行政院國家科學委員會補助專題研究計畫成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※
※　　　　　　　　　　　　　　　　　　　　　　　※
※　　　**通訊與訊號處理晶片用模組庫之發展(II)**　　　※
※　　　　　　　　　　　　　　　　　　　　　　　※
※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：■個別型計畫　　□整合型計畫

計畫編號：NSC 89－2215－E－002－029

執行期間：88 年 8 月 1 日　至　89 年 7 月 31 日

計畫主持人：闕志達

執行單位：國立臺灣大學電機資訊學院電機系

中　華　民　國　八　十　九　年　八　月　十　日

行政院國家科學委員會專題研究計畫成果報告

通訊與訊號處理晶片用模組庫之發展(II)

Development of Module Library for Communication/Signal Processing System Chips

計畫編號: NSC 89－2215－E－002－029
執行期限: 88年8月1日至89年7月31日
主持人: 闕志達　　**計畫參與人員**: 劉明倫
國立臺灣大學電機資訊學院電機系
電子信箱: chiueh@cc.ee.ntu.edu.tw

一、中文摘要

本計劃目的在建立一適用於基頻訊號處理單元及資料路徑模組設計之參數化模組設計架構。該架構由 C++ **硬體類別、**C++**參數化模組庫與晶胞庫構成。此架構下設計者指定系統參數後，利用參數化模組可自動產生** C++ **功能階層模擬程式碼與供晶片實體設計閘之階層** Verilog **硬體描述。藉此設計流程我們可達到高設計抽象度及重複使用率，較短之設計時間並避免設計人員分別撰寫兩系統描述時可能發生的錯誤。本計劃第一年中我們工作在利用** TSMC 0.6／0.35μm **兩製程設計與測試晶胞庫，並提出參數化模組之** C++**類別之雛型，利用晶胞庫試作底層之參數畫模組。第二年中我們設計較複雜及多樣化參數化模組，修改原** C++**參數化模組類別及設計流程以提昇功能階層模擬之效率，以及將參數化模組庫與視窗圖形式程式設計環境結合，讓設計者使用更加便利。**

關鍵詞: **模組庫、晶胞庫、參數化模組**

Abstract

The main objective of the project is to developing a parametric module design framework that is suitable for designing the baseband signal processing/datapath units. The framework is composed of C++ hardware related data classes, parametric module libraries and the cell libraries. In this framework, designers specify the system parameters and use the module generators to generate the C++ function simulating code and gate-level Verilog code used for physical design. The design flow achieves high-level abstraction and high reusability, short design time, and less prone to human errors. In the first year we focused on the design and verification of the cell libraries using TSMC 0.6/0.35μm technologies. Moreover, we proposed a C++ parametric module class, and use the class to design several simple modules. In the second year we try to design more modules, provide module with a variety of architectures, refine the hardware data classes to increase the efficiency of function simulation, and slightly modify the design framework. On the other hand, we integrate the module libraries into conventional C/C++ window-based programming environments to provide a friendly user interface and better class encapsulation.

Keywords: module library, cell library, parametric module

二、計劃緣由與目的

本計劃目的在發展通訊系統訊號處理統晶片使用之模組庫及模組自動產生系統。其中參數化模組庫除多種數位訊號處理模組產生器之設計外還包含金氧半製程之高效能低功率晶胞

庫之設計。使用者利用模組產生器設定模組參數需求之後，系統可自動輸出閘階層電路模組與對映功能模擬程式碼。該驗證程式碼可供較高階層之系統模擬使用，而閘階層電路模組配合上述晶胞庫，再利用目前商用自動佈局軟體即可迅速完成晶片之實體設計。晶胞庫設計部份包含利用國科會晶片製作中心提供TSMC 0.6μm 1P3M／0.35μm 1P4M 兩製程設計電路、完成實體與抽像佈局設計，並針對各晶胞進行完整功能模擬與驗證、參數萃取並對製程變異／溫度變化進行廣泛測試。此部份於計劃第一年完成。參數化模組設計環境之建立包含實作參數化模組抽象類別、常用數位訊號處理模組之架構選取／實作、程式執行結果驗證(驗證合成模組之功能正確性)、與圖形設計環境之提供。依據計劃時程第二年中我們設計更複雜之上層模組，並配合現有之視窗程式設計軟體提供使用者界面。此外為增加功能模擬效率，我們修改第一年提出的參數化模組抽象類別(此為模組之核心)及對映模組設計流程，加入新的功能，修正已知的程式錯誤並開始撰寫程式註解及使用手冊。以下針對模組設計環境部份進一步說明。

三、研究方法與結果討論

3.1 C++ 參數化模組架構

圖(一)為目前定義之 ASIC 系統晶片設計流程。C++參數化模組設計流程旨在整合訊號處理系統設計之前端(front-end)工作。一般訊號處理系統具有高度資料路徑特性(datapath intensive)，即模組由近似之子模組構成，子模組間連結遠比一般控制電路多，但整體系結構卻較為規則(多為陣列或樹狀型式)。此特性致使資料路徑電路較不適由通用軟體作邏輯合成。早期設計者多以位元切片(bit-slice)結構設計資料路徑模組(製作 hard-IP，輸出佈局)，期能提供可變匯流排寬度之模組，提高設計之重複使用率。然在目前模組複雜度增加，架構亦為多樣化的狀況下利用較高階描述進行參數化模組之設計，產生 RTL 或閘階層描述再行實體設計(soft-IP 或 firm-IP)已為趨勢。在此我們提出的設計流程考量如下：

- 資料路徑模組之細部結構由參數化模組根據給定參數求出，而非利用邏輯合成軟體產生。用於設計的程式語言除具一般設計功能外還需有足夠的應用程式以利模組設計者除錯。
- 系統依據模組結構資訊自動產生高階模擬程式碼與閘階層電路表示。高階模擬的效率相當重要。
- 模組提供參數資訊以利設計者作可行性評估或架構比較。

我們據此以 C++為模組設計語言，並規劃基礎類別 BaseModule 供各種參數化模組繼承。類別納入功能如：

- 模組物件建構解構、記憶體管理
- 模組命名、型態判別與註冊機制
- 依據給定參數配置電路結構
- 安排／移除管線(pipeline)結構或邏輯閘輸出之緩衝器
- 產生高階模擬程式碼(亦為硬體導向之 C++程式)與 Verilog 閘階層電路
- 傳回如路徑延遲時間(利用靜態時序分析得)、邏輯閘數、扇入扇出數、佈局面積、消耗功率等模組參數。此處在於快速估計參數值。因參數估計函數均宣告為虛擬(virtual)，模組設計者若需更精確參數估計可自行設計並重載該函數。

此外參數化模組設計時須指定模組結構資訊(structural informantion)。因 C/C++ 非為硬體描述設計，我們另外定義硬體變數類別(圖(二))以描述模組連結，其主要特點為：

- 儲存模組連結圖形(graph)資訊。
- 任意精確度定點數(至 2^{32} 位元)具備

算數／邏輯／比較運算等功能。
- 可作位元階層定址。
- 支援變數之連結語法，即由不同訊號的排合組合產生新的訊號。

如圖(三)，藉由上述硬體類別我們可用C++描述精確之模組結構資訊。

3.2 模組設計流程改變

在第一年中我們提出的參數化模組包含功能模擬的功能，即模組產生器接受參數後即可依據輸入資料進行運算，確定功能符合標準後再產生閘階層電路模組。其後發現，參數化模組受限於需儲存模組結構資訊，資料結構相當龐大且進行運算實缺乏效率，故第二年中我們將線上功能模擬部份自模組產生器抽離。其作法在新增模組產生 C++功能模擬程式碼功能。模組產生器產生模擬程式碼後可配合其它程式碼編譯並執行功能模擬，如此可降低功能模擬所需記憶體，並提昇模擬效率。

3.3 同部電路之功能模擬模型

我們希望利用C++撰寫並執行cycle-based的模擬，因為對於同步電路模組此精確度已足夠。我們提出一般模組產生器輸出功能模擬程式碼宣告如圖(四)左。不同模組均宣告新類別，類別內定義對映於各模組連結之變數、子模組及成員函數。其中Evaluate函數執行模組功能模擬。Update函數更新模組內狀態變數（即正反器輸出），對映於連接至正反器之時脈訊號。Clear函數重置模組內狀態變數，對映於連接至正反器之重置訊號。圖(四)右為功能模擬主迴圈。系統先呼叫Clear函數以重置該模組。在每個時脈週期內，給定新輸入後呼叫一次Evaluate函數，若模組內含正反器則呼叫 Update函數（呼叫 Update函數代表一次時脈觸發）。接著就可以讀取功能模擬結果。如圖(五)，這種同步功能模擬模型若加入重取樣模組提供非同步觸發，亦可模擬一搬通信系統常見傳輸與接收端之非同步狀況。

3.4 參數化模組功能模擬方式

在第一年中我們提出參數化模組執行功能模擬的方式為迭次完成。如圖(六)，因並未假設子模組按資料流順序執行 Evaluate 函數，若資料流的方向(白色箭頭)為由右至左，而子模組執行計算先後為反向(按子模組 1、2、3、4 執行)，則每個子模組計算一次後，輸出仍不會正確。需要經過多次計算待其收斂，我們評估這種功能模擬方式所耗時間約與系統複雜度呈平方相關，因此仍需再改善。

第二年中我們嘗試修改文獻中之層級化(levelization)相關演算法以求出子模組資料相關性(data dependency)，因而得到子模組之執行順序。層級化演算法如圖(七)，此程序相當於利用單位延遲模型(unity-delay model)進行靜態時序分析，迭次求出各模組輸出延遲時間，排列子模組最大延遲時間即得其資料相關性。文獻所提數種相關演算法均假設系統已攤平(flatten)至無階層式結構，故不能應用在此參數化模組系統。在此我們提出適用於階層式結構演算法並用以實作求取資料相關性之函數，因細節較複雜故不贅述。

圖(八)為 C++功能模擬之範例。此處為 piecewise-hyperbolic 數位式內差模擬。使用者需於主函數中寫入模擬迴圈(圖左)。此外尚可加入其它程式碼用以分析或顯示模擬結果(圖右)。圖型顯示輸入為純弦波，而模組輸出因量化關係將產生額外 DC 成份。

3.5 設計環境之整合

提供圖型設計環境除可讓使用者方便使用系統外，亦可間接避免使用者更動程式核心。在此我們嘗試將模組產生器設計與 BCB (Borland C++ Builder) 程式設計環境整合。這部份包括將各種模組產生器包裝成 Borland Non-VCL 元件，置入元件盤供使用者點選外，還包括設計圖形界面用以瀏覽參數化模組內部結構/模組資訊，並呼叫參數化模組提供的功能。圖(九)為模組產生器在 BCB 整合設計環境(IDE)下的工作情形。使用者將模組產生器元件加入專案後設定連結與細部參數；元件將此時模組的延遲時間、邏輯閘數、佈局大小等參數均顯示於物件瀏覽器中。

圖(十)為參數化模組瀏覽器，用於在程式執行時間瀏覽模組資訊。該瀏覽器似於 MS Window 檔案總管，表列模組(此處為前述內插器)之樹狀層級結構，輸出入埠規格及子模組資訊。點選子模組可以以該子模組為數根(root)看到其下之子模組詳細資料。切換至其它頁還可觀察模組之靜態時序分析(圖(十一))、資料相關性或其它參數估計結果。圖(十二)為一實作 54x54 平行式樹狀乘法器範例，我們列出其參數估計結果，並進行實體設計以求取真實參數。在比較中可發現如晶片面積之參數估計誤差較小(7%，為一般可接受範圍)，而線路延遲因未避開 false path，其結果較實際延遲時間高出甚多。

四、計劃成果自評

本計劃參數化模組產生器設計部份，目前共計完成參數化模組有：

- 各種 Cell Array (底層模組)
- Binary Adder (CPA, CSA, and CLA)
- Comparator
- Wallace Tree Adder
- Accumulator/Integrator
- Array Multiplier (Constant/Variable, signed/unsigned)
- Baugh-Wooley/Booth (radix-4) Tree Multiplier
- Serial-Parallel Multiplier
- Constant/Variable Coefficient Filter (Direct/Transposed Form)
- Synchronous Programmable Up-Down Counter with Strobe/Reset
- Shifter and FIFO
- Complex Multiplier
- ROM, Sine-Cosine ROM、NCO
- MAC, Correlator
- Interpolator (Linear, Piecewise-Hyperbolic, and Cubic Type)
- FFT processor

其中 ROM 的部份為 RTL 輸出需利用其它軟體合成電路，其餘參數化模組產生器均可輸出閘階層電路，達成計劃預期目標。為降低模組延遲時間，減少配置管線級(pipeline)需求，如濾波器及內差器等複雜模組均有製作 Carry-Save 型態的版本。

表(一)為本系統與當前模組設計系統之比較。我們提出的 C++系統設計架構嘗試整合資料路徑系統前端設計流程，並發展常用基頻訊號處理之參數化模組與底層晶胞庫。系統只需利用一般 ANSI C++ 編譯器即可於任一平台編譯執行。電路設計者可撰寫 C++ 原始檔宣告模組或以視窗界面設定模組參數，產生功能模擬程式碼與閘階層硬體描述，進行功能模擬或利用底層晶胞庫進行自動佈局。因可利用既有模組建構複雜度更高的參數化模組或系統，預期可大幅縮短常用基頻訊號處理模組設計與驗證時間，減少設計者發生的錯誤。此部份研究成果已於今年 IEEE 國際電路與系統會議發表。就今後發展方向而言，未來可將晶片測試如 Boundary Scan 或 Arithmetic BIST 電路加入參數化模組

產生器以加強模組之可測試性(testability)。此外應加強與邏輯合成軟體(如 Synopsis)間整合，以處理控制型態電路。

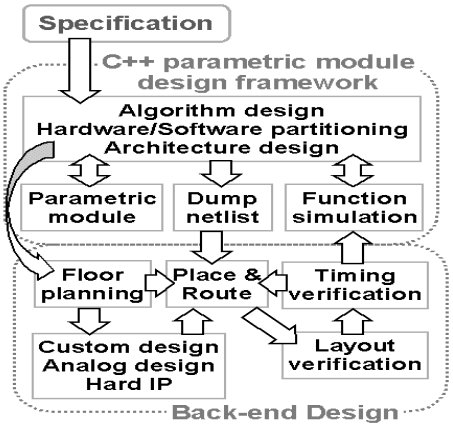
五、參考文獻

1. Neil H. E., *Principle of CMOS VLSI design : a systems perspective –2nd ed.*, 1992.
2. J. Sanguinetti, "Bridging the design gap with Cynthesis," CynApps Company web page: http://www.cynapps.com/, 1999.
3. Frontier Design Press, "C spans floating to fixed-point gap," Frontier Design Company web page: http://www.frontierd.com/, Jun. 1999.
4. J. Nurmi, "Portability Methods in parametrized DSP Module Generators," *Proc. IEEE Workshop on VLSI Signal Processing,* pp. 260-268, Oct. 1993.

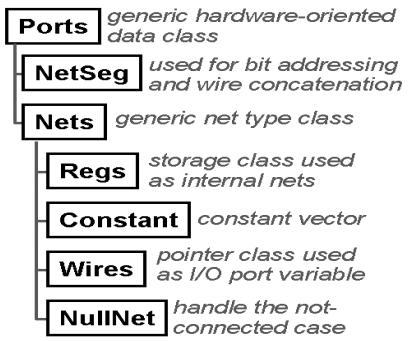
六、已發表論文

Ming-Luen Liou and Tzi-Dar Chiueh, "A Parametric Module Design Framework and Its Application to Gate-level Datapath/DSP Module Synthesis," *Proc. 2000 IEEE Int. Symp. Circuits Syst.,* Vol. 2, pp.41-44, May. 2000.

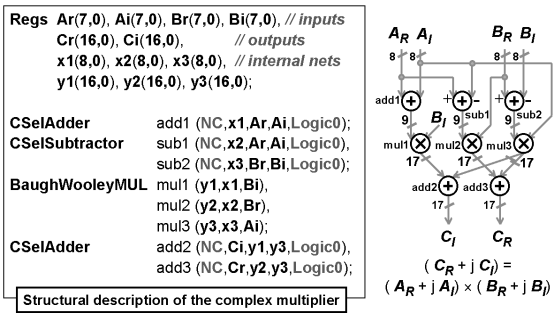
七、圖表



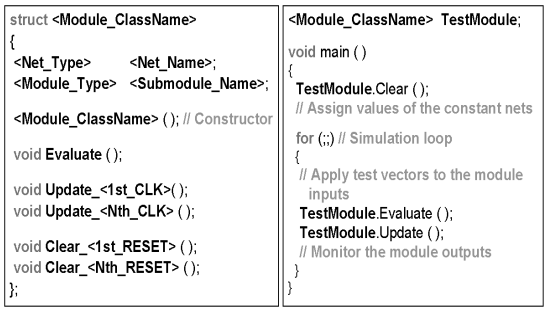
圖(一)、ASIC 系統設計流程與 C++參數化模組設計架構之關係



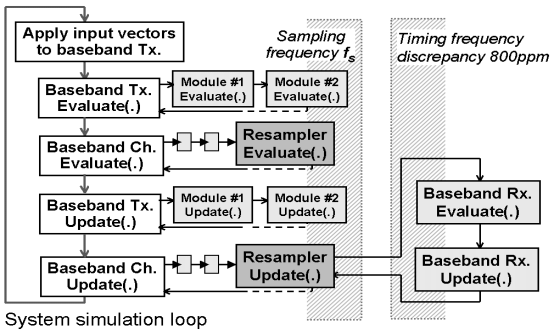
圖(二)、硬體變數類別之繼承關係



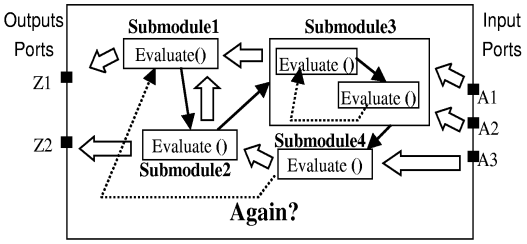
圖(三)、參數化模組之結構描述 (C++)



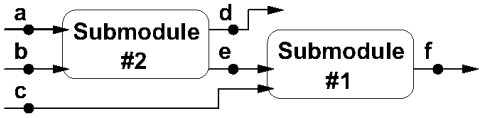
圖(四)、同步系統 C++功能模擬模型



圖(五)、適用於非同步系統模擬實例



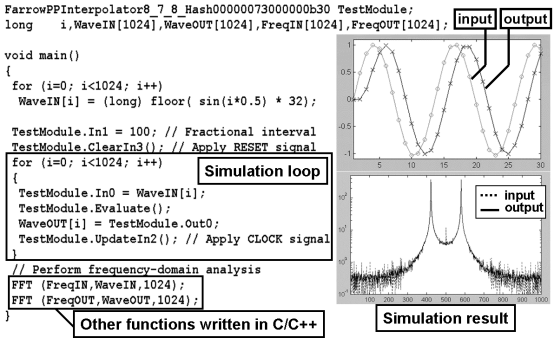
圖(六)、遞回式功能模擬示意圖



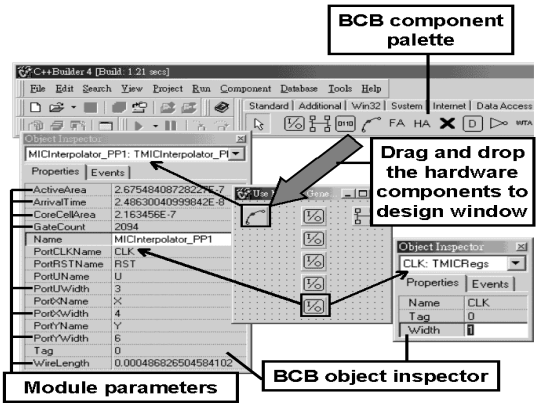
Submodule #1： f = max {e,c} +1

Submodule #2： d = e = max {a,b} +1

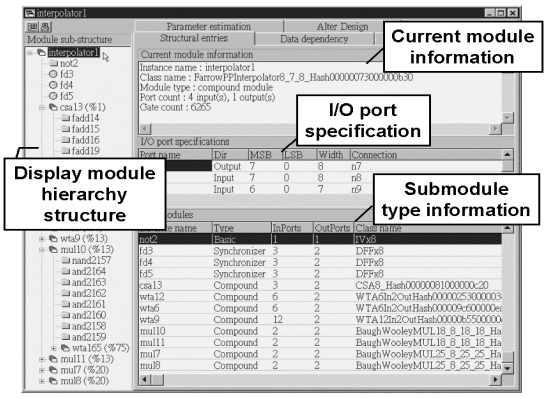
圖(七) 層級化過程(levelization process)之示意圖



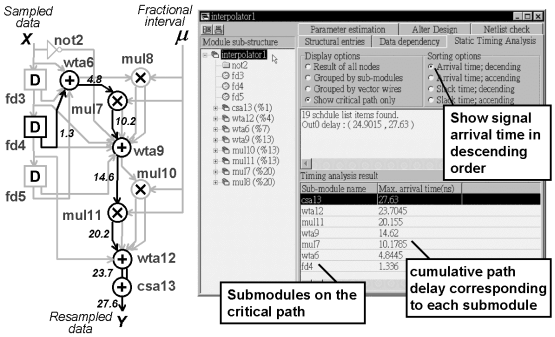
圖(八)、模組功能模擬輸出



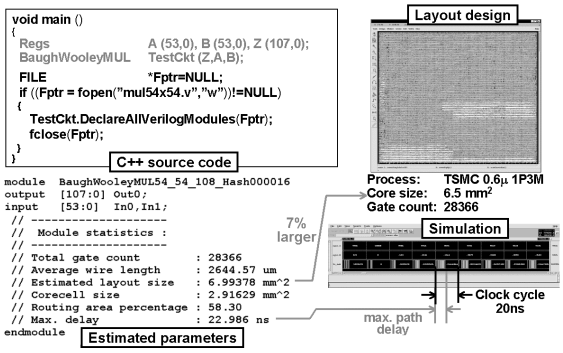
圖(九)、BCB 設計環境之整合



圖(十)、視窗界面之模組瀏覽器外觀



圖(十一)、模組靜態時序分析輸出



圖(十二)、模組參數估計函數輸出與實體設計結果比較

	Traditional design flow	Synopsis Module Compiler	C-to-RTL design framework [2][3]	Proposed design framework
Abstraction	low	high	high	high
Design language	C/C++ and HDL	MCL	C/C++	C++
Modeling	behavior data-flow structural	behavior data-flow	behavior data-flow	structural
Required tools for behavior simulation and netlisting	C/C++ compiler	Module Compiler HDL simulator	C++ compiler C-RTL translator logic synthesizer	C++ compiler
Functions encapsulated in C/C++ modules	behavior simulation	N/A	behavior simulation	beh. simulation netlisting param. estimation (more...)
Simulation model	cycle-based and event-driven	event-driven	cycle-based	cycle-based
Design reusability	low	high	high	high

表(一)、資料路徑模組設計系統比較