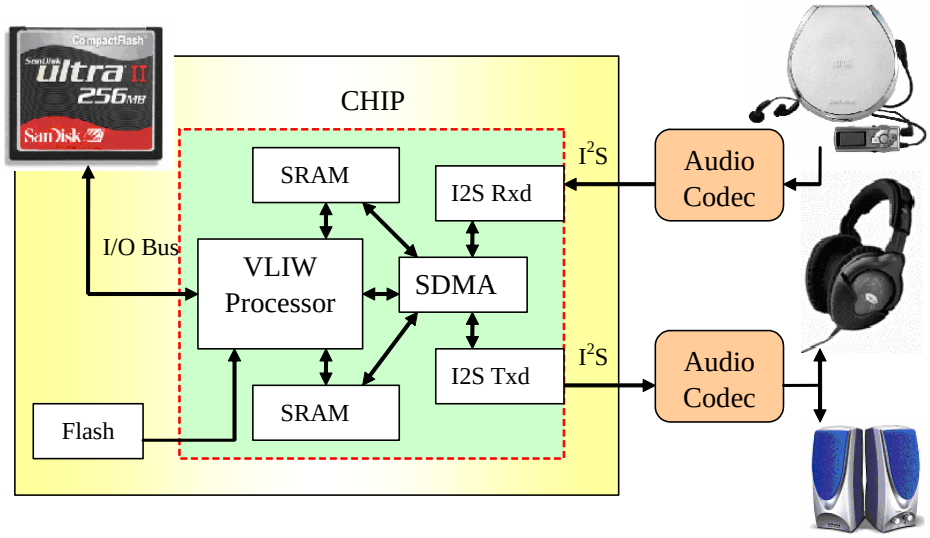


圖 5-2：應用-MP3 編碼、解碼

複雜的多媒體應用上，內建記憶體容量將不敷使用，未來可在匯流排掛上外部記憶體的控制器，使得在有大量記憶體需求的多媒體應用上，仍能發揮其強大的效能。由於數位訊號處理相當多樣化，可針對數位訊號處理，發展不同的定址模式及資料排列方式，以增強智慧型 DMA 的能力。此外，現有的微處理器已發展多時，效能普通卻有低成本的優勢，未來可發展小型微處理器結合智慧型 DMA，使得其成為更低成本兼具計算能力的解決方案。

本篇論文提出一個智慧型 DMA 控制器。在傳輸方面，支援所有的傳輸模式，且為了在取得資料時，有效率地選取資料，降低傳輸頻寬，增加速度，智慧型 DMA 控制器設計了四種定址方法：遞增/遞減（Increasing/Decreasing）、環狀定址（Circular Addressing）、鏡射定址（Mirror Addressing）及索引定址（Index-based Addressing）。而針對數位訊號處理，此四種定址方法配合乘加運算器（MAC），可達到輔助處理器的數位訊號處理功能。且同時支援雙通道資料記憶體的快速向量運算，擁有直接直援兩個記憶體的能力。周邊匯流排支援 APB

匯流排標準，內建常用的音源介面（I^2S），共可外掛八個 I/O 裝置，若有需要可在匯流排上加裝符合 APB 標準的介面，擁有記憶體到記憶體、記憶體到周邊、周邊到記憶體、周邊到周邊四種傳輸模式。而在加上 MAC 運算元後，僅增加 10%的 Gate Count，成本相當低。

在驗證方面，發展一顆 VLIW 處理器，透過和智慧型 DMA 的結合，讓原本處理器的效能達到數位訊號處理器的水準。VLIW 處理器結合智慧型 DMA 控制器晶片，已在國家晶片中心（CIC）下線，未來此顆晶片可用做低成本的數位訊號處理器來使用，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. Madisetti, VLSI Digital Signal Processors: An Introduction to Rapid Prototyping and Design Synthesis, IEEE Press, 1995.

[2] Luca Breveglieri and Luigi Dadda, “A VLSI inner product macrocell,” *IEEE Trans. on VLSI*, vol. 6, no. 2, pp. 292-298, 1998.

[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] C. M. Yuen, K. F. Tsang, and W. H. Chan, “Direct memory access frequency synthesizer for channel efficiency improvement in frequency hopring communication,” *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS 2000)*, pp.485-488, 2000.

[5] Mattias O’Nils and Axel Jantsch, “Synthesis of DMA controllers from architecture independent descriptions of HW/SW communication protocols,” *Proceedings of IEEE International Conference on VLSI Design (ICVD)*, 1999.

[6] ARM Ltd, PrimeCell Single Master DMA Controller (PL081), rev. r1p1, http://www.arm.com, Technical Reference Manual, 2003.

[7] Alan V. Oppenheim, Ronald W. Schafer, and John R. Buck, Discrete Time Signal Processing, 2nd Edition, Prentice Hall, 1999.

[8] ARM Ltd, AMBA Specification, rev. 2.0, http://www.arm.com, 1999.

[9] John L. Hennessy and David A. Patterson, Computer Architecture, 3rd Edition, Morgan Kaufmann, 2003.

[10] John L. Hennessy and David A. Patterson, Computer Organization & Design：The Hardware / Software Interface, 2nd Edition, Morgan Kaufmann Publishers,

1998.

[11] 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.

[12] C. G. Lee and M. G. Stoodley, “Simple vector microprocessors for multimedia applications,” *Proceedings of 31st Annual ACM/IEEE International Symposium*, pp. 25-36, Dec. 1998.

[13] A. Kalavade and E. A. Lee, “A hardware-software codesign methodology for DSP application,” *IEEE Design & Test of Computers*, vol. 10, pp. 16-28, Sept. 1993.

[14] Steve Fuber, ARM System-on-Chip Architecture, 2nd edition, Addison-Wesley Professional, 2000.

[15] 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.

[16] 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.

[17] Philips Semiconductor, I2S bus Specification, http://www.semiconductors.philips.com,1996.

[18] Donald E. Thomas and Philip Moorby, The Verilog Hardware Description Language, Kluwer Acadmic Publishers, 1994.

[19] Pran Kurup, et al., Logic Synthesis Using Synopsys, 2nd Edition, *K*luwer

Academic Publishers, 1997.

[20] Nian Shyang Chang, Cell-Based IC Physical Design and Verification, Chip Implementation Center, July, 2004.

[21] GlobalUnichip, “UAPC-5110 DMA Controller Lite,” http://www.globalunichip.com.tw, 2002.

[22] Faraday, “Direct Memory Access Controller: Faraday/UMC FTDMAC020,” rev. 1.2, www.faraday.com.tw, 2003.

[23] Yuan-Hao Huang, Hsi-Pin Ma, Ming-Luen Liou, and Tzi-Dar Chiueh, “A 1.1 G MAC/s Sub-Word-Parallel digital signal processor for wireless communication applications,” *IEEE Journal of Solid-State Circuits*, vol. 39, no. 1, 2004.

[24] 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.

附錄

A 測試程式

Memory A to Memory A	Memory A to Memory B
SDMAB 0,0_00000001_0000000	*SDMAB 0,1_00000001_0000000*
SDMAB 1,0_000000111111111	*SDMAB 1,0_000000111111111*
SDMAB 2,0_00000001_0000000	*SDMAB 2,0_00000001_0000000*
SDMAB 3,0_000000000000000	*SDMAB 3,1_000000000000000*
SDMAB 4,01_10_00_0100000000	*SDMAB 14,00000000_00000000*
SDMAB 5,0_1_000_000_00_0_000_1_1	*SDMAB 4,01_10_00_1000000000*
DMAOK	*SDMAB 5,0_1_000_000_00_0_000_1_1*
MOVRC R2,2	*DMAOK*
LABEL:A2A	*MOVRC R2,2*
GDMA R3	*LABEL:RAM_COPY*
JNER R3,R2,A2A	*GDMA R3*
	JNER R3,R2,RAM_COPY

Memory A to Memory B (Circular)	Memory A to Memory B (Mirror)
SDMAB 0,0_00000111_0000000	*SDMAB 0,1_00000111_0000000*
SDMAB 1,0_000000000000000	*SDMAB 1,0_000000000000000*
SDMAB 2,0_00000001_0000000	*SDMAB 2,0_00000001_0000000*
SDMAB 3,1_000000000000000	*SDMAB 3,1_000000111111111*
SDMAB 14,00100000_00000000	*SDMAB 14,00100000_00000000*

SDMAB 4,10_10_00_1000000000	*SDMAB 4,10_01_00_1000000000*
SDMAB 5,0_1_000_000_00_0_000_1_1	*SDMAB 5,0_1_000_000_00_0_000_1_1*
DMAOK	*DMAOK*
MOVRC R2,2	*MOVRC R2,2*
LABEL:A2B_circular	*LABEL:A2B_mirror*
GDMA R3	*GDMA R3*
JNER R3,R2,A2B_circular	*JNER R3,R2,A2B_mirror*

Inner product	Convolution
SDMAB 0,0_00000001_0000000	*SDMAB 0,0_00000001_0000000*
SDMAB 1,0_000000000000000	*SDMAB 1,0_000000000000000*
SDMAB 6,0_00000001_0000000	*SDMAB 6,0_00000001_0000000*
SDMAB 7,1_000000000000000	*SDMAB 7,1_000000111111111*
SDMAB 4,10_00_00_1000000000	*SDMAB 4,10_00_00_1000000000*
SDMAB 10,10_00_00_1000000000	*SDMAB 10,01_00_00_1000000000*
SDMAB 14,00000000_00000000	*SDMAB 5,0_1_000_000_00_0_001_1_1*
SDMAB 15,00000000_00000000	*SDMAB 11,0_1_000_000_00_0_001_1_1*
SDMAB 5,0_1_000_000_00_0_001_1_1	*DMAOK*
SDMAB 11,0_1_000_000_00_0_001_1_1	*MOVRC R2,0*
DMAOK	*LABEL:CONVOLUTION*
MOVRC R2,0	*GDMA R3*
LABEL:Inner_product	*JNER R3,R2,CONVOLUTION*
GDMA R3	
JNER R3,R2,Inner_product	

DCT0	DCT4
SDMAB 0,0_00000001_0000000	*SDMAB 0,0_00000001_0000000*
SDMAB 1,0_000000000000000	*SDMAB 1,0_000000000000000*
SDMAB 6,1_00000000_0000000	*SDMAB 6,1_00001000_0000100*
SDMAB 7,1_000000000000000	*SDMAB 7,1_000000000000100*
SDMAB 4,10_00_00_0000100100	*SDMAB 4,10_00_00_0000100100*
SDMAB 10,10_00_00_0000100100	*SDMAB 10,10_00_00_0000100100*
SDMAB 14,00000000_00000000	*SDMAB 15,01001001_00000000*
SDMAB 5,0_1_000_000_00_0_001_1_1	*SDMAB 5,0_1_000_000_00_0_001_1_1*
SDMAB 11,0_1_000_000_00_0_001_1_1	*SDMAB 11,0_1_000_000_00_0_001_1_1*
DMAOK	*DMAOK*
MOVRC R2,0	*MOVRC R2,0*
LABEL:DCT0	*LABEL:DCT4*
GDMA R3	*GDMA R3*
JNER R3,R2,DCT0	*JNER R3,R2,DCT4*

DWT_A0	DWT_A4
SDMAB 15,00000101_00000000	*SDMAB 1,0_000000000000000*
SDMAB 0,0_00000001_0000000	*SDMAB 7,1_000000010000000*
SDMAB 1,0_000000111111000	*SDMAB 4,10_00_00_0000001001*
SDMAB 6,1_00000001_0000000	*SDMAB 10,10_00_00_0000001001*
SDMAB 7,1_000000010000000	*SDMAB 5,0_1_000_000_00_0_001_1_1*
SDMAB 4,10_00_00_0000001001	*SDMAB 11,0_1_000_000_00_0_001_1_1*
SDMAB 10,10_00_00_0000001001	*DMAOK*

SDMAB 5,0_1_000_000_00_0_001_1_1 *SDMAB 11,0_1_000_000_00_0_001_1_1* *DMAOK* *MOVRC R2,0* *LABEL:DWT_A0* *GDMA R3* *JNER R3,R2,DWT_A0*	*MOVRC R2,0* *LABEL:DWT_A4* *GDMA R3* *JNER R3,R2,DWT_A4*

DWT_D0	DWT_D6
SDMAB 15,00000100_00000000 *SDMAB 1,0_000000111111011* *SDMAB 7,1_000000011000000* *SDMAB 4,10_00_00_0000000111* *SDMAB 10,10_00_00_0000000111* *SDMAB 5,0_1_000_000_00_0_001_1_1* *SDMAB 11,0_1_000_000_00_0_001_1_1* *DMAOK* *MOVRC R2,0* *LABEL:DWT_D0* *GDMA R3* *JNER R3,R2,DWT_D0*	*SDMAB 1,0_000000000000111* *SDMAB 7,1_000000011000000* *SDMAB 4,10_00_00_0000000111* *SDMAB 10,10_00_00_0000000111* *SDMAB 5,0_1_000_000_00_0_001_1_1* *SDMAB 11,0_1_000_000_00_0_001_1_1* *DMAOK* *MOVRC R2,0* *LABEL:DWT_D6* *GDMA R3* *JNER R3,R2,DWT_D6*

B 佈局驗證結果說明

1. DRC

```
--------------------------------------------------------------------------------
--- RULECHECK RESULTS STATISTICS (BY CELL)
---
--------------------------------------------------------------------------------
--- SUMMARY
---
TOTAL CPU Time:                  382
TOTAL REAL Time:                 392
TOTAL Original Layer Geometries: 307776 (2422821)
TOTAL DRC RuleChecks Executed:   386
TOTAL DRC Results Generated:     0 (0)
```

DRC 驗證無誤

2. LVS

```
                OVERALL COMPARISON RESULTS


        #       ####################        _   _
      #         #                  #        *   *
   #    #       #      CORRECT     #          |
    # #         #                  #        \___/
     #          ####################
```

LVS 驗證無誤

C Tapeout Review Form

1. 晶片概述：

   1-1. 專題名稱：具有 Smar -DMA 之 VLIW 架構多媒體訊號處理器

   1-2. Top Cell 名稱：SD297

   1-3. 使用 library 名稱：

   ___ CIC_CBDK35

   ___ CIC_CBDK25

   v CIC_CBDK18

   版本: ***v2***

   1-4. 是否使用 CIC 提供之 Memory？ ***Yes***

   1-5. 工作頻率：***100 MHz***

   1-6. 功率消耗：***400mW***

   1-7. 晶片面積：***2375*** X ***2372***

2. 設計合成：

   2-1. 使用之合成軟體？***Synopsys design compiler***

   2-2. 是否加入 boundary condition：

   v input drive strength、***v*** input delay、***v*** output loading、***v*** output delay

   2-3. 是否加入 timing constraint：

   v specify clock (sequential design)

   ___ max delay、___ min delay (combinational design)

   2-4. 是否加入 area constraint？ ***No***

   2-5. 合成後之 report 是否有 timing violation？***No***

   ___ 有 setup time violation、___ 有 hold time violation

   2-6. 合成後之 verilog 是否含有 assign 描述？ ***No (手動修過)***

   2-7. 合成後之 verilog 是否含有 *cell* 之 instance name？ ***No***

   2-8. 合成後之 verilog 是否含有反斜線 \ 之 instance name 或 net name？ ***Yes***

3. 可測試性設計(前瞻性晶片必填)：

   3-0. 使用之設計軟體？***DFT compiler***

   3-2. 使用之 ATPG 軟體？***Tetramax***

   3-3. 使用 Embedded memory 數量: SRAM***2***，ROM***0***

   Memory 大小:***512x32 (Word x bit)***

   測試方法: BIST***Yes***，or 其他測試方法***N/A***

   若使用 BIST,其 Test Algorithm 爲何?***Moving Inversion (13N March)***

   同時有多個 memory，是否共用 BIST controller ***No***，BIST controller 數量***2***

3-4. Scan Chain Information

Flip-Flop 共有多少個？ ***4970***

Scan chain 的數量共有多少條？ ***1***

Scan chain length (Max.) ？ ***137781.929***

3-5. Uncollapsed fault coverage 是否超過 90% ？ ***Yes***，爲多少？ ***98.45%***

ATPG pattern 的數目爲多少？ ***983***

註：若使用 Synopsys TetraMAX 來產生 ATPG pattern，請使用 `set faults -fault_coverage` 指令指定 TetraMAX 產生 fault coverage information

若使用 SynTest TurboScan 之 asicgen 來產生 ATPG pattern，請以 atpg pessimistic fault coverage 的值爲準

4. 佈局前模擬

4-1. gate level simulation 是否有 timing violation？ ***No***

___ 有 setup time violation、___ 有 hold time violation

5. 實體佈局

5-1. 使用之 P&R 軟體？ ___ Apolo、***v*** SE

5-2. power ring 寬度？ ***8*** 是否已考量 current density(1mA/1um)？ ***Yes***

5-3. 是否考慮 output loading？ ***Yes***

5-4. 是否加上 Clock Tree？ ***Yes***

5-5. 是否加上 Corner pad？ ***Yes***

5-6. 是否加上 IO Filler？ ***Yes***

5-7. 是否加上 Core Filler？ ***Yes***

5-8. 是否上加 Bonding Pad？ ***Yes***

以下(A-1)爲使用 Apollo 者才須回答

A-1. 是否執行 Fill Notch and Gap 步驟？ _____

以下(S-1 至 S-2)爲使用 SE 者才須回答

S-1. power ring 上是否有 overlap vias？ ***No***

S-2. 是否確定 IO Row 和 Corner Row 互相貼齊？ ***Yes***

6. 佈局後模擬

6-1. 是否做過 post-layout gate-level simulation？ ***Yes***

STA(static timing analysis) 軟體？ ***Primetime / Modelsim***

6-2. 是否做過 post-layout transistor-level simulation？ ***No***

6-3. 已針對以下環境狀態模擬：___ SS、___ TT、___ FF

6-4. 晶片取得時將以何種方式進行測試？ ***I MS 測試機台***

6-5. 模擬時是否考量輸出負載影響？ ***Yes***

7. DRC/LVS 驗證

7-1. 是否有 DRC 錯誤？ ***No*** 錯誤原因： _____

_____

驗證 DRC 軟體？ ***Calibre***

是否有不作 DRC 的區域？ ***No***

7-2. 是否有 LVS 錯誤？ ***No***

驗證 LVS 軟體？ ***Calibre***

是否有非 CIC 提供的 BlackBox？ ***No***

設計者簽名: 蘇育緯/周經翔/鍾仁峰　　指導教授簽名: 林進燈

D Module I/O

- Chip I/O

Pin Name	I/O	Width	Function
Global			
CLK	I	1	System Clock
Reset	I	1	Synchronous Reset
Scan Chain			
Scan_in	I	1	Scan Chain Input
Scan_en	I	1	Scan Chain Enable
Scan_out	O	1	Scan Chain Output
Memory BIST			
BistMode	I	1	BIST Mode Select
BistFail	O	1	BIST Failure
Finish	O	1	BIST Finish
ErrMap_A	O	1	Error Map of Ram A
ErrMap_B	O	1	Error Map of Ram B
Program Rom			
pro_mem_address_w	O	16	Program Rom Address
pro_mem_data_w	I	64	Program Rom Data
I2S interface			
BCLK	I	1	I2S Bit Clock
LRCLK	I	1	I2S Word Select Clock
SDIN	I	1	I2S Data Input
BCLK_out	O	1	I2S Bit Clock Output
LRCLK_out	O	1	I2S Word Select Clock Output
SDOUT	O	1	I2S Data Output

- Processor Modules

1. PC Counter / Branch Protect Stage:
2. Instruction Fetch Stage:

I/O Type	Pin Name	Width
Input	CLK	1
Input	Reset	1

Input	pc	16
Input	j_bit	1
Input	jump_done	1
Output	pc_predec	16
Output	pro_mem_address	16

3. Program Memory Stage:

4. Instruction Predecoder Stage:

I/O Type	Pin Name	Width
Input	CLK	1
Input	Reset	1
Input	pc_predec	16
Input	instruction	64
Input	j_bit	1
Input	jump_done	1
Output	op_a	6
Output	op_b	6
Output	pre_instruction_a	32
Output	pre_instruction_b	32
Output	pc_dec	16

5. Instruction Decoder Stage:

I/O Type	Pin Name	Width
Input	CLK	1
Input	Reset	1
Input	instruction	32
Input	op	6
Input	pc_dec	16
Input	j_bit	1
Input	jump_done	1
Output	rd	6
Output	direct	20
Output	op_reg	6
Output	pc_reg	16
Output	flag	2
Output	dec_to_reg	6

Output	address1	6
Output	address2	6
Output	DMA_address	4

6. Register File Stage:

I/O Type	Pin Name	Width
Input	CLK	1
Input	Reset	1
Input	pc_reg	16
Input	op	6
Input	rd	6
Input	flag	2
Input	direct	20
Input	address1	6
Input	address2	6
Input	address3	6
Input	reg_w_data	32
Input	Device_Int	4
Input	int_done	1
Input	dec_to_reg	6
Input	DMA_address	4
Output	pc_alu	16
Output	op_alu	6
Output	rd_alu	6
Output	addr1_alu	6
Output	addr2_alu	6
Output	flag_alu	2
Output	direct_alu	20
Output	reg_r_data1	32
Output	reg_r_data2	32
Output	reg_to_alu	31:0
Output	DMA_address_alu	3:0

7. ALU Stage:

I/O Type	Pin Name	Width

Input	CLK	1
Input	Reset	1
Input	pc	16
Input	op	6
Input	rd	6
Input	flag	2
Input	direct	20
Input	A	32
Input	B	32
Input	jump_done	1
Input	mem_output_data	32
Input	alu_work	1
Input	addr1_alu	6
Input	addr2_alu	6
Input	reg_to_alu	32
Input	Intn	2
Input	RegOut	32
Input	DMA_address	4
Output	Cout	32
Output	C_address	6
Output	j_bit	1
Output	jump_addr	32
Output	mem_w_address	9
Output	mem_input_data	32
Output	wren	1
Output	Pin0	16
Output	Pin1	16
Output	mem_r_address	9
Output	int_done	1
Output	mgrant_a	1
Output	mgrant_b	1
Output	Rcen	1
Output	Rwen	1
Output	Raddr	4
Output	RegIn	32

- Smart DMA Modules

Pin Name	I/O	Width	Function
Global			
CLK	I	1	System Clock
rstn	I	1	Synchronous Reset
Setting DMA			
mgrant0	I	1	Memory A Select
mgrant1	I	1	Memory B Select
rcen	I	1	Register Chip Enable
rwen	I	1	Register Write Enable
raddr	I	4	Register Address
RegIn	I	32	Register Data Input
RegOut	O	32	Register Data Output
Intn	O	2	Channel Controller Interrupt
Memory Interface			
cen0	O	1	Chip Enable of Memory A
oen0	O	1	Output Enable of Memory A
wen0	O	1	Write Enable of Memory A
a0	O	9	Address of Memory A
d0	O	32	Data Input of Memory A
q0	I	32	Data Output of Memory A
cen1	O	1	Chip Enable of Memory B
oen1	O	1	Output Enable of Memory B
wen1	O	1	Write Enable of Memory B
a1	O	9	Address of Memory B
d1	O	32	Data Input of Memory B
q1	I	32	Data Output of Memory B
I2S interface			
BCLK	I	1	I2S Bit Clock
LRCLK	I	1	I2S Word Select Clock
SDIN	I	1	I2S Data Input
BCLK_out	O	1	I2S Bit Clock Output
LRCLK_out	O	1	I2S Word Select Clock Output
SDOUT	O	1	I2S Data Output