

流量管理中基于事件的事务级建模方法

宋俊存, 王新安

(北京大学深圳研究生院 集成微系统科学与工程与应用重点实验室, 广东 深圳 518055)

摘 要: 提出了流量管理中基于事件的事务级建模方法, 包括通信接口函数封装, 流水线 and 调度结构建模方法. 同时实现了通用的事件延时队列, 方便对延时和吞吐率控制. 以流量管理中包调度为例, 对非基于事件和基于事件两种事务级模型的仿真性能进行比较. 基于事件的事务级建模大大提高了仿真速度, 缩短了产品开发周期.

关键词: 流量管理; 基于事件的事务级建模; 事件延时队列; 吞吐率

中图分类号: TP393

文献标识码: A

文章编号: 1000-7180(2010)12-0030-04

Event-based Transaction Level Modeling Methodology in Traffic Management

SONG Jun-cun, WANG Xin-an

(Key Lab of Integrated Microsystem Science & Engineering Applications,
Shenzhen Graduate school of Peking University, Shenzhen 518055, China)

Abstract: This paper presents the event based transaction level modeling for traffic management, including communication interface function encapsulation, the modeling approach of pipeline and scheduling structure. At the same time, the realization of payload event queue makes it convenient to control the latency and throughput. Using the non-event based and event based transaction level modeling approach realizes packet scheduling used in the traffic management and compare the simulation time. The simulation result shows that event based transaction level modeling accelerates the simulation. As a result, Event-based transaction level modeling reduce the development period.

Key words: traffic management; event-based transaction level modeling; payload event queue; throughput

1 引言

随着芯片集成度以摩尔定律增加, 功能越来越强大, 实现越来越复杂, 在系统设计前期进行建模仿真的需求越来越强烈. 在这种背景下, SystemC 应运而生, SystemC 是 C++ 的一个类库, 在软件的基础上加入了硬件特性来模拟硬件功能, 引入并发、通道、模块等概念, 方便建模人员采用适当的建模方法对硬件进行快速建模, 完成复杂系统的设计^[1-3]. 一般来说, 系统的抽象程度和精确度是相互矛盾的. 抽象程度越高, 建模、仿真速度越快, 但是精确度越低; 相反, 抽象程度越低, 精确度越高, 但是建模、仿真速度越慢. Beltrame 等提出了一种融合了多抽象层次的方法, 允许仿真过程中不同抽象层次的切

换, 让设计人员在仿真速度和精度之间折中^[4]. 面向结果的建模 (ROM)^[5] 是一种根据仿真过程的状态进行精度自适应的事务级建模方法, 该方法首先估计一个预期的时间, 等待预期的时间后, 根据状态更新预期值进行补偿.

随着网络速度的不断提高, 流量管理芯片 (Traffic Management, TM) 的性能越来越高, 设计难度不断提高, 实现越来越复杂, 在设计前期建模进行算法和架构探索的需求越来越强烈. 为了快速进行各种算法和架构的比较, 需要模型在保证一定精确度的同时具有高的仿真速度, 文献[6]提出了一种精度自适应的事务级建模方法, 正好适应了 TM 这一需求, 因此, 本文使用基于事件的方法对 TM 建模.

2 基于事件的 TM 事务级建模

在网络芯片数据传输过程中,模块间基本上点对点通信,这主要是由于数据传输速率很高,用总线传输数据很难达到所需的性能要求,因此,针对 TLM2.0 中的通信接口函数进行封装,使在网络数据传输中用起来更加方便.同时,根据 TM 特点,从建模角度看,模块内部功能处理可以分为流水线和调度两种结构,针对这两种结构,提出了基于事件的建模方法.

2.1 通信接口函数封装

采用 SystemC 建模,通信通过通道(channel)来进行抽象^[7],通信由模块对通道中函数的调用完成.为了使 SystemC 模型具有更好的兼容性和复用性,2008 年 6 月,TLM2.0 标准颁布,在 TLM2.0 中,针对 SoC 的特点,提出了一套标准的接口以及 Generic payload 事务类型以便模块能够更好的重用^[7].然而在 TM 芯片的数据传输中,模块间基本上点对点通信,Generic payload 的规范性所带来的好处在 TM 并不能得到体现,反而会由于需要管理可能的内存泄漏(实际传输的数据 char*)带来调试的麻烦.因此,通过把对 TLM2.0 的接口重新进行封装,使得传输的数据可以通过模版的方式来选择需要传递的事务.函数接口如下:

```
template<class ESLGenericPayload> class
FWTransportIf : public virtual tlm_fw_transport_if
<ESLBaseProtocolTypes<ESLGenericPayload>> {
public:
virtual tlm::tlm_sync_enum
nb_transport_fw(ESLGenericPayload& trans,
tlm::tlm_phase& phase, sc_core::sc_time& t)
= 0;
}
```

这样就解决了 TLM2.0 中由于传输数据指针可能带来的内存泄漏问题,同时也使接口保持和 TLM2.0 中的一致,不失通用性.

2.2 流水线结构建模

在 TM 芯片中,常会涉及到报文的检错、纠错、切割、重组等功能处理.由于有些功能比较复杂,在一个时钟周期内无法完成,因此设计为流水线结构,通过几拍来完成整个功能,满足性能设计要求,如图 1 所示.

为了提高仿真速度,流水线结构可以用两个参数来抽象:吞吐率和延时.吞吐率反映了模块处理事务的能力,延时反映了事务从开始处理到处理完成

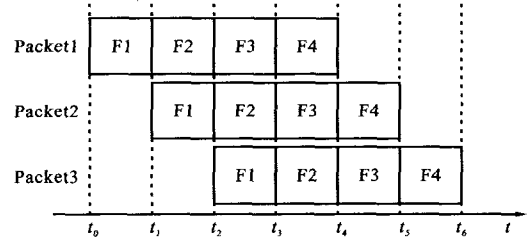


图1 流水线结构

所需的时间.流水线结构可通过如图 2 所示的方式进行抽象.

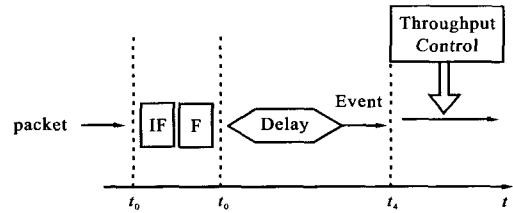


图2 流水线结构建模抽象

首先,通过接口 IF(第 2.1 节中封装的函数接口 nb_transport_fw)接收包;然后进行功能 F 处理,这都在同一时间完成,没有涉及线程;接着经过 4 拍延时之后,触发一个事件,进而触发线程把包发送出去.由于模块处理包的速率是一定的,因此需要进行吞吐率的控制.

由上述处理方式可以看出,只需要一个线程就可以完成功能 F 的处理,减少了线程切换所需的开销;同时,线程对事件敏感,整个处理过程是基于事件的,在没有包需要处理时,仿真时间直接往前推进,没有线程的空转,从而加快了模型的仿真速度.

为了方便对延时及吞吐率的控制,实现了一个通用的事件延时队列(Payload Event Queue, PEQ),事件延时队列工作原理是把事务放到延时队列中延时一段时间后,触发一个事件,通过线程对这个事件进行响应处理事务.

放入事件延时队列的函数接口如下:

```
void notify(const TransactionType& trans,
const sc_core::sc_time& t);
```

获取延时队列的事务的函数接口如下:

```
bool get_next_transaction(TransactionType& trans, const sc_core::sc_time& minInterval);
```

其中 minInterval 是下一事务与当前发送事务的最小时间间隔,用来控制吞吐率.

2.3 调度结构建模

在网络系统数据传输过程中,经常发生网络阻塞情况,为提高服务质量(QoS),需要对不同的业务

流数据包进行调度处理,这种情况不能用流水线结构的方式进行抽象建模,因为包与包之间存在相互依赖的关系,这类结构为调度结构。

调度结构建模同样分为吞吐率和延时两部分,处理过程如图 3 所示。

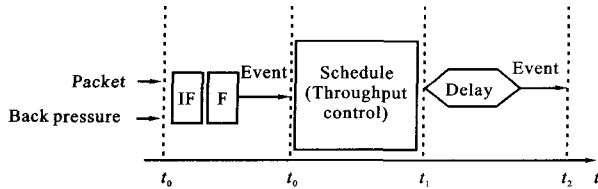


图 3 调度结构建模抽象

首先,通过接口 IF 接收进来的包和反压信息;然后进行包入队和反压信号处理,这都在同一时间完成,没有涉及任何线程,如果满足调度条件(队列中有包且没有反压),就触发一个事件;接着调度线程调度包,同时进行吞吐率控制,把调度出来的包放入延时队列进行延时处理;最后经过($t_2 - t_1$)延时之后,触发一个事件,进而触发线程把包发送出去。

调度线程执行过程如图 4 所示。

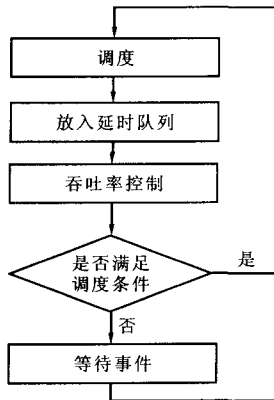


图 4 调度线程执行过程

在调度线程中,首先进行调度功能;然后把调度好的事务放入延时队列进行延时;接着进行吞吐率控制,线程等待吞吐率的倒数这段时间;最后根据队列状态判断是否满足调度功能处理的要求,如果不满足,则等待触发事件,否则直接进行下一轮调度处理。

由上述处理方式可以看出,需要两个线程来完成调度功能处理,减少了线程切换开销;同时,两个线程都对事件敏感,整个处理过程基于事件,在不满足调度要求时,仿真时间往前推进,没有线程的空转,因此加快了模型仿真速度。

3 TM SRR 调度处理建模

在网络数据传输过程中发生拥塞现象时,为了提高 QoS,需要对某些包进行丢弃和调度,调度算

法对于提高 QoS 具有重要的意义,本文对 SRR (Stratified Round Robin)^[8]算法使用非基于事件和基于事件的方法进行建模。TM SRR 调度处理模型如图 5 所示。

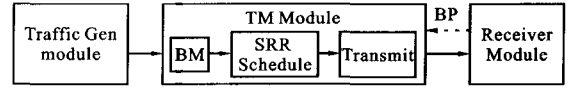


图 5 TM SRR 调度处理模型示意图

Traffic Gen 模块产生数据包,发送到 TM 模块进行尾部丢弃处理;入队后进行 SRR (Stratified Round Robin) 调度,然后经过一定的延时,发送数据包到 Receiver 模块;如果 Receiver 模块发生拥塞,数据包发送不出去,就会对 TM 模块产生反压信号 BP。

使用非基于事件事务级的方法建模如图 6 所示。

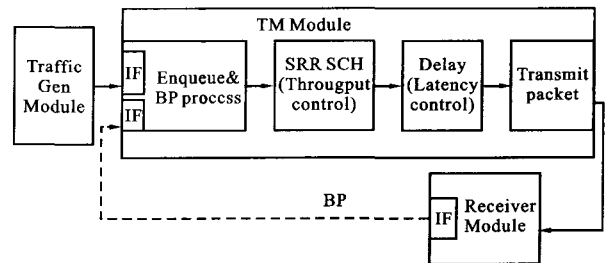


图 6 非基于事件的 TM SRR 调度处理建模

Traffic Gen 模块发送数据包给 TM 模块, TM 根据队列状态进行 BM 处理,决定是否进行包丢弃,经过 SRR 调度线程调度后,包延时线程进行延时操作,最后通过发送包线程把包发送出去。在非基于事件的事务级建模中,所有线程都对时钟敏感,即使没有事务处理,也要进行线程切换。

使用基于事件的事务级方法建模如图 7 所示。

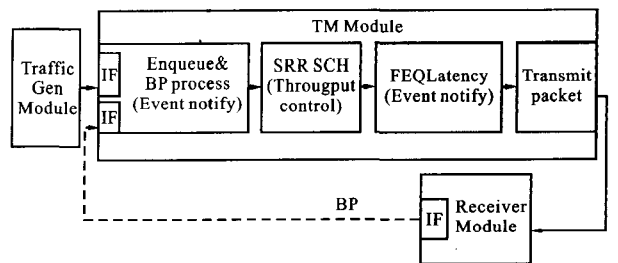


图 7 基于事件的 TM SRR 调度处理建模

TM 模块接收 Traffic Gen 模块发送的数据包以及 Receiver 模块的反压信号,进行尾部丢弃策略后判断是否满足 SRR 调度条件,当队列中有包,同时没有反压信号时,满足 SRR 调度条件,触发一个事件, SRR 调度线程响应这一事件进行包调度处理。

在 SRR 调度线程中,首先进行 SRR 调度处理;然后把调度出来的事务放入延时队列延时;接着进行吞吐率控制,线程进行等待,等待时间为吞吐率的倒数;最后根据内部状态判断是否需要继续进行 SRR 调度,如果不满足 SRR 调度要求,则等待触发事件,否则直接进行 SRR 调度。

在延时队列中进行延时后,触发一个事件,发送包线程响应这个事件把数据包发送出去,发送包线程对延时队列事件敏感,不会造成线程空转。

在基于事件的事务级建模中,模块线程都对事件敏感,如果没有事件,仿真时间直接往前推进,不会造成线程空转,造成不必要的线程切换开销。

4 仿真结果

为了评估仿真性能,我们在 Traffic Gen 模块发送包长分别为 64、791、1 518 Bytes 的包,进行三组数据的仿真结果对比,如图 8 所示。

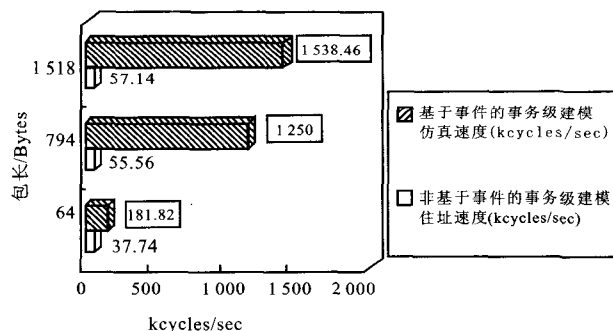


图 8 仿真对比结果

由图 8 仿真结果可以看出,基于事件的事务级建模相对于非基于事件事务级建模,仿真速度提高了大概一个数量级;在流量一定的情况下,包长越大,发送包的个数就越少,所需处理的事务也越少,基于事件的事务级建模仿真速度越快,这是因为在基于事件的事务级建模中线程对事件敏感,每次线程处理都是有效的处理,在没有事务需要处理时,仿真时间直接往前推进,不会造成线程空转,从而加快了模型的仿真速度,提高了仿真效率。

5 结束语

研究了基于事件的事务级建模方法,包括

通信接口函数的封装,流水线和调度结构的建模方法以及通用事件延时队列对延时和吞吐率的控制,实现了 TM SRR 调度处理的建模。相比于非事件的事务级建模方法,基于事件的事务级建模方法大大提高了模型的仿真速度,提高了开发效率。

参考文献:

- [1] Design Automation Standards committee of the IEEE Computer Society. IEEE Standard SystemC Language Reference. Manual[S/OL]. [2009-09-01]. <http://standards.ieee.org/getieee/1666/download/1666-2005.pdf>.
- [2] OSCI. The Open SystemC Initiative[EB/OL]. [2009-09-10]. <http://www.systemc.org>.
- [3] 陈琳,徐金甫,章轶,等. 基于 SystemC 的事务级建模研究[J]. 微电子学与计算机, 2006,23(11):213-216.
- [4] Giovanni Beltrame, Donatella Sciuto, Cristina Silvano. Multi-accuracy power and performance transactions-level modeling[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), 2007, 26(10): 1830-1842.
- [5] Schirner G, Domer R. Fast and accurate transaction level models using result oriented modeling[C]// IEEE/ACM. International Conference on Computer-Aided Design (ICCAD). San Jose, CA:IEEE/ACM, 2006(11): 363-368.
- [6] Rauf Salimi Khaligh, Martin. A latency, preemption and data transfer accurate adaptive transaction level model for efficient simulation of pipelined Buses[J]. IEEE, 2008,23(9):37-42.
- [7] OSCI. TLM-2.0 USER MANUAL[S/OL]. [2009-09-01]. <http://www.systemc.org/downloads/standards>.
- [8] Sriram Ramabhadran, Joseph Pasquale. The Stratified Round Robin Scheduler: Design, Analysis and Implementation[J]. IEEE/ACM Transactions on Networking, 2006, 14(6):1362-1373.

作者简介:

宋俊存 男,(1984-),硕士.研究方向为集成电路设计。

王新安 男,副教授,博士生导师。