植基於 SOPC 之浮點加速器

國立虎尾科技大學資訊工程系
徐元寶 hsuyp@nfu.edu.tw
蔣惟丞 enjoysea0605@yahoo.com.tw
莊政憲 sa020816x@yahoo.com.tw
張建華 kiwirider555@yahoo.com.tw

摘要

本文的目的主要在為一SOPC系統增加浮點運算功能，首先利用Verilog硬體描述語言設計一浮點運算電路(Float Point Unit，簡稱FPU)，並且透過客製化指令技術(Custom Instruction，簡稱CI)將FPU與Nios II嵌入式處理器做整合後，使用Qurtus II電路開發軟體將整合的電路做編譯，產生硬體組成檔，再將硬體組成檔掛載到Nios II SOPC發展平台上做驗證。此外我們也透過μC/OS-II嵌入式作業系統來執行硬體資源的控制，完成的系統具有浮點數運算的功能，也由於作業系統的控管，使得Nios II SOPC發展平台能達成多工之功能。

關鍵詞：Nios II嵌入式處理器、μC/OS-II嵌入式作業系統、FPU

Abstract

Enhancing a SOPC system by equipping it with floating point operation functions is the purpose of this article. Through the Verilog hardware description language we design a floating point unit (FPU), which is then integrated into the Nios II embedded processor by the technology of Custom Instruction(CI). The Quartus II circuit development software compiles the integrated circuit to produce a hardware configuration file for the purpose of being finally downloaded onto the Nios II SOPC development platform for verification. Besides, the μC/OS-II operating system is ported on the SOPC system to control and monitor the system resources. Such that can the entire system, under the control of operating system, not only execute floating point operations but also the multitasking functions.

1、前言

由於積體電路設計與製造技術提升，在每塊晶片中，已經能夠包含上億個電晶體，達成在晶片中整合許多由IC組成的電子系統技術已經被達成，發展成所謂的系統晶片(System On Chip，簡稱為SOC)，系統晶片不再是只有單一功能的電路，而是將訊號處理、擷取與輸入輸出等完整的系統功能整合在一起，成為有專門功能的電子系統晶片。

但SOC由於投片製造ASIC的成本商，對於數量小或處於發展階段的SOC若馬上投片生產，需要投入較多的資金，承擔較大的試製風險。為解決這個困境而有SOPC的出現。SOPC將系統處理機制、各層次電路及元件的設計整合在一FPGA晶片上，其最大的有優點為不必投片製造ASIC即可完成試製產品，上市時間快。另一方面，由於積體電路與電子設計自動化工具(Electron Design Automation，簡稱為EDA)的快速發展，使得電子系統的設計者，可以利用現有的技術和EDA工具的幫助完成系統級的設計並在FPGA系統晶片上實現。

SOC 和 SOPC 的設計都是以矽智財(Intellectual Property，簡稱為 IP)為基礎，IP 是以硬體描述語言為主要的硬體功能性設計，並且加上以電腦為平台的 EDA 工具來達成電子系統的設計者所要的功能[2][3][4]。

2、研究方法

由於 Nios II 處理器無法處理浮點數運算，為增加此功能，本專題的研究流程，如圖 1 所示，步驟介紹如下：

- 第一步：研究方向分成，硬體平台架設與浮點運算電路設計。
- 第二步：硬體平台架設與浮點運算電路設計完成後，將硬體平台與浮點運算電路做整合。
- 第三步：整合時，必須利用客制化指令的技術，產生軟體發展時所需的客制化 C 函式。
- 第四步：由於產生之浮點運算電路計算出的結果為 32 位元 IEEE754 格式，不易驗證，所以利用程式設計將輸入的數值與輸出的數值轉成十進制數值。
- 第五步：增加網路存取之功能，透過固定 IP 讀取到 Nios II SOPC 平台，並控制 Nios II SOPC 平台上的 LED 與網頁。
- 第六步：由於系統平台只能提供一位使用者執行運算或控制平台，則透過 μC/OS-II 嵌入式作業系統控管硬體資源，達成多工。
- 第七步：系統平台可提供兩部電腦同時執行浮點數運算或讀取網頁。

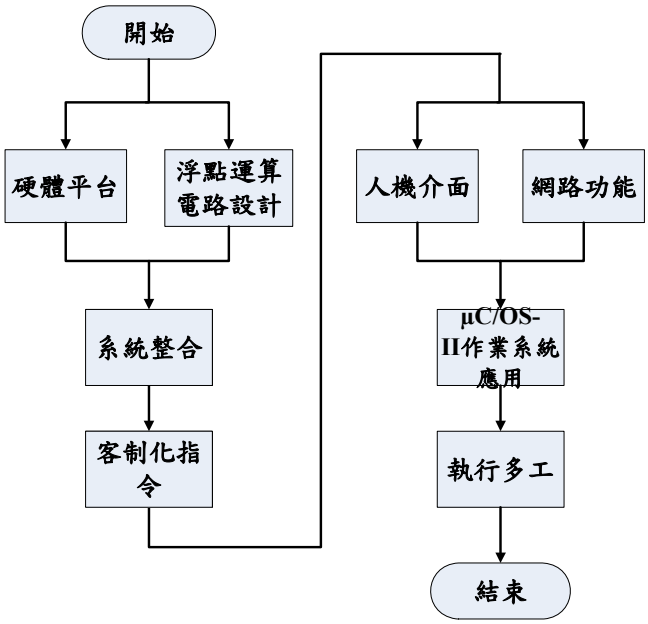

圖 1 專題研究流程圖

2.1 Verilog 浮點運算電路設計

IEEE754 是由 IEEE(Institute of Electrical and Electronics Engineers，電子電機工程師協會)訂定，為表示浮點數格式的一種標準，目前大部份電腦都是採用此標準來表示浮點數，此標準定義兩種格式如下：

- 單精確度(Single precision)：32bits 長度的 2 進制格式，分別由 1bit 符號位元、8bits 指數位元、23bits 係數位元組成，如圖 2 所示。
- 雙精確度(Double precision)：為 64bits 長度的 2 進制格式，分別由 1bit 符號位元、11bits 指數位元、52bits 係數位元組成，如圖 3 所示。

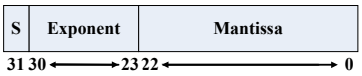

圖 2 單精確度(32bits)浮點數表示格式

圖 3 雙精確度(64bits)浮點數表示格式

IEEE754 為了把最小的指數部份表示成 $000\ldots000_2$、最大的指數部份表示成 $111\ldots111_2$，在 single precision 中，指數部份為 8bits，範圍值為 0~255，由於實際的指數有正負，正負各佔一半，所以其範圍為-126~127，並且採用「偏移表示法(Bias expression)」，其中 $0(00000000_2)$ 與 $255(11111111_2)$分別表示 0 與無限大，真正能表示的範圍在 1~254 間，實際指數(E^{true})=無號指數(E)－偏移值(bias)，single precision 偏移值為 127、double precision 偏移值為 1023，其浮點數表示公式如下：

$$(-1)^s \times (1+.M) \times B^{E-b} \qquad (1)$$

S： 正負符號為 1bit
M： 假數(係數)
B： 基底為 2
E： 指數欄位之指數
b： 偏移值，single precision 為 127、double precision 為 1023

在公式 2-1，(1+.M)的 1 為隱藏位元(Hidden bit)，填入 IEEE754 浮點數表示格式並非包含此位元，因此可在固定位元下，提升 1 位元的精確度，所有數在填入 IEEE754 浮點數表示格式前，必須先正規化成(1)之形式，再填入表格中，若無法在有效範圍內表示，即為非正規化。 以上介紹，了解 IEEE 754 格式後，依此觀念使用 Verilog 硬體描述語言，設計浮點加減乘除運算電路單元，架構如圖 4 所示。

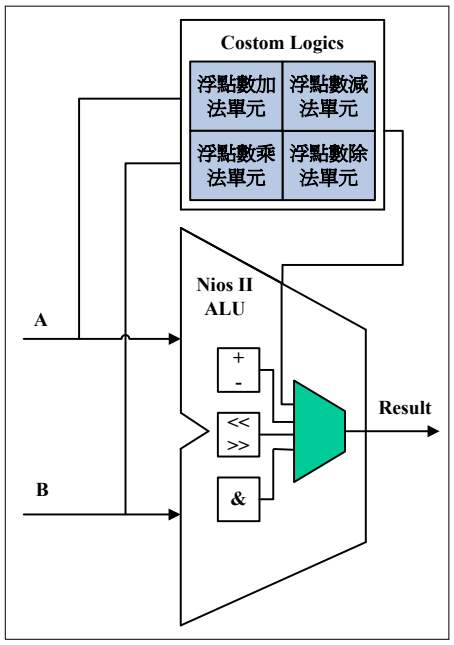

圖 4 IEEE 754 浮點運算電路架構圖
(參考圖片來源：[5])

完成浮點運算電路後，利用 ModelSim 電路測試軟體，測試 FPU 的正確性，並模擬結果，本文只列出加法模擬，如圖 5，opa、opb 為兩輸入值，result 為浮點運算單元計算的值，expect_result 為期望值，ErrorCount 用來記錄錯誤個數，若 result 與 expect_result 比對不相等，則表示運算結果有錯誤 [6]。

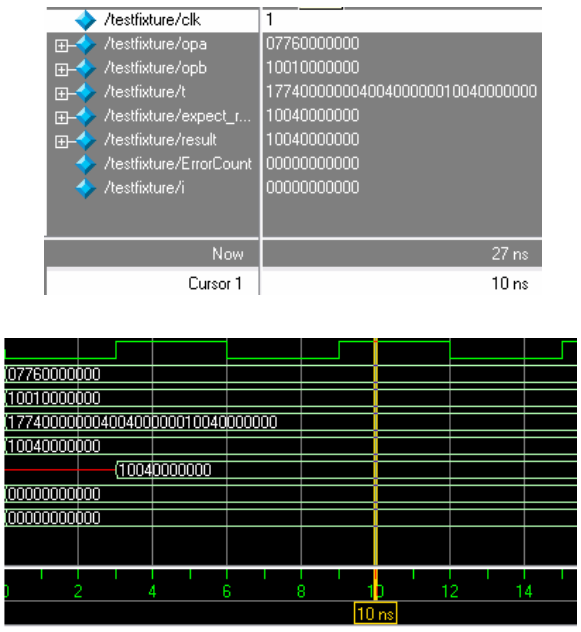

圖 5 浮點數加法模擬結果

2.2 硬體平台

SOPC設計流程如圖6所示，第一步驟是定義系統，包括定義處理器、記憶體介面、週邊元件以及客戶指令(Custom Instructions)等方面。定義完系統後，SOPC Builder完成Generate階段，接著產生使用者設計之系統平台，系統平台產生後，分成兩個方向進行設計，分別為硬體設計與軟體設計。在硬體設計方面，使用QuartusII工具對硬體描述語言原始程式及EDIF檔進行邏輯電路合成之編輯。在軟體設計方面，使用Nios II IDE軟體發展工具與軟體資源(例如：Header檔、程式庫與週邊硬體驅動程式等)，設計與編輯應用程式碼。最後，將硬體設計與軟體設計依序下載至發展電路板驗證。

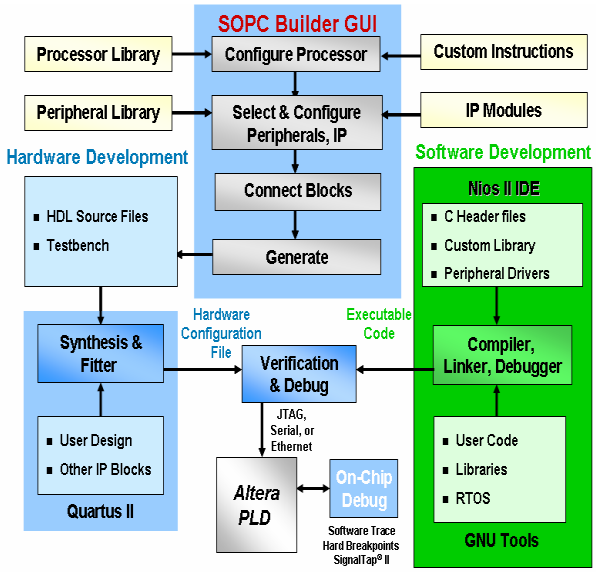

圖6 SOPC發展流程圖
(圖片來源： [5])

3、客制化指令(Custom Instruction，簡稱 CI)

客制化指令是 Nios II 一項特殊技術，它將使用者所輸入的資料，當成 Nios II 處理器的算術運算單元(ALU)輸入資料，這部份必須在 SOPC Builder 中，把使用者設計之硬體電路與 Nios II 處理器做整合，使 ALU 增加了使用者所需要的運算功能[5]。

4、軟體發展過程

使用SOPC Builder架設完系統平台後，即可進行自訂週邊硬體設備的程式設計，以達到所需要的功能，此部份需使用Nios II IDE(Integration Development Environment)，Nios II IDE是Altera公司所開發的工具。搭配Quratus II設計SOPC系統而使用的軟體發展工具。其支援軟體發展的程式語言有，C、C++與組合語言[5]。

Nios II IDE軟體發展工具包含了：

- GNU Tool Chain：以標準GNU GCC compiler、assembler和linker工具為基礎。
- Instruction Set Simulator：當使用者沒有發展板時，可提供執行程式的功能與除錯的功能。
- Hardware Abstraction Layer System Library(HAL)：以ANSI C為基礎，提供硬體的driver與提供使用者呼叫硬體函式驅動硬體，系統發展者需使用嵌入式作業系統時，須將嵌入式作業系統的Source code先引進HAL中。
- 嵌入式作業系統：Nios II SOPC平台提供移植嵌入式作業系統，當系統發展者需要時，可從HAL函式庫中呼叫。

5、實作過程

5.1 系統架構

SOPC builder提供許多IP，但並非每一個IP都需使用，本專題使用到的硬體如下：

- Nios II處理器並增加客制化的電路到ALU
- Avalon Bus
- Uart
- SRAM Memory
- Flash Memory
- PIO(Reset)
- Timer(system timer)
- lan91c111(網路)
- jtag_Uart(Download Cable)

當IP的選擇完成後，使用SOPC Builder執行整合，成為所需之系統平台，整合過程中，若無錯誤產生，必須回到Quartus II設定FPGA晶片對應之腳位，並且Compile硬體是否正確，若沒有錯誤產生，則產生硬體元件如圖7所示，將硬體組成檔燒錄到Nios II SOPC發展板上，接著即朝向軟體開發流程。

圖7 硬體元件圖

5.2 Nios II客制化C函式使用

客制化電路設計測試完成後，Nios II IDE 會產生該客制化電路對應之軟體巨集函式庫，使用者可以 C 函式呼叫的方式與客制化電路溝通，其函式名稱為：ALT_CI_NAME(A,B)，其中 NAME 為硬體模組的名稱，A 與 B 為使用者需要輸入給硬體電路的兩個運算元，輸入與輸出的長度皆限制為在 32 位元。

5.3 µC/OS-II嵌入式作業系統

µC/OS-II是一套具有可攜性、可裁剪、可搶先且即時多工的核心。µC/OS-II是由ANSI C語言所撰寫而成，包含一部分組合語言，使其能適應於不同架構的處理器核心，µC/OS-II由µC/OS升級而來，並且做相當多之改進，µC/OS中所有的功能µC/OS-II都有包含，為一套高度被使用的嵌入式作業系統[7]。

本專題也使用µC/OS-II來對軟體的執行與硬體資源執行控管，首先，將客制化C函式封裝到µC/OS-II中的OSTaskCreate()或OSTaskCreateExt()函式，以上兩個函式為µC/OS-II建立process所用，µC/OS-II執行前至少需要建立一個process，而本文建立了兩個process。

OSTaskCreateExt()函式可提供使用者建立process，設定process的優先權、是否需使用系統堆疊或所封裝的工作是否需傳遞參數等，最後使用OSStart()函式開始執行多工的系統運作。

5.4 人機介面

由於本文所設計之浮點運算電路，只能接收和輸出符合 IEEE754 格式的值，若要讓使用者直接輸入或解讀 IEEE754 格式的值，很不方便。在驗證方面，也有所困難，所以撰寫轉換程式，使系統能夠較人性化，本文將數值轉換的程式加到客制化C函式的前後端，架構如圖 8 所示，五邊形表程式轉換的部份，方形表人性化的輸入與輸出。

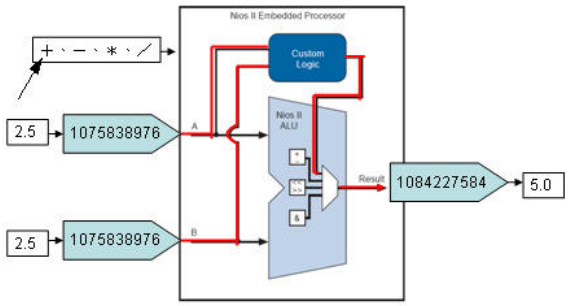

圖 8 程式整體架構圖
(圖片來源：[5])

軟體程式設計部分，如圖8所示，分成三部份，前端轉換，硬體計算，後端轉換，前端負責將使用者輸入的一般十進制數值轉成客制化C函式所需要的十進制值（兩者不同）；主程式是根據使用者的需求去判斷說執行那種運算？再呼叫相對應的客制化C函式，以及判斷是否結束執行；後端的部份負責將客制化C函式輸出結果轉成一般人易懂的十進制數值，圖9為前端、主程式與後端執行流程。

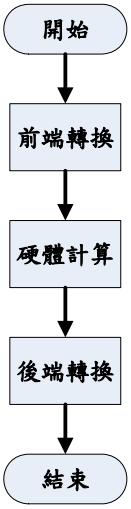

圖 9 程式執行流程

5.5 新增網路功能

將Nios II IDE內建範例做修改，將原本只能透過區域網路對Nios II SOPC發展板做讀取的範例，經過修改能透過網際網路對Nios II SOPC發展板做讀取，並透過µC/OS-II即時作業系統將網路的功能與浮點運算電路同時執行，達成多工[1]。

5.6 實際平台

由於 µC/OS-II 嵌入式作業系統的使用，可提供兩部電腦同時存取 Nios II SOPC 平台，也由於增加網路功能，使得完成的平台有三個種類，說明如下：

- 第一類：如圖 10 所示，兩部電腦透過不同介面連接到 Nios II SOPC 發展板，並透過多工方式，使兩部電腦同時執行浮點數運算。

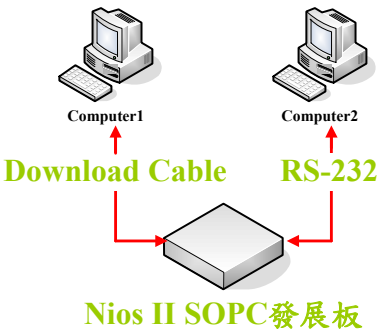

圖 10 第一類實際系統平台

➢ 第二類：如圖 11 所示，一部電腦透過網路，控制發展板上的 LED 七段顯示器，另一部透過 RS-232 執行浮點數運算。

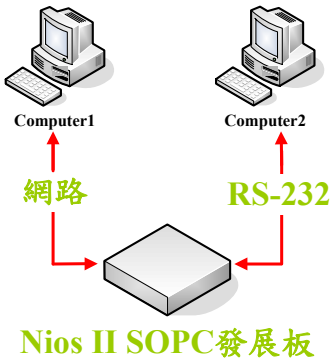

圖 11 第二類實際系統平台

➢ 第三類：如圖 12 所示，由於有指定固定 IP 給 Nios II SOPC 發展板，所以可以透過網際網路，存取發展板上的網頁。

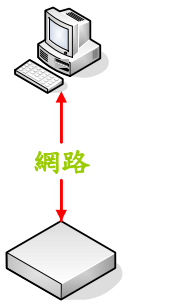

圖 12 第三類實際系統平台

7、結果討論

本文將兩個 I/O 驅動程式不同的浮點數運算控制程式分別封裝到 µC/OS-II 嵌入式作業系統 OSTaskCreateExt()的函式中，形成兩個 process，接著在主程式中呼叫 µC/OS-II 的 OSStart()函式開始執行，執行時，由於將兩台電腦的優先權不同，所以將先執行圖 10 中，Computer1 的工作，當 Computer1 執行工作產生閒置時，µC/OS-II 作業系統會將 Nios II 處理器控制權切給 Computer2，並執行 Computer2 的工作。

執行一開始，先讓使用者選擇要執行那種運算，有加、減、乘與除，使用者選擇一項運算後，即可分別輸入兩個數值並計算出結果，由於客制化 C 函式無法接受浮點數值，所以必須轉換，再輸入給客制化 C 函式，在此也將客制化 C 函式輸入與計算後的輸出值一並列出，以便做對照，如測試圖 13。

圖 13 測試圖

8、討論

經過這次實驗，發現SOPC技術發展嵌入式系統的功能很齊全，也很具彈性，以可程式化的FPGA晶片來實做，降低成本與開發時間，由於SOPC技術漸漸成為IC設計的主流，利用SOPC技術與硬體描述語言兩者整合，能夠使得系統晶片成為有特殊功能的硬體系統晶片。

此次實驗中，了解到Nios II處理器如何與使用者邏輯溝通加上作業系統如何應用，Nios II處理器與使用者邏輯溝通方法有兩種，第一種是自訂使用者邏輯，第二種為Custom Instruction技術，由於一開始採用自訂使用者邏輯以致於必須要解決浮點運算電路與Avalon Bus溝通的問題，必須再使用Verilog撰寫一個介面電路，但是由於介面電路的製作過於困難，使得專題進行受到阻礙，即採用另一種溝通技術，即Custom Instruction，即達成互相溝通之結果，Nios II處理器與浮點運算電路能整合在一起時，即開始發展軟體流程的部分，在軟體發展部份由於必須有作業系統來執行硬體資源做控管，最後慢慢的從單工，而達成多工。

參考文獻

[1] Altera晶片公司，(http://www.altera.com/)。
[2] 廖裕評,"SOPC 設計實務", 全華出版。
[3] 廖裕評、陸瑞強,"系統晶片設計－使用 Quartus II",全華出版。
[4] 李宜達,"以 NIOS 為基礎的 SOPC 設計與實作",全華出版。
[5] 茂綸股份有限公司,"Training data" 。
[6] 黃英叡、黃稚存、張銓淵、江文啟,"Verilog 硬體描述語言",全華出版。
[7] JEAN J. LABROSSE 原著、黃文增編譯, "MicroC/OS-II 即時作業系統", 全華出版。