片上系统设计中软硬件协同验证方法的研究

严迎建[***]　　刘明业[*]
[*](北京理工大学 ASIC 研究所 北京 100081)
[**](信息工程大学电子技术学院 郑州 450004)

摘　要：讨论一种面向片上系统(SOC)设计的基于指令集模拟器和硬件模拟器的软硬件协同验证方法。该方法能够在 SOC 设计的早期对整个系统功能进行验证，能够为设计者提供一个纯虚拟的软硬件协同验证环境。重点讨论协同模拟过程中软硬件交互事件的产生和处理方法，以及软硬件模拟器之间的同步和优化方法，并且给出了事件驱动硬件模拟器的协同模拟控制算法。最后给出了一个基于 ARM7TDMI 的设计验证实例。

关键词：片上系统，协同验证，协同模拟，指令集模拟器
中图分类号：TP337　　**文献标识码**：A　　**文章编号**：1009-5896(2005)02-0317-05

A Hardware-Software Co-verification Method for SOC Design

Yan Ying-jian[***]　　Liu Ming-ye[*]
[*](*ASIC Research Center*, B*eijing Institute of Technology*, *Beijing* 100081, *China*)
[**](*College of Electronic Technology*, *Info. Eng. University*, *Zhengzhou* 450004, *China*)

Abstract　This paper presents a hardware-software co-verification method for SOC design, which is based on instruction set simulator and hardware simulator, and used to validate function of SOC in the early design phase. The generating and processing methods of interactive events between hardware and software simulator during co-simulation are discussed in detail. An algorithm of synchronizing between hardware and software simulator is presented, and to reduce the synchronization overhead, some optimizing methods are introduced. Finally, an co-verification example of a design based on ARM7TDMI is given.

Key words　System-on- chip, Co-verification, Co-simulation,　Instruction set simulator

1　引言

随着集成电路技术的发展，出现了片上系统 (System On Chip, SOC)，一种高度集成的单芯片嵌入式系统，它将信号采集/转换、输入/输出接口、存储器、微控制器等集成在单个芯片上，再配上实时操作系统(RTOS)、应用软件等系统所需的软件模块，在一块芯片上实现信号采集、转换、存储、处理等功能，从而获得更高的系统性能。SOC 设计是以软硬件协调设计为主要设计方法的基于 IP 核重用的芯片设计技术，其所关注的焦点不再是某个新功能的设计与实现，而是 IP 核的选择和使用、软件与硬件功能的划分以及软硬件模块的集成和验证等。

SOC 的高度集成特性、高性能要求和其高额的设计/投片费用要求设计者在布局、布线及投片前，必须保证系统功能正确无误。因此在设计实现之前，设计者必须对系统功能进行验证，与普通芯片设计中的数字系统验证不同，SOC 设计中必须采用软硬件协同验证技术对包括软硬件在内的整个系统功能进行验证[1]，这也是软硬件协调设计过程中的关键步骤。软硬件协同验证使得设计者可以在设计的早期发现系统功能、软硬件划分、软硬件之间的接口以及设计中的其它问题，在硬件原型构造之前，就能在一个纯“软”的环境中对系统进行软硬件集成测试。本文讨论一种基于指令集模拟器和硬件模拟器的软硬件协同模拟验证方法，该方法能够在 SOC 设计的早期对整个系统功能进行验证。

2　协同模拟验证方法

由于 SOC 规模庞大、结构复杂，如果采用硬件模拟器进行全芯片的功能验证，必然耗费大量的时间，所以在进行详细的门级验证之前，必须对整个系统功能进行验证。

2003-08-25 收到，2003-12-19 改回
国家部级基金资助项目

基于指令集模拟器和硬件模拟器的软硬件协同模拟技术是一种高效、低代价的系统功能验证方案[2]，并且由于软件是以解释的形式执行，可以很方便地对软件进行调试。由于指令集模拟器和硬件模拟器都是常用的软件和硬件模拟工具，上述协同模拟方法完全符合软件和硬件设计工程师的设计习惯。采用上述协同模拟技术，作者在 WINDOWS 平台上构造了一个面向 SOC 设计的软硬协同验证环境，如图1所示为该环境的总体结构。

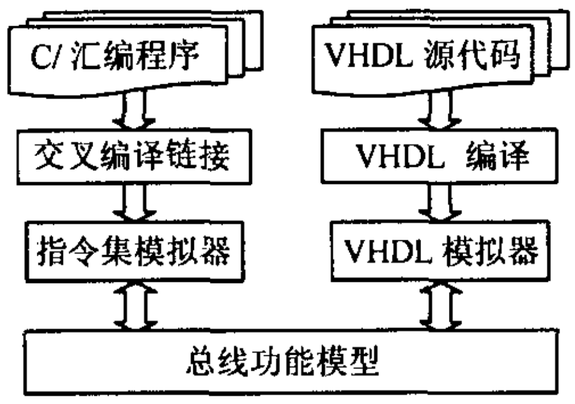

图1 协同验证环境总体结构

软件采用汇编（面向特定嵌入式处理器），C，C++等语言中的一种或几种语言混合编写，用户编写的源程序经过交叉编译器和连接器（如 GCC 工具）编译链接形成 ELF 等格式的文件，输入到指令集模拟器进行软件模拟。处理器外围硬件行为模型采用 VHDL 硬件描述语言设计，VHDL 源描述经过编译后，由硬件模拟器模拟。在 SOC 工作过程中，运行于处理器核上的软件通过总线和外围硬件进行信息交互，而对 SOC 进行软硬件协同模拟过程中，“运行”于指令集模拟器上的软件则需要通过总线功能模型与硬件模拟器上的外围硬件进行信息交互。在本协同验证环境中，总线功能模型采用 C++语言构造，指令集模拟器通常也是采用 C 语言或 C++构造，因此可以将总线功能模型与指令集模拟器链接成一个程序——软件模拟器——来模拟整个处理器核在系统运行过程中的活动，换句话说就是指令集模拟器与总线功能模型共同组成了处理器模型。软硬件模拟器之间通过 Windows sockets 进行通信。

本环境中硬件模拟器采用编译型事件驱动 VHDL 模拟器——BCVS(BIT Compiled VHDL Simulator)，该模拟器属于北京理工大学 ASIC 研究所开发的数字系统自动设计工具 Talent2000 的模拟验证子系统[3]。指令集模拟器是一种基于处理器指令集体系结构(ISA)模型的软件模拟技术，通过模拟每条指令在目标处理器上的执行效果来模拟目标机软件的执行过程。因此，如果要支持某种处理器进行 SOC 设计验证，就需要添加该种处理器的指令集模拟器和总线功能模型。ARM7TDMI 处理器是 ARM 公司的 ARM7 处理器系列成员之一，是目前广泛应用的 32 位高性能嵌入式 RISC 处理器核，市场上已有众多采用该处理器核的 SOC 芯片。因此，作者首先利用采用解释性技术[4]和 C++构造了 ARM 体系结构版本 4 指令集模拟器，然后采用 C++构造了 ARM7TDMI 总线功能模型，并将二者链接成 ARM7TDMI 软件模拟器，进行面向单处理器 SOC 设计的软硬件协同模拟验证方法研究。指令集模拟器和硬件模拟器的实现技术在相关文献中已有较多论述[4,5]，所以下面将重点讨论协同模拟过程中软硬件交互事件产生和处理方法，以及软硬件模拟器之间的同步及优化算法，这也是实现软硬件协同模拟的关键技术。

3 软硬件交互事件产生与处理

总线功能模型是软硬件模拟器进行协同模拟的交换界面。一方面，它根据指令集模拟器提供的指令执行情况以及处理器状态确定相应的总线时序，并以事件的形式发送给硬件模拟器；另外一方面它还要将硬件模拟器传送来的外围硬件的输入信号转换为中断请求、数据输入等，请求嵌入式软件（指令集模拟器）进行处理。因此，协同模拟中的软硬件交互事件实际上就是处理器总线信号变化事件，软硬件交互事件被定义为一个如{N_{port}，V_{value}，T_{time}，E_{type}}形式的四元组，其中 N_{port} 表示端口的名称；V_{value} 表示端口的新状态值；T_{time} 表示当前时间；E_{type} 表示事件类型，因为在协同模拟过程中，除了传递端口状态变化信息以外，还要通过事件对协同模拟进行控制，例如协同模拟的启动、停止、软硬件模拟器之间的同步等操作。

下面以 ARM7TDMI 的寄存器加载指令为例讨论软硬件交互事件产生与处理过程，如图 2 为该指令的指令周期时序图(部分信号)[5]，其中 MCLK 为时钟信号。

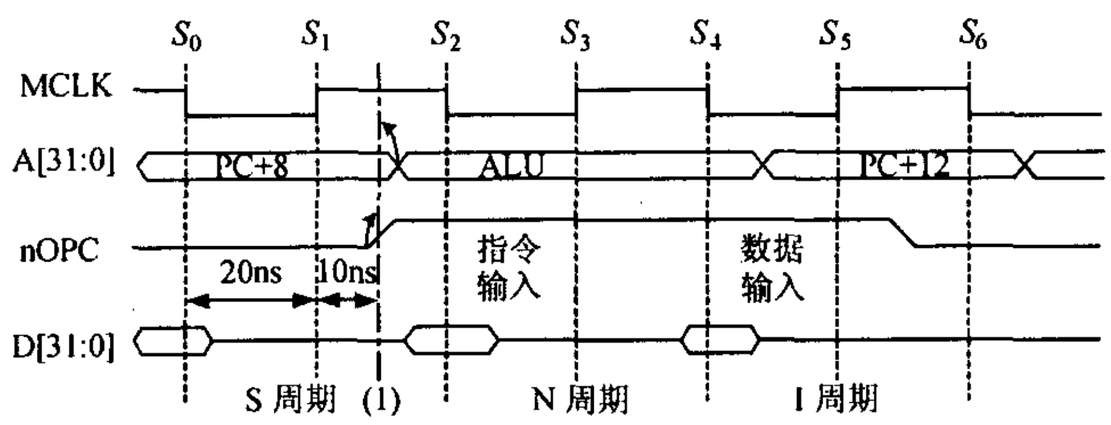

图2 ARM7TDMI 处理器寄存器加载指令周期时序

当 ARM7TDMI 指令集模拟器进行寄存器加载指令模拟时，其总线功能模型根据上述时序图产生相应的事件，各输出信号的状态每发生一次变化就需要产生一个事件。指令总线时序图只能精确到时钟边沿，事件产生时其时间值可以取为事件所在 MCLK 电平的中间位置，如图 2 中的点划线(1)所示，因为主要用于系统功能的验证，所以软硬件协同模拟验证的结果不会因此受到影响。事件产生后，通过 Windows sockets 被发送到硬件模拟器。

因为软硬件交互事件的意义和事件驱动硬件模拟器的事件意义基本相同，因此接收到总线事件后，硬件模拟器只要根据事件名称、时间以及处理器引脚和测试台中信号的映射关系，将事件插入到事件表的对应位置即可。硬件模拟完成后，硬件模拟器将外围硬件输入到总线的信号事件发送给总线功能模型。接收到上述事件后，总线功能模型再根据指令周期时序图，将事件转换为中断请求、数据输入等，请求指令集模拟器处理，最终实现软硬件模拟器的交互。因为总线功能模型只关心总线管脚在时钟边沿的状态，因此在处理事件时，如果同一信号在 MCLK 同一个电平内发生多次变化，只有最后一个事件的状态值为有效值，所以总线功能模型接收到事件后不能立即处理，而必须等到 MCLK 同一电平的全部事件接收完毕后，才能进行处理。

4　协同模拟同步及优化方法

4.1 协同模拟同步

指令集模拟器以指令为单位进行模拟，时钟每次推进一个指令周期，指令周期长度则随指令的不同而不同。硬件模拟器通常采用事件驱动模拟算法，它以事件为调度对象，硬件模拟器按照事件发生的时间进度推进时钟，时间单位一般由用户的设计指定，例如纳秒，微秒等。因此必须采用一定的机制对二者进行同步，并且它们之间能否同步将直接影响到协同模拟的正确性。指令集模拟器和硬件模拟器之间通常采用 Lock-step[6]方式进行同步。采用该同步方式必须首先确定同步点，保证在两个同步点之间的时间间隔内不发生软硬件交互事件。如前文所述，总线功能模型是软硬件模拟器之间的信息交换界面，所以同步点会因为总线功能的不同而不同。而硬件模拟器是以事件发生的时间顺序推进时钟，要保证其时钟不会越过同步点，就必须引入同步事件类型，即时间为同步点的事件，硬件模拟器调度到该事件时，需要向总线功能模型发回一个同步事件，说明硬件模拟器的时钟已推进到同步点。

下面就以 ARM7TDMI 处理器为例讨论软硬件协同模拟的同步方法，ARM7TDMI 总线对某些信号在 MCLK 上升沿检测，例如 nWait(等待信号，用于延长总线周期)，而对另外一些信号则需要在 MCLK 下降沿检测，例如 nFIQ(快速中断)，nIRQ(快速中断)等，所以同步点可以确定为 MCLK 的两个边沿。例如寄存器加载指令就需要 6 个同步点 S_1，S_2，S_3，S_4，S_5，S_6，如图 2 所示，即 MCLK 信号电平每发生一次变化就需要一次同步。图 3 为寄存器加载指令的协同模拟同步过程。

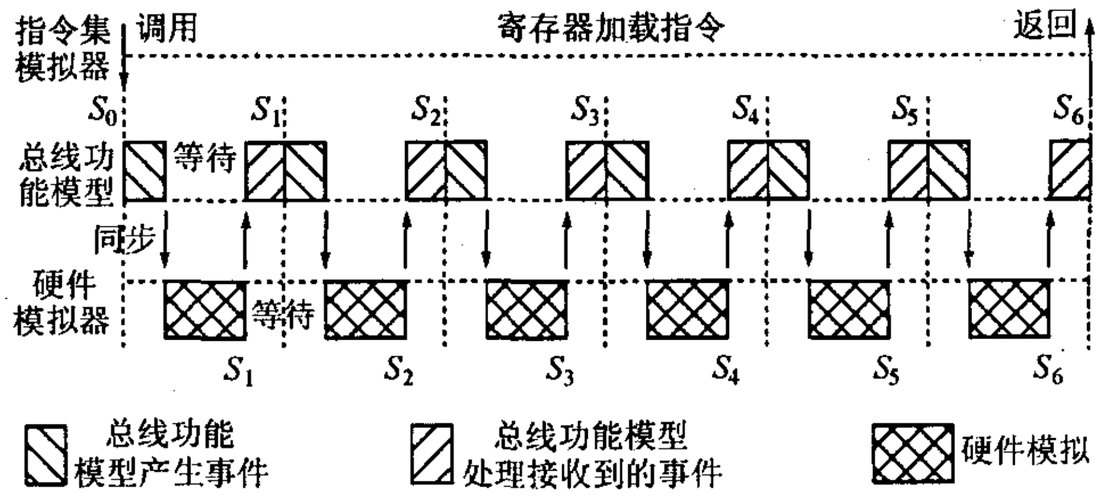

图 3　ARM7TDMI 寄存器加载指令的同步过程

当“执行”到一条寄存器加载指令时，存储器地址计算完成后，指令集模拟器调用总线功能模型的相应函数进行指令总线时序的模拟。总线功能模型总共需要与硬件模拟器进行 6 次同步，才能完成该指令周期的协同模拟，其中每一模拟步又分为 3 个步骤，下面以第 1 步为例讨论一个模拟步的处理过程：

(1) 总线功能模型根据指令周期时序产生本模拟步由软件触发的总线事件，并将事件发送到硬件模拟器，然后发送一个时间为下一个同步点例如 S_1 的同步事件。之后总线功能模型就处于等待状态，等待硬件模拟器完成本步模拟。

(2) 接收同步事件 S_1 后，硬件模拟器才能开始模拟，以保证不会越过同步点。推进到同步点后，硬件模拟器向总线功能模型发回同步信息，之后就处于等待状态，等待总线功能模型发送下一个同步点。

(3) 总线功能模型接收到硬件模拟器发回的同步信息后，根据指令总线时序处理本模拟步硬件模拟器发送的事件，到此完成了本模拟步的协同模拟。

紧接着再进行下一步的模拟，如此进行 6 次同步，总线功能模型将模拟结果返回到指令集模拟器，最终完成寄存器加载指令的协同模拟。

4.2　同步的优化

如前文所述，采用 Lock-step 方式进行协同模拟同步，一条寄存器加载指令就需要 6 次同步，如果不对同步进行一定的优化，必然会因为过多同步开销使得整个协同模拟效率大受影响。下面就讨论几种提高模拟效率的方法。

ARM7TDMI 处理器只有加载/存储指令才能访问存储器，其它的指令（例如数据处理类指令）运行期间，除取指令以外不进行任何总线访问，指令又是位于指令集模拟器的存储器模型，而不是位于硬件模拟器上的外围硬件模型，因此总线功能模型不需要产生相应的地址、存储器请求信号。另外对于加载/存储指令，例如上述寄存器加载指令，当所访问的地址在存储器空间内时，数据也是来自指令集模拟器的存储器模型，也无需产生地址、存储器请求

等信号。只有当指令访问的地址为 I/O 地址时，即通过存储器映射 I/O 访问外围硬件时，总线功能模型才需要产生相应的地址、存储器请求信号。

基于 ARM7TDMI 处理器核的 SOC 设计中，处理器主要通过存储器映射 I/O 访问外围硬件，外围硬件则通过中断信号请求软件服务的。ARM7TDMI 在当前指令运行中间无法响应中断，必须等到当前指令运行结束后，并且 ARM7TDMI 的中断请求信号 nFIQ，nIRQ 是电平触发的，所以只有在当前指令周期的最后检测到外围硬件有中断输入时，才需要请求指令集模拟器进行中断处理。因此对于所有不进行 I/O 访问的指令，例如数据处理指令，包括访问的地址不是 I/O 地址的寄存器加载/存储指令，不管其指令周期包括多少时钟周期，只需要一次同步，如图 4 所示。

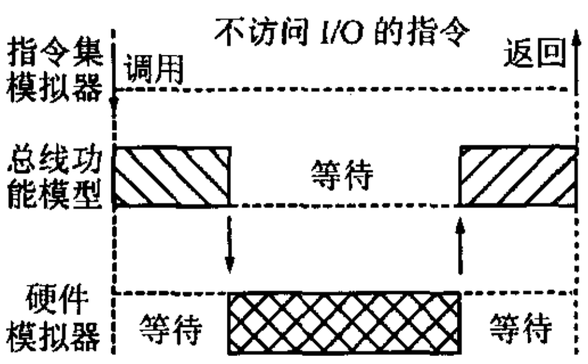

图 4 优化后的同步过程

协同模拟中，外围硬件设计并不一定用到所有的总线信号，并且有些信号不需要在软硬件模拟器之间进行传递，例如 MCLK 信号，该信号为时钟信号，其频率固定。因此，在协同模拟开始时，可以在软件模拟器中设定需要传递的信号，这样可以减少软硬件模拟器之间传递的信号数量。采用上述几种方法可以减少大量不必要的同步和总线事件传递，从而大大提高了协同模拟的效率。

5 硬件模拟器的协同模拟控制

为了进行协同模拟，必须对硬件模拟器进行适当修改。首先必须为硬件模拟器添加一个事件接收模块以接收总线功能模型发送的事件，我们采用一个单独的线程实现，该线程和硬件模拟主线程之间采用内存共享变量进行通信，如算法 1 所示。如果接收到的事件为总线端口事件，则根据事件时间将其插入到统一的事件表中。如果接收的事件为同步事件，则首先在事件表中插入一个时间等于同步点的空事件，以确保硬件模拟器在进行事件调度时不会跳过同步点，同时还要更新同步共享变量 Syn_Time (保存同步点时间)。如果接受到的是协同模拟结束事件，则将协同模拟结束变量 G_bCosimExit 置为 TRUE。另外硬件模拟的运行控制算法也必须进行适当修改，如算法 2 所示。当时钟推进到同步点后，硬件模拟器向总线功能模型发回一个同步事件，然后硬件模拟器就处于等待状态，直到 Syn_Time 大于当前时间值(即下一个同步事件的到达)。

算法 1 事件接收线程

```
while(!G_bCosimExit)
{
  switch (事件类型)
    {
    case Bus_Event:
          插入事件表
    case Syn. Event :
          将同步事件插入事件表
          Syn_Time= 事件时间
    case Cosim_Exit :
          G_bCosimExit=TURE
    }
}
```

算法 2 硬件模拟线程

```
while(! G_bCosimExit )
{
    if (调度到同步事件)
    {
        向总线功能模型发回同步事件
        while( Syn_Time==当前时钟&&!
              G_bCosimExit )
        {
            等待直到 Syn_ Time > 当前时钟
        }
    }
    处理下一个事件
}
```

6 举例

为了验证本文讨论的协同验证方法，作者设计了一个基于 ARM7TDMI 处理器核数控机床控制系统，并对该系统进行软硬件协同模拟验证。利用该系统，用户可以根据工件加工图纸的要求编写数控程序，控制数控机床 3 个步进电机的运动，而 3 个电机又分别控制工件在 X，Y，Z 3 个坐标方向的运动。该系统硬件部分由一个可编程定时器/计数器、一个可编程中断控制器、一个步进电机控制器和一个简单的总线控制器组成，如图 5 所示。其中步进电机控制器通过发送脉冲控制步进电机的运动，ARM7TDMI 通过存储器映射 I/O 访问中断控制器、定时器、步进电机控制器，反过来中断控制器、定时器通过中断信号请求软件服务。除了硬件以外，还需要设计相应的软件，首先必须为中断控制器、定时器、步进电机控制器编写相应的驱动程序，

然后利用上述驱动程序编写工件加工程序，通过设定步进机在 3 个坐标方向的运动速度和行程，完成整个工件的加工过程控制。

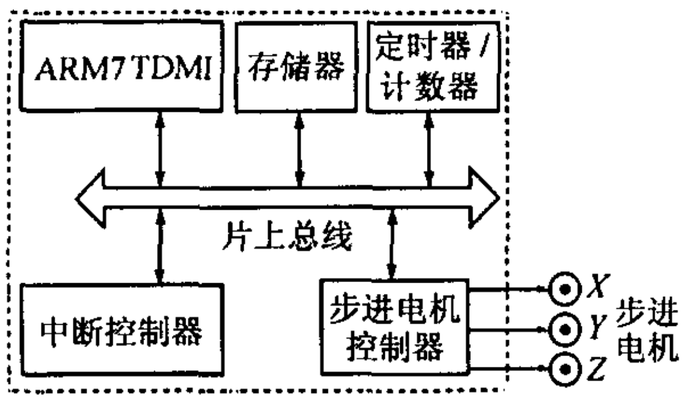

图 5　数控机床控制系统

作者首先采用硬件模拟器分别对定时器、中断控制器、步进电机控制器等进行了模拟验证。然后将整个系统集成起来进行了协同模拟验证，其中处理器核、存储器由指令集模拟器和总线功能模型模拟，系统其它部分由硬件模拟器模拟。首先作者对中断控制器、定时器/计数器、步进电机控制器及其驱动程序进行了验证，然后验证了数控程序能否按照加工要求控制步进电机控制器发出控制脉冲。通过协同模拟验证，发现问题，修改设计，然后再模拟，最终完成了满足要求的系统功能和软硬件接口设计。另外利用该协同验证系统，作者还进行其它若干基于 ARM7TDMI 处理器的 SOC 设计实例的协同验证，同样取得了令人满意的效果。实践证明本文讨论的方法完全能够满足单处理器 SOC 设计中系统功能验证的要求，并且通过采用文中讨论的几种同步优化方法，协同模拟速度得到大幅度提高。

7　结束语

设计验证是 SOC 设计的关键技术之一，贯穿整个 SOC 技术。介绍了一种基于指令集模拟器和硬件模拟器的协同模拟验证方法，该方法能够帮助设计者在设计早期发现系统功能、软硬件接口以及软硬件划分等方面的问题。本文设计流程各个阶段，本文主要研究其中的软硬件协同验证介绍的协同验证方法主要面向包含单处理器核的、结构相对简单的 SOC 设计，但许多情况下 SOC 会包含多个处理器核、支持 DMA 等，下一步作者将针对上述功能复杂的 SOC 协同验证方法进行研究。

参 考 文 献

[1] Lai Jinmei,Yao Qingdong. Software/hardware co-design for system on chip[C]. The Fourth International Workshop on CSCW in Design. Compiegne, France: 1999: 237 – 240.

[2] Rowson A. Hardware/software co-simulation[C]. The 31st Design Automation Conference. San Diego, CA, USA, 1994: 439 – 440.

[3] 吴清平, 刘明业. VHDL 编译型事件驱动模拟核心库.计算机研究与发展[J], 2002, 39(1): 17 – 22

[4] Zhu Jianwen, Gajski D. An ultra-fast instruction set simulator. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*[J], 2002, 10(3): 363 – 373.

[5] ARM Limited. ARM7TDMI Data Sheet. ARM DDI 0029E, http:// www. arm.com/documentation/, 2002.

[6] Liu Jie, Lajolo M, Sangiovanni-Vincentelli. Software timing analysis using HW/SW cosimulation and instruction set simulator [C]. The International Workshop on HW/SW Codesign. Seattle, WA, USA, 1998: 65 – 69.

严迎建：　男，1973 年生，讲师，博士生，研究方向为嵌入式系统软硬件协调设计、协同验证.

刘明业：　男，1934 年生，教授，博士生导师，长期从事 EDA 科研与教学工作.