* DEC 28 2008

# [[Verilog]基於SystemC的軟硬體協同驗證](https://mouein.pixnet.net/blog/post/22750065)

**作者：**同濟大學超大型積體電路研究所 張志傑 林濤 **日期：**2007-2-1

**來源：**本網

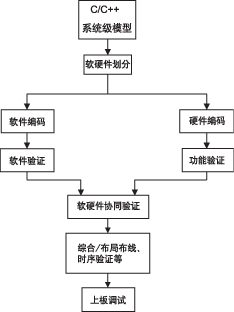
積體電路設計規模越來越大，速度越來越快，複雜度也越來越高，因此有必要在更高的抽象級別對設計物件進行描述和驗證，以實現更快的模擬速度及軟硬體的協同驗證。但是傳統的設計在前期完成軟硬體的劃分後，對軟硬體部分進行單獨設計，在軟硬體的整合平臺沒有完成之前，很少進行軟硬體的協同模擬驗證。許多重要的性質如性能、成本、即時性等，很大程度上是在設計早期決定的，在傳統設計流程中，要到設計的後階段進行系統整合和驗證時才能發現，這時再返回設計的開始階段修改設計，必然會大大延長設計週期，增加設計成本，其工作量往往也很大。  
隨著開發語言的不斷完善，出現了些支援更高抽象級別的設計語言，如SystemC、System Verilog等，這些語言較好地實現了軟硬體的協同驗證。其中SystemC以更接近於C++的風格特點，而受到設計者的青睞。本文介紹了採用SystemC語言，在設計的前期實現軟硬體協同驗證的方法。

## SystemC的簡介

SystemC是由開放性的SystemC聯盟組織(Open SystemC Initiative，OSCT)推出的一種系統級設計和驗證語言，SystemC是基於C++的建模平臺，其本質是在C++的基礎上新增了硬體擴展庫和模擬核，支援EDA設計中的各個抽象層次，如暫存器級、行為級、系統級的建模，能夠表達併發性、即時性、互動性等硬體模型的概念。  
SystemC完全支援C++的資料類型，如果從系統級建模而言，完全可以將SystemC作為C++來處理，可以使用標準的C++語言開發工具來建立、模擬、調試和探索設計物件的各種體系結構和演算法描述。在本文中，對SystemC的調試採用了Visual C++ 6.0，對其進行的模擬用了ModelSim SE 6.1f。  
使用SystemC，可以像使用HDL" onclick="tagshow(event)" class="t\_tag">VHDL或者Verilog語言一樣進行RTL級和行為級的建模。SystemC中主要的類包括模組(modules)、進程(processes)、埠(ports)、信號(signals)等，還有一個調度內核，負責在每一個時鐘週期調度進程和更新信號值。為了支援暫存器傳輸級的並行描述，SystemC採用了與傳統硬體描述語言基本相同的調度模式——基於△(delta)延時。一個△週期包括求值和更新兩個階段，在一個時間點上，這樣的△週期會持續出現，直到再求值前後的結果不再發生變化。  
SystemC中的模組用關鍵字SC\_MODULE來定義，其埠定義與Verilog類似。當存在多個模組時，在SystemC中用頂層函數sc\_main()來實現各個模組的連接，沒有該函數的代碼是無法編譯的。在SystemC中，進程是一個基本的執行單元，它被調用來模擬目標系統的行為。進程的行為是多樣化的，可以實現某個函數的 功能，也可以在運行過程中被掛起，並且進程是並存執行的，一個進程中不能包含或直接調用其他進程。SystemC的進程主要有兩種，方法進程 (SC\_METHOD)和執行緒進程(SC\_THREAD)。方法進程是唯一可以綜合的RTL進程，它的特點是當敏感表上有事件發生，它就被調用，調用後立 即返回。只有該類進程返回後，模擬系統的時間才有可能前進，因此該類進程不能被掛起。執行緒進程不是RTL級進程，它可以被掛起和重新啟動，所以它的一個用 途是用來描述驗證平臺(testbench)。

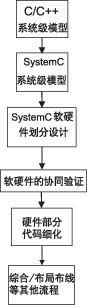
## 基於SystemC的硬體軟體協同驗證技術

在傳統的系統設計中，系統設計人員自己選擇語言(通常是C或C++)編寫可執行程式， 調試並驗證功能後，對系統進行軟硬體的劃分，並將需要硬體實現的部分交給RTL設計組，RTL設計組再在C/C++基礎上重新編寫設計物件，以將其綜合成 門電路，該流程如圖1所示。在這個過程中，軟體部分往往是用C/C++來編寫的，對這部分的驗證可以用Visual C++等來實現，而硬體部分通常是用Verilog或VHDL來編寫的，這就需要設計者將C/C++轉換為HDL，對其驗證也只能單獨用Modelsim 等工具來完成，這樣很難在早期實現協同驗證，必須等軟硬體都各自驗證完成，再用基於軟硬體的整合平臺來驗證，但此時設計已經基本完成，如果發現錯誤，其修 改工作量是很可觀的。



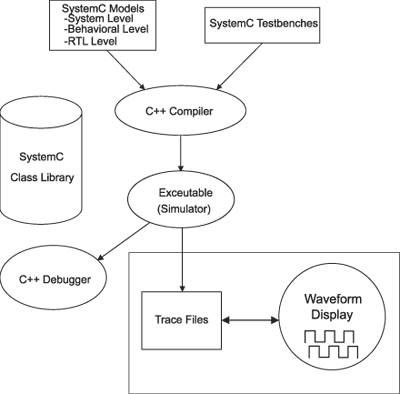
**圖1  傳統設計過程圖**

與傳統的方法相比，基於SystemC的設計流程與以前的設計流程本質區別在於，使用一種統一的語言就可以完成從系統級到RTL，從軟體到硬體的全部設 計，整個設計的軟硬體可以協同設計和模擬。SystemC提供了在高抽象級別使用抽象資料類型對系統的硬體部分進行建模，支援軟硬體早期的協同驗證。圖2 為基於SystemC的設計流程，從C/C++到HDL的過程不需要再作轉換，可以用一種語言完成所有設計和驗證。



**圖2  SystemC設計流程圖**

在圖2中，由於SystemC實質上就是C++，所以把C/C++的模型轉換為SystemC的模型並沒有多大難度，設計者完全可以用SystemC進行 設計，並用它生成各種測試基準進行驗證。同時軟硬體協同驗證的階段也提前到了設計的初期，這樣在硬體平臺實現之前就可以對設計的合理性和可行性做出充分的 評價，並可以儘早對設計進行優化。  
用SystemC進行軟硬體協同設計和驗證的具體過程如圖3所示。通過使用SystemC類庫，設計者可以編寫目標系統的系統層(system level)、行為層(behavioral level)或RTL層的SystemC模型代碼。SystemC類庫在其中起到兩個作用：一是提供了描述硬體行為如併發、層次化模組、埠、時鐘等這些 類的實現；二是提供了一個輕型的進程調度核。使用標準的C++編譯器，如Visual C++等，可以將設計者編寫的SystemC代碼進行編譯、連結，得到一個可執行的程式，該程式的執行的結果就是設計者所設計的目標系統的類比運行，用於驗證設計正確與否的各種測試基準同樣使用SystemC編寫，並且隨同設計一起編譯，SystemC的代碼調試可以使用任何常見的C++調試環境進行調試，如Visual C++。另外在模擬中還可以產生波形檔，並使用常見的波形顯示工具如Modelsim等進行分析。



**圖3  SystemC設計流程圖**

上面提到的在系統級和行為級時，軟硬體劃分之後，對整個系統的功能用SystemC來實現軟硬體的協同驗證，接下來就是設計中硬體部分的細化，在RTL級 實現。一種理想的情況當然是在RTL級也一直沿用SystemC，對系統級的SystemC進一步的細化到RTL級。但事實上，現在SystemC對 RTL級的支援並不夠，所以將SystemC轉換為可綜合的HDL就不可避免。  
SystemC不僅可以在高抽象層次單獨對設計進行軟硬體的協同驗證，還可以和Verilog或VHDL一起進行聯合的驗證，這就為真正意義上的協同驗證提供了可能性。  
在ModelSim SE 6.1f及其以上的版本中，都支援對SystemC和HDL的混合驗證。在這種情況下，設計的硬體部分已經用HDL語言完成，而軟體部分完全可以用SystemC來實現。在整合平臺沒有完成時，只要能用SystemC實現軟硬體的通訊， 實現軟硬體協同驗證顯然可以做到。而SystemC充分考慮到這一點，在ModelSim SE 6.1f中很好地支援了混合的編譯和驗證。在ModelSim SE 6.1f中，對Verilog的module編譯用vlog命令，對SystemC的module編譯用sccom-g命令，編譯完成後用sccom- link命令將所有的物件連接起來，從而可以完成編譯和模擬，具體的過程可以參閱ModelSim SE 6.1f的用戶手冊，本文也講在接下來的例子中有針對性地進行說明。

## 軟硬體協同驗證舉例

下面將以H.264解碼器為例，介紹如何進行軟硬體的協同驗證。限於篇幅，只對其中關鍵性的步驟給出相關的分析和介紹。  
根據上文中給出的過程，在完成對解碼器進行軟硬體劃分後，對軟硬體部分用SystemC語言來設計，並進行Visual C++下的協同驗證，限於篇幅，對這部分不做說明，設計者可以自己參考SystemC的語法完成。這之後，將硬體部分用Verilog實現。在完成軟體部 分和硬體部分各自單獨的驗證之後，為了實現軟硬體的協同模擬，在軟體部分加了一個實現系統匯流排傳輸功能的進程，而硬體部分的各個模組則作為接在系統匯流排上 的單元，這樣以來，軟體部分就可以通過進程實現與系統匯流排上的硬體模組通訊。實驗室中對H.264解碼器完成了軟硬體的劃分，並完成了行為級協同驗證，在 接下來的過程中本文將對讀者關心的幾個關鍵步驟進行簡單介紹。

### C/C++向SystemC的轉化

由上文中可以看出，C/C++模型向SystemC模型的轉換是協同驗證的開端。C語言向SystemC的轉換，可以認為是C語言向C++轉換的過程。在H.264解碼器設計中，軟體部分是用C語言描述的，故而這裡將著重完成C模型向SystemC模型的轉換。  
在所有的C語言代碼中，程式的開始都是main()函數，通過main函式呼叫其他函數。而在SystemC中，主要成分則是進程。於是很容易想到用進程 來代替函數的功能，但是直接的轉換是行不通的，因為進程是並行的。根據H.264解碼器的C語言代碼，把它整個作為一個module，先定義 SystemC的一個頂層模組SC\_MODULE(H264top)，接下來對H264top這個模組定義埠和進程，對埠的定義可以根據具體情況而 定，而對於進程的定義，必須把原來的C語言的main函式定義成一個方法進程SC\_METHOD(prc\_main)，這個取名為prc\_main進程在 功能上完全等同於main函數，通過這個進程來調用原先的其他函數。完成這些轉換後，再加上一個頂層函數sc\_main()來實現連接，這樣軟體部分向 SystemC的轉換就基本完成，可以到Visual C++中編譯調試了。

### 完成軟硬體部分的結合，實現協同模擬

這一步是整個協同模擬中最為關鍵的一步。一般意義上，軟體部分和硬體部分無法直接連接，它們之間必須要有一條系統匯流排來實現通訊。  
在SystemC改寫的軟體模組中，為了要實現軟硬體的互動，用Modelsim實現軟硬體協同模擬，定義了一個執行緒進程SW2HW，用於軟體與硬體之間 的通訊。在用SystemC實現這這個進程時，要注意使它的行為符合系統匯流排的傳輸協定。根據SystemC語法，SW2HW作為模組H264top的一 個進程，它與方法進程prc\_main是並行的，並且兩者不可以互相調用，那麼更談不上互相通訊。為瞭解決這個矛盾，SystemC中指出，進程可以和模 塊的其他成員函數相互調用。於是在實現軟硬體通訊的進程SW2HW後，還需要把原來在進程prc\_main中的，需要和硬體互動的函數重新在模組頂層模組 H264top中定義為成員函數。這樣軟硬體的互動才能真正實現。

## 結語

以往，硬體軟體協同驗證技術往往是使用一些複雜的資料結構作為描述、分析和綜合的平臺，這樣的結構只能稱為虛擬平臺，它需要複雜的工具來整合不同軟硬體設 計語言和進行協合驗證。本文的研究採用一個可執行的SystemC描述來代替這個虛擬平臺，在設計過程中，設計者可以使用一個可執行的SystemC描述 做為設計物件，在完成RTL級的設計後，也可以通過SystemC編寫的通訊機制實現軟硬體的即時通訊，達到軟硬體協同模擬的目的。