第三章 MIPS CPU 規劃設計

第一節 MIPS CPU 相關知識

壹、CPU 基本概念

中央處理器(Central Processing Unit, CPU)是電腦最重要的核心部分。目前最常將 CPU 組成架構分為「資料路徑」(Data Path)及「控制單元」(Control Unit)二大部分。資料路徑包括一個處理邏輯電路及一組執行資料處理的暫存器；控制單元則由決定資料處理次序的邏輯線路所構成[18]。但在資料路徑的結構方面，因涉及多個暫存器與暫存器以及暫存器與記憶體之間的資料轉移概念，在初學者對 CPU 的概念形成上，並不易通盤了解並從事 CPU 設計。因此，在本章研究者以電腦組成的硬體模組架構來做說明與教學。將 CPU 各部分執行的工作分為五大部分，分別是：算術及邏輯運算單元(Arithmetic and Logic Unit, ALU)、暫存器堆(Register File)、記憶體(Memory)及程式計數器(Program Counter PC)，以指令類別為區分，說明各指令類的資料傳遞路徑，經資料路徑整合，最後由控制單元統籌一切程式控制流程。

CPU 為數位計算機的關鍵元件。其功能可對由記憶體所提取的指令加以解碼並執行轉移、算術、邏輯和控制運算，以處理存於內部暫存器、記憶體或 I/O 介面單元的資料。基本上，CPU 提供了一個或一個以上的匯流排和外界相接以便傳遞指令、資料和控制訊息[18]。

MIPS CPU 指令集，於 1980 年代初期被設計出，經過將近 20 年的演進與發展，指令集亦日益龐大，由 MIPS R2000、R3000、R4000，乃至於 64 位元之 MIPS R4600 處理器的開發與生產[44]，可知使用 MIPS 指令集的系統不計其數，因此選用 MIPS CPU 為實作教學，應為適當且適時的。

貳、指令執行時脈週期

依指令執行時脈週期將 MIPS CPU 分類，可將其分為單一時脈週期(Single Clock Cycle)、多重時脈週期(Multiple Clock Cycles)及管線化(Pipelined)。

在單一時脈週期設計下，每個指令執行時脈週期必須有相同長度，因此計算機中最長的指令執行路徑將決定整個時脈週期的長度，通常為記憶體載入指令，因為它必須使用到所有的功能單元，因此在其他指令都必須與記憶體載入指令一樣，具有相同的時脈週期長度的設計下，所以 CPU 整體執行效能無法較佳化。

而多重時脈週期是解決單一時脈週期缺點的辦法之一。多重時脈週期允許每個指令具有個別的指令執行週期,因此 CPU 整體架構不須拘泥於固定指令執行週期，而執行週期較短的指令所耗費的 CLOCK 數減少，效能因而提高。在硬體需求方面，多重時脈週期允許每個功能單元在一個指令中能多次使用，因功能單元分享所以在硬體的需求量相對減低。因此其最主要的優點為：

一、不同的指令有不同的時脈週期，執行效能較佳。

二、在指令執行過程中功能單元能分享，硬體需求較低。

管線化允許重疊指令的執行以提高執行效能。一般來說 MIPS 的指令分成五個管線階段：指令提取、指令解碼、執行運算或計算位址、存取運算元、結果回寫。當第一道指令至指令解碼時，第二道指令之指令提取亦同時進行；第一道指令執行運算時，第二道指令進行指令解碼，同時進行第三道指令的提取，依此類推，即為管線化的設計。將每道指令執行順序進行適當分割，即能充分運用每個功能單元，提高執行效能，並有效減低硬體的需求量[41]。

參、Multiple Cycles MIPS CPU 相關知識

在第二章總結討論中提及，依實際教學所需，在評估需求之下，決本研究決定採用 Multiple Clock Cycles 之 32 Bits MIPS CPU 設計，原因是在大專院校層級進行這類教學，在設計難度上對學生而言是適當的，另一方面，在參考各方文獻後發現，尚未發現有 Multiple Clock Cycles MIPS CPU 的硬體設計文獻。

Multiple Clock Cycles MIPS CPU 基本架構圖如圖 3-1 所示[41]，相關知識依記憶體、暫存器堆及指令執行步驟說明如下：

一、每一個記憶體位址的內存資料量大小為 1Byte，完整的 32 Bits 指令分別被儲存在 4 個記憶體位址中，故而 32 Bits 之 MIPS CPU 之程式計數器每次遞增 4。MIPS CPU 未將程式計數器獨立成一個模組，而是充分利用 ALU 模組。在每次指令存取週期(Instruction Fetch)之間，ALU 模組即執行程式計數器功能，進行記憶體位址計算指向下一道指令的位址，在此部分的執行係經由控制單元中的 ALU_SrcB、PC_Write 控制訊號所控制。

二、暫存器堆：基本 MIPS CPU 共有 32 個一般暫存器(General Purpose Register)、32 個浮點運算暫存器(Floating Point Register)、32 個特殊暫存器(Special Purpose Register)，每種暫存器資料長度皆為 32 Bits。

1. 一般暫存器：不限於專門用途，適用於所有運算。

2. 浮點運算暫存器：專用於支援浮點運算指令之資料，能用於單精度浮點數(Single-Precision Floating-Point Number)運算，而倍精度浮點數運算(Double-Precision Floating-Point Number)則可用 2 個浮點運算暫存器搭配使用。

3. 特殊暫存器：在實際的 CPU 內，特殊暫存器用於儲存狀態字組(Status Word)，或是處理器發生例外處理時，做為例外狀況處理暫存器之用。

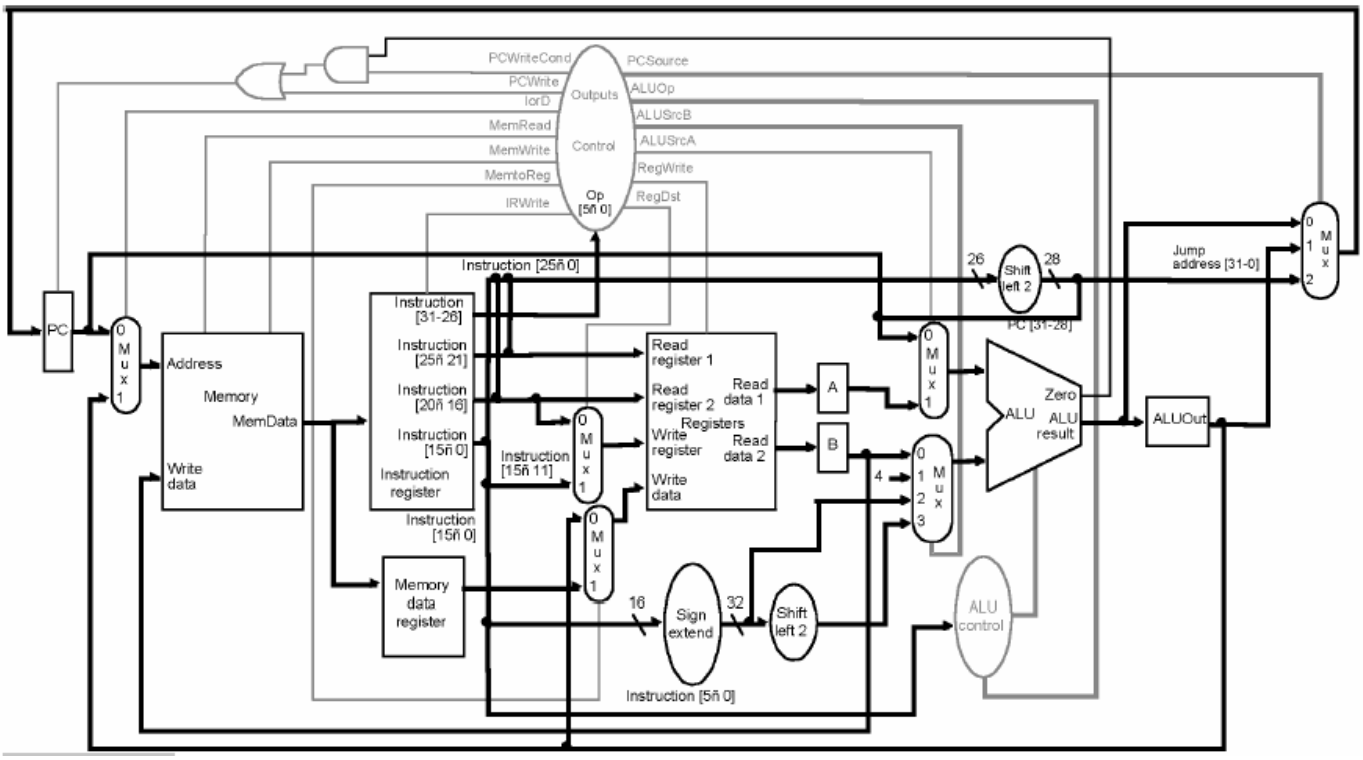

圖 3-1 32 Bits Multiple Clock Cycles MIPS CPU 基本架構圖

三、指令執行步驟：分別為指令提取、指令解碼 (Instruction Decode)、指令執行(Instruction Execution)、記憶體存取(Memory Access)及寫回(Write Back)五步驟[41]。各步驟工作項目如下：

4. 指令提取：將記憶體內存之指令，由記憶體搬移至指令暫存器(Instruction Register, IR)暫存，IR = Memory [PC]。在指令提取階段，ALU 同時計算出下一道指令的位址，PC = PC + 4。

5. 指令解碼：進行指令解碼工作，並將前一步驟所提取出的指令指定的暫存器內容提供給 ALU，再進行指定之運算工作。若為程式分支(Branch)類指令，則在此時同時計算分支指令的目的位址。

6. 指令執行：依據步驟 2 的結果開始執行指令。依不同指令類別的不同，進行下列的運算工作。MIPS CPU 指令集格式及指令類別，請參考附錄 A 之相關說明。
   a. 記憶體位址的計算。
   b. 執行算術邏輯運算指令：若為暫存器類(Register Type, R-Type)指令，則進行 ALUOut = A function B 的運算。

c. 分支指令：若 A = B，則 PC = ALUOut，完成分支指令的執行。

d. 跳躍指令(Jump, J-Type)：PC = PC[31-28] || (IR[25-0]<<2)。

7. 記憶體存取：對記憶體進行存取，若為 R-Type 的算術邏輯運算指令則在此時完成。

8. 回寫：將資料由記憶體回寫至指定暫存器，完成所有指令執行。

執行步驟摘要如表 3-1 所示。各控制訊號設定及說明如表 3-2 所示。

表 3-1 MIPS CPU 指令執行步驟表

指令類別 / 步驟	R-Type 指令	記憶體相關指令	Branch 指令	Jump 指令
指令提取	IR = Memory[PC] PC = PC +4			
指令解碼	A = Reg[IR[25-21]] B = Reg[IR[20-16]] ALUOut = PC + (Sign-extend (IR[15-0])<<2)			
指令執行	ALUOut = A op B	ALUOut = A + sign-extend(IR[15-0])	If (A == B) then PC = ALUOut	PC = PC[31-28] || (IR[25-0]<<2)
記憶體存取	Reg[IR[15-11]] = ALUOut	Load: MDR = Memory[ALUOut] Or Store: Memory[ALUOut] = B		
回寫		Load: Reg[IR[20-16]] = MDR		

表 3-2 MIPS CPU 指令執行控制訊號設定表

Signal Name	Value	Effect
MemRead	0	None
	1	Contents of memory at the read address are output on read data out
MemWrite	0	None
	1	Memory contents at write address is replaced by value on data output
ALUSrcA	0	The first ALU operand is the PC
	1	The first ALU operand comes from the register given by the rs field
ALUSrcB	00	The second ALU operand comes from the register given by the rt field
	01	The second ALU operand is the constant 4
	10	The second ALU operand is the sign-extended bits [15-0] of IR
	11	The second ALU operand is the sign-extended & shifted bits [15-0] of IR
ALUOp	00	Addition operation
	01	Subtraction operation
	10	Function depends on the contents of the code field in the IR
	11	Undefined
RegDst	0	The destination register comes from the rt field of the IR
	1	The destination register comes from the rd field of the IR
RegWrite	0	None
	1	Enables writing the register given by the write register number
MmemtoReg	0	The value for the written register comes form the ALU
	1	The value for the written register comes form data memory
IorD	0	The memory address comes from the PC
	1	The memory address comes from the ALU
IRWrite	0	None
	1	The value from memory is written into IR

PCWrite	0	
	1	PC write, source is defined by PCSource
PCWriteCond	0	
	1	PC write IF ALU indicates a ZERO condition
PCSource	00	PC = ALU output
	01	PC = target register
	10	PC = jump target address (PC + 4 [29-26] & IR[25-0]<<2)
	11	undefined

第二節 MIPS CPU 模組規劃

一般學生在設計數位系統電路時，常因為系統的龐大而不知由何下手設計，因而顯得不知所措。以 VHDL 硬體描述語言做數位電路系統的設計，採用「由上而下的模組化設計(Top Down Modeling)」，把數位電路系統不斷向下拆分，強調一個完整的數位系統架構是由數個功能不同的「模組」所組成，在此，「模組」是指具有某功能的數位電路單元。設計者分別將數個模組設計完成並測試功能後，再將每個模組慢慢連接、組合，以形成所需的數位電路系統。在學習者的學習設計概念上，有助於清晰系統設計的架構，並簡化設計步驟及設計難度。使用模組化設計具有下列優點：

一、模組化的教學利於實施課程規劃。

二、學生易瞭解學習的目標及學習方向。

三、可依學生程度的不同來彈性增減課程單元。

因此在本章中，將本研究所設計之 CPU 及 FPGA 實驗平台模組化，希冀學習者能整合以往所接受之專業知能，按程序將 MIPS CPU 設計完成並模擬功能，最後再將程式下載到自製的 FPGA 實驗平台中，完成驗證工作，達到計算機結構中 CPU 設計教學的研究目的。

整體研究軟硬體設計架構分為三大部分，分別為 MIPS CPU 設計單元及 I/O 介面控制模組設計單元、FPGA 實驗平台，並將本研究之設計原理動作做一說明。

一、MIPS CPU 設計單元：以 Xilinx ISE v6.2 為程式開發環境，利用 VHDL 硬體描述語言為工具，設計 Multiple Clock Cycles 之 MIPS CPU。

二、I/O 介面控制模組設計單元：開發 I/O 模組之控制電路，未來 CPU 若

需外部 I/O 做顯示或輸入需求，即可配合 I/O 介面控制模組單元執行。

三、FPGA 實驗平台：外部 I/O 電路之實際硬體電路。其功能為進行 CPU 或各種 I/O 實作需求。

四、設計原理動作說明：整合「MIPS CPU 設計單元」與「I/O 介面控制設計單元」，將所設計之程式下載於同一顆 FPGA 晶片中。CPU 能接受內部的指令並做處理，功能測試先以軟體模擬方式測試，軟體模擬成功後下載至實驗平台進行實體驗證。MIPS CPU 整合所設計的 I/O 介面控制模組，利用輸出裝置(文字型 LCD、七段顯示器、LED)，顯示內部指令運算執行結果，與軟體模擬結果共同進行分析、比對測試結果。I/O 介面控制模組用於產生 I/O 裝置控制訊號，因 MIPS CPU 模組設計中未包含 I/O 控制訊號產生模組，故而將 MIPS CPU 與 I/O 介面控制模組各別設計，達成 MIPS CPU 與嵌入 I/O 介面控制設計之研究目的。MIPS CPU 可再配合輸入裝置(指撥開關、按壓開關、4x4 鍵盤)相關輸入控制模組，更可達成使 MIPS CPU 接受由外部輸入指令執行功能。在本研究中的 I/O 介面控制模組單元中之每一模組，皆控制每一組不同的外部顯示電路或輸入控制電路。

壹、實作指令集

在本研究中實作的記憶體相關(memory-reference)指令、算術邏輯運算(arithmetic-logic)指令、分支指令，已包涵 MIPS 指令集中的主要指令。在實作指令集編碼(Encoding)方面，遵循 MIPS CPU 指令集編碼並未做任何修改，如附錄 A 表 A-4 所示，因此能與其他 MIPS CPU 相容。實作指令表如表 3-3 所示。

表 3-3 實作指令表

Mnemonic	OP Code	Description	Operation	Inst. Type
LW	100011	Load word	$t = MEM[$s + offset]	M
SW	101011	Store word	MEM[$s + offset] = $t	M
ADD	000000	Add	$d = $s + $t	R
ADDU	000000	Add unsigned	$d = $s + $t	R
SUB	000000	Subtract	$d = $s - $t	R
SUBU	000000	Subtract unsigned	$d = $s - $t	R
AND	000000	Bitwise and	$d = $s & $t	R

OR	000000	Bitwise or	$d = $s | $t	R
XOR	000000	Bitwise exclusive or	$d = $s ^ $t	R
NOR	000000	Bitwise nor	$d = ~($s | $t)	R
SLL	000000	Shift left logical	$d = $t << shamt	R
SRL	000000	Shift right logical	$d = $t >> shamt	R
SRA	000000	Shift right arithmetic	$d = $t << shamt	R
ADDI	001000	Add unsigned	$t = $s + immed	I
ADDIU	001001	Add immediate unsigned	$t = $s + immed	I
ANDI	001100	Bitwise and immediate	$t = $s & immed	I
ORI	001101	Bitwise or immediate	$t = $s | immed	I
XORI	001110	Bitwise exclusive or immediate	$d = $s + immed	I
BEQ	000100	Branch on equal	If $s = $t advance_pc(offset); Else advance_pc	B
BGEZ	000001	Branch on greater than or equal to zero	If $s >= 0 advance_pc(offset); Else advance_pc	B
BGTZ	000111	Branch on greater than zero	If $s > 0 advance_pc(offset); Else advance_pc	B
BLEZ	000110	Branch on less than or equal to zero	If $s <= 0 advance_pc(offset); Else advance_pc	B
BLTZ	000001	Branch on less than zero	If $s < 0 advance_pc(offset); Else advance_pc	B
BNE	000101	Branch on not equal	If $s != $t advance_pc(offset); Else advance_pc	B
J	000010	Jump	PC = nPC ; nPC = (PC & 0xf0000000) | target	J

貳、模組規劃並定義功能需求

為遵循模組化設計、由上而下設計的原則，將 MIPS CPU 每一單元功能向下拆分為各個小模組，整體 MIPS CPU 設計與教學模組架構規劃為 4 個層級(Layer)，如圖 3-2 所示。Layer1 為完整 32 Bits 之 MIPS CPU 之 VHDL Code。Layer2 則由 Mano[13]對於 CPU 架構規劃的概念，將 MIPS CPU 區分為資料路徑單元與控制單元二個模組，其下再定義資料路徑單元所需之各分支模組為 Layer3，其中 Layer3 包含了基本算術邏輯運算單元、ALU 控制單元(ALU Control)、指令暫存器單元(Instruction Register)、記憶體單元(Memory)、及暫存器堆(Register File)。

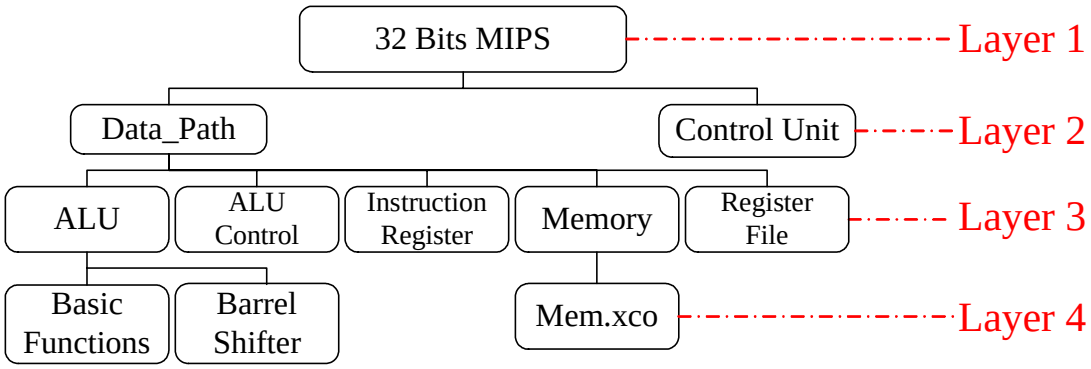

圖 3-2 MIPS CPU 模組架構圖

各模組詳細說明如下：

一、ALU：依指令集規劃下，設計具備 MIPS CPU 基本之算術邏輯運算功能，此單元除了做暫存器及立即值之算術邏輯運算外，亦須扮演程式計數器的角色，負責計算下一道指令的記憶體位址或程式分支位址。ALU 模組其下又分為 Basic Function 模組及 Barrel Shifter 模組。Basic Function 模組負責執行指令集中的 ADD、SUB、AND、OR、XOR、NOR 等相關運算；Barrel Shifter 模組負責執行指令集中的移位指令運算，如 SLL、SRL、SRA 指令。

二、ALU Control：依每道指令之不同，分別產生控制 ALU 執行算術邏輯運算之訊號產生單元。

三、Instruction Register：暫存由記憶體送出之指令，並依下級運算所需將指令依指令格式分別送至下級模組。

四、Memory：記憶體為指令及資料儲存所在，在本研究中設計為一個 256 x 32 Bits 大小之 Block RAM。Block RAM 係利用 Xilinx ISE v6.2 中之 Core Generator 產生，故有一 Mem.xco 檔。

五、Register File：MIPS CPU 主要之暫存器堆，包含有 32 個一般用途暫存器(General Purpose Registers)。

本研究的模組規劃其優點為：

一、未來若要實現乘、除法指令，可透過整合 Basic Function 模組與 Barrel Shifter 模組達成乘、除法運算指令，因此在本設計所改良的 MIPS CPU 架構圖是易於擴充設計的，並在教學應用上，亦可將這類指令擴充設

計成進階實作單元。

二、Memory 模組中，使用 FPGA 內部已規劃之 Block RAM，其優點為減少 FPGA 之 Gate Count 使用量，在節約教學資源的觀點上，可選用 Gate Count 數較小的 FPGA，同時亦可達到教學與實作的需求。或者更可設計使用外部記憶體 IC，透過 MIPS CPU 與外部記憶體資料交換的過程，達到使用外部記憶體元件使用及控制之教學目的。

將 MIPS CPU 設計各模組功能依指令集規劃完畢後，則進入實作設計及教材與實作撰寫程序，相關之部分教材章節規劃參考附錄 B。