基于 xMAS 模型的 SpaceWire 信誉逻辑的形式化验证

李艳春[1] 李晓娟[1] 关 永[1] 王 瑞[1] 张 杰[2] 魏洪兴[3]
(首都师范大学信息工程学院电子系统可靠性重点实验室 北京 100048)[1]
(北京化工大学信息科学与技术学院 北京 100029)[2]
(北京航空航天大学机械工程及自动化学院 北京 100191)[3]

摘 要 空间总线(SpaceWire)协议是应用于航空航天领域的高速通信总线协议,保证其可靠性至关重要。但是由于通信系统具有队列量、分布控制和并发性等特点,传统仿真模拟的验证方法存在不完备性的问题,采用模型检测方法对高层次属性进行验证时,通常会出现状态爆炸的问题。基于 xMAS 模型对 SpaceWire 通信系统中的信誉逻辑进行形式化建模、验证,xMAS 模型既保留了底层的结构信息,又可以验证高层次的属性。对通信系统中信誉逻辑进行抽象进而建立了 xMAS 模型,提取了可发送性、可接收性和数据一致性等 3 个关键属性,运用定理证明工具 ACL2 对关键属性的正确性进行了自动验证。该方法为验证指导下的系统设计提供了有效的参考。

关键词 xMAS 模型,信誉逻辑,SpaceWire,形式化验证,ACL2
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.2.026

xMAS-based Formal Verification of SpaceWire Credit Logic

LI Yan-chun[1] LI Xiao-juan[1] GUAN Yong[1] WANG Rui[1] ZHANG Jie[2] WEI Hong-xing[3]
(Beijing Key Laboratory of Electronic System Reliability Technology, Information Engineering College, Capital Normal University, Beijing 100048, China)[1]
(College of Information Science & Technology, Beijing University of Chemical Technology, Beijing 100029, China)[2]
(School of Mechanical Engineering and Automation, Beihang University, Beijing 100191, China)[3]

Abstract SpaceWire protocol is a high-speed communication bus protocol applied to aerospace, so it is very important for communication system to ensure the reliability of the design. Due to the presence of a large number of queues, distributed control and concurrency, the traditional verification methods have incomplete defects and state explosion when model checking occurs. This paper presented a formal verification method of credit logic in SpaceWire communication system with xMAS model. xMAS model retains the structural information in lower level and can verify high-level attributes. The paper built an abstract xMAS model for credit logic and listed three key properties including sending, receiving and data consistency. Correctness of the properties was verified automatically by the ACL2 theorem proving tool. It can provide effective reference for system design under the guidance of verification.

Keywords xMAS model, Credit logic, SpaceWire, Formal verification, ACL2

1 引言

SpaceWire 协议是欧空局(ESA)为应对复杂空间任务而提出的一种高速、全双工串行总线网络协议。它是基于两个商用标准 IEEE 1355-1995 和 IEEE 1596.3(LVDS),通过对 IEEE 1355 可靠性、功耗、抗辐射等方面的改进后,使其能更好地满足航空航天应用而提出的一种专门用于空间高速数据传输的总线标准[1]。目前已有 30 多个国家在人造卫星和航天飞船中把它应用到了有效载荷、高速、海量数据传输网络中。由于 SpaceWire 通信系统通常应用在航天等苛刻环境中,系统设计的微小错误都有可能产生巨大的经济损失甚至人员伤亡,因此,对 SpaceWire 通信系统进行正确性的验证具有重要的现实意义。

当今的计算设备例如笔记本、服务器、嵌入式系统等都整合了大量的处理单元和内存单元。SpaceWire 通信系统也是一种高可靠嵌入式系统。在复杂的通信系统中,通信架构的设计在系统的正确性与性能方面起到至关重要的作用,因此验证通信架构对整个系统来说都十分必要。由于消息传输的

到稿日期:2015-06-15 返修日期:2015-08-05 本文受国际科技合作计划(2011DFG13000,2010DFB10930),国家自然科学基金项目(61373034,61303014,61379019),北京市教委科研基地建设项目(TJSHG201310028014),北京市属高等学校创新团队建设与教师职业发展计划项目(IDHT20150507),北京市教委(KM201510028015)资助。

李艳春(1989—),男,硕士生,主要研究方向为形式化验证;李晓娟(1968—),女,博士,教授,主要研究方向为形式化验证、计算机网络,E-mail:lixj@mail.cnu.edu.cn(通信作者);关 永(1966—),男,博士,教授,博士生导师,主要研究方向为形式化验证、高可靠嵌入式系统;王 瑞(1981—),女,博士,讲师,主要研究方向为形式化建模和验证、高可靠嵌入式系统;张 杰(1967—),女,硕士,副教授,主要研究方向为形式化验证;魏洪兴(1974—),男,博士,副教授,主要研究方向为形式化验证、高可靠嵌入式系统。

并发性，传统的模拟仿真方法几乎无法完成。而模型检测方法可以解决这个难题，并且可以满足完备性的需要。但是由于通信系统越来越复杂，分布控制、流水线越来越深以及使用大量的队列、模型检测方法验证，导致模型的状态空间爆炸。为了解决这个问题，使用高层次的微架构模型来验证通信架构已经成为一种发展趋势。Intel 公司提出一种在微架构层面设计和形式化验证网络系统的方法——xMAS（eXecutable MicroArchitectural Specification），该方法可以获得设计者的意图[4]，并为验证指导下的系统设计提供有效的参考。该方法是一种基于数据流的形式化验证方法，以数据流的角度对通信系统进行抽象建模，既可以保留底层电路的结构信息，又可以验证高层次的属性。

目前，国外在使用一种已经定义好的功能元件库[4]对通信系统建模，在生成归纳的不变量[2]和检测所有可能存在的系统死锁[5]等方面取得了重大的进步。国内对 xMAS 的研究还比较少。xMAS 模型验证工具有很大的不足：Freek Verbeek 等人设计的一款软件 WickedXmas[12]，可以验证死锁属性和归纳不变量等式。它的不足之处在于受限于 xMAS 模型的规模太小和验证属性过于单一。文献[8]提出一种能够验证更复杂规模更大的 xMAS 模型的办法：在通用片上网络 GeNOC 的环境下，将 xMAS 模型引入其中，并且在 ACL2 中形式化。此部分工作仅将 xMAS 模型中的部分元件形式化，而且为了使 xMAS 模型适应 GeNOC 的环境，出现了一些冗余，不够简单，但是采用 ACL2 定理证明工具来形式化 xMAS 模型对本文的工作有很大的启发。

本文采用 xMAS 模型方法来验证由首都师范大学高可靠嵌入式系统实验室根据 SpaceWire 协议规范实现的电路设计。实验室用定理证明器 HOL4 成功验证了 SpaceWire 译码电路[14]，但证明过程需要人工引导，难度较高；另外，用 LTL 模型检测方法验证 SpaceWire 检错机制[15]，受限于模型规模，模型规模太大会导致状态空间爆炸。本文在上述工作的基础上，以信誉逻辑为核心，采用 xMAS 模型对 SpaceWire 通信系统中的信誉逻辑部分进行验证：首先，对信誉逻辑部分进行抽象建立 xMAS 模型；其次，在定理证明器 ACL2 中对 xMAS 模型形式化，创建 xMAS 模型库；最后，利用库中函数将建立的 xMAS 模型转换成 ACL2 的模型，提取了可发送性、可接收性和数据一致性 3 个关键属性，运用定理证明工具 ACL2 对关键属性的正确性进行自动验证。该工作不仅验证了信誉逻辑的功能，还可以为验证指导下的系统设计提供有效参考。

本文第 2 节对 ACL2 工具进行介绍；第 3 节对 xMAS 模型进行介绍；第 4 节介绍 xMAS 模型在 ACL2 中形式化；第 5 节对 SpaceWire 信誉逻辑进行形式化建模；第 6 节提取属性并进行验证；最后总结全文，给出未来工作的重点。

2 ACL2

应用 Common Lisp 计算逻辑（A Computational Logic for Applicative Common Lisp，ACL2）是德克萨斯大学发展的定理证明器，是由一个程序语言、一套一阶逻辑的可拓理论以及一个机械化的定理证明器所组成的软件系统。ACL2 从设计上支持基于归纳逻辑理论的自动推理，可应用于软件或硬件系统的验证，已用来形式化验证很多系统，比如浮点单元系统、高速仿真系统以及 Sun 公司的 Java 虚拟机系统等。

ACL2 通常包括公理和定理。公理分为两种，一种是 ACL2 原始函数；另一种是用户用宏 defun 定义的函数。用户通过 defthm 添加验证系统属性的定理。ACL2 定理证明器根据已知的公理，自动验证用户定义的定理的正确性。

本文用 ACL2 对 xMAS 模型进行形式化和属性的验证。需要用 ACL2 自定义函数和定理。

函数定义给出了函数名、参数名和函数体：

```
(defun fact (n)
  (declare (xargs:guard (natp n))
    (if (zp n)
    1
    (* n (fact (1-n)))))                    (1)
```

式(1)定义了一个函数，用来计算自然数的阶乘。此函数是一个递归函数，在可终止的条件下，才能被 ACL2 逻辑接受，(1)中的第二行代码的目的在于检测函数参数是否合理，以至于 fact 函数可以终止递归。

定理定义给出了定理名称和内容：

```
(defthm property-name
  (implies (natp n)
    (equal (fact n) (* n (fact (- n 1))))))    (2)
```

式(2)根据式(1)的定义给出了一个显而易见的定理，用于验证用户定义函数的正确性。

3 xMAS

xMAS（eXecutable MicroArchitectural Specification）模型是 Intel 公司最近提出的一种在微架构层面设计和形式化验证网络系统的方法，以数据流的角度对通信系统进行抽象建模，既可以保留底层电路的结构信息，又可以验证高层次的属性。通过信道按照一定的规则将例化后的 xMAS 元件连接起来构建网络。该模型由信道和元件组成。

3.1 信道

信道由两个布尔类型的控制信号（irdy 和 trdy）和一个数据信号（data）组成。在 xMAS 模型中，信道与两个元件连接。其中一个叫做起始端，它通过元件的输出端口将数据发送到信道中。另一个叫做终端，它通过元件的输入端口从信道中接收数据。irdy 为真表示起始端准备好发送数据，trdy 为真表示终端准备好接收数据。起始端决定 irdy 和 data 信号，终端决定 trdy 信号。当信道的信号 irdy 和 trdy 同时为真时，数据从起始端通过信道传输到终端。

3.2 元件

xMAS 元件库由 queue、function、source、sink、fork、join、switch 和 merge 8 个元件组成，如图 1 所示。

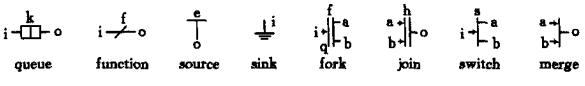

图 1 xMAS 元件

- queue：一种能够储存数据的元件。
- function：一种用于数据转换的元件。
- source：一种能够非确定性地发送数据的元件。
- sink：一种能够非确定性地接收数据的元件。
- fork：一种将一路输入同时分叉成两路输出的元件。
- join：一种将两路输入联结成一路输出的元件。

• switch:一种为数据包选择传输路径的元件。

• merge:一种起到仲裁器作用的元件。

上述每个元件都决定着输入信道的 trdy 和输出信道的 irdy、data。每个元件的功能都被同步等式来形式化地表达出来[2]。

如图 2 所示,该 xMAS 模型由 3 条信道将一个 source、两个能够储存 k 个数据包的 queue 和一个 sink 顺序连接而成。source 可以非确定性地发送数据包,该数据包经过两个队列,最终被 sink 接收。这个例子虽然很简单,但是可以让我们对 xMAS 模型有个感官认识。

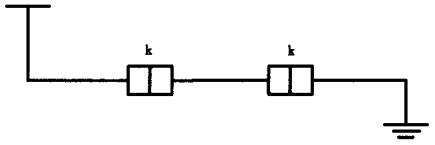

图 2 简单的 xMAS 模型

4 xMAS 模型在 ACL2 中形式化

本文的主要工作是用 xMAS 模型对 SpaceWire 系统进行抽象建模,进而提取并验证关键的属性。然而目前没有一个工具能够很好地完成对 xMAS 模型属性直接验证的工作,为此本文采取的解决办法是在 ACL2 中建立 xMAS 模型库,进而可以间接地验证 xMAS 模型。xMAS 模型在 ACL2 中形式化正是连接 xMAS 和 ACL2 的纽带,可以将 xMAS 模型转换成 ACL2 可以识别的模型。因此这部分工作是后面验证工作的基础,也是本课题需要突破的重点内容。

此部分是在 ACL2 中建立 xMAS 模型库,主要包括 xMAS 元件的形式化和 xMAS 网络的形式化两部分内容,目的在于可以将 xMAS 模型转换成 ACL2 模型。

4.1 xMAS 元件的形式化及其关键属性的验证

xMAS 元件的形式化的理论基础是 xMAS 元件的同步等式,该等式对元件的输入与输出有着严格的规定。下面以 function 元件为例,详细介绍 function 元件的形式化。

首先了解一下 xMAS 对 function 元件的同步等式:

o. irdy := i. irdy o. data := f (i. data) i. trdy := o. trdy (3)

式(3)表达出了 function 元件需要根据前一个周期中输入输出信道的信号值计算出下一个周期输入信道的 trdy(即 i. trdy)和输出信道的 irdy 和 data(即 o. irdy 和 o. data)。

因此,可以很容易用 ACL2 形式化表达出 function 元件:

```
(defun fun (in out f-data)
(list (list (car in) (nth 1 out) (nth 2 in)) (list (car in) (nth 1 out) f-data))
)
```

在 ACL2 中用 fun 函数来表达 function 元件,in,out,f-data是函数的输入参数,in,out 分别代表元件的输入信道和输出信道,它们都是形如(irdy trdy data)的列表。(car in)表达的是输入信道的第零项的值 irdy,即 in. irdy。同理(nth 1 in)表达的是 in. trdy,(nth 2 in)表达的是 in. data。

同理,其它 7 个元件也可用 ACL2 形式化表达出来,本文不再赘述。

从 function 元件的同步等式可以看出,输入信道的(trdy and irdy)与输出信道的(irdy and trdy)同真同假。从 xMAS 模型角度分析,输入信道与输出信道的连通情况是同步的。

通过刚才定义出的 function 元件,我们可以用定理验证提出的属性。

属性 1 当输入信道连通时,输出信道必连通。

```
(defthm synchronization-in-out-function-1
(implies (trdy-and-irdy (car (fun in out f-data)))
(trdy-and-irdy (nth 1 (fun in out f-data)))))
```

属性 2 当输出信道不连通时,输入信道必连通。

```
(defthm synchronization-in-out-function-2
(implies (trdy-and-irdy (nth 1 (fun in out f-data)))
(trdy-and-irdy (car (fun in out f-data)))))
```

通过 ACL2 定理证明器自动验证表明,属性 1 和属性 2 是正确的。这样的同步属性与队列数据包的传输有着密切的关系,后文会详细说明。

4.2 xMAS 网络的形式化及其网络的形式建模

4.2.1 xMAS 网络的表达

为了将 xMAS 模型转化为 ACL2 模型,有必要对 xMAS 网络进行下列定义。

定义 1 xMAS 网络 xmasnetwork 由 3 部分组成,即 xmasnetwork=〈components channels sequential-networks〉,其中 components 是 xMAS 网络中所有元件的集合;channels 是 xMAS 网络中所有信道的集合;sequential-networks 负责所有队列的数据包更新。

定义 2 xMAS 元件 component=〈id type ins ous param〉,其中 id 用来标识不同的元件;type 表示元件的类型,type 只能在 queue、function、source、sink、fork、join、switch 和 merge 中取值;ins 表示元件的所有输入信道的集合,输入信道的条数由元件的类型决定;ous 表示元件的所有输出信道集合,输出信道的条数也是由元件的类型决定。param 表示元件本身的参数。

定义 3 xMAS 信道 channel=〈id init target param data〉,其中 id 标识不同的信道;init 表示该信道起始端元件的 id;target 表示信道终端元件的 id;param 表示与信道连接的两端元件的参数;data 表示信道的值。

定义 4 xMAS 时序网络 sequential-network=〈init target〉,init 表示时序元件(即 source 和 queue)组成的集合,当满足一定条件时,集合中的所有元件会同时进行数据输出操作。Target 表示时序元件(即 sink 和 queue)组成的集合,当满足一定条件时,集合中的所有元件会同时进行数据输入操作。

根据上述的 4 个定义,我们可以将基于图形语言的 xMAS 模型转换成 ACL2 中能够识别的 xMAS 网络 xmasnetwork,它表达出了 xMAS 模型的整体结构。

为了 xMAS 网络的正确性,xmasnetwork 必须满足下列约束:

• components、channels 和 sequential-networks 中的每一项都不能重复。

• components 中所有元件的输入输出端口数量的总和是 channels 中所有信道数量总和的两倍。

• channels 中的每条信道都是与该信道相连的终端元件的输入信道。

• channels 中的每条信道都是与该信道相连的起始端元

件的输出信道。

• components 中的每个元件都是该元件输入信道的终端元件。

• components 中的每个元件都是该元件输出信道的起始端元件。

上述约束对一个 xMAS 网络来说，是检测 xmasnetwork 正确性的重要指标。

在 ACL2 中，用 xmasnetworkp 函数表达：

```
(defun xmasnetworkp (ntk)
(and (no-duplicatesp-equal (xmasnetwork-channels ntk))
  (no-duplicatesp-equal (xmasnetwork-components ntk))
  (no-duplicatesp-equal (xmasnetwork-sequential-networks ntk))
  (all-components-port-number=all-channels-number * 2 ntk)
  (every-channel-is-its-target-ins ntk)
  (every-channel-is-its-init-ous ntk)
  (every-component-is-its-ins-target ntk)
  (every-component-is-its-ous-init ntk)
  ))
```

那么，对于每个具体 xMAS 网络 ntk 来说，都要满足网络正确性定理：

```
(defthm xmasnetwork-correctness
(xmasnetworkp ntk))
```

4.2.2 xMAS 网络的形式建模

xMAS 网络是一个可执行的网络，包括组合逻辑部分和时序逻辑部分。组合逻辑部分是由 function、fork、join、switch 和 merge 5 个组合逻辑元件按照一定的顺序连接而成的网络，负责更新信道的信号值。时序逻辑部分是由 source、sink 和 queue 3 个时序元件按照一定的顺序组合而成的网络，主要负责更新所有队列的数据。

因此，xMAS 网络的运转主要包括信道控制信号值的计算和队列中数据的更新。

在 ACL2 中，用 channel-calculate 函数完成信道控制值的计算任务，用 data-transfer 函数完成所有队列数据的更新任务。

channel-calculate 主要算法如下：

```
(defun channel-calculate(channels components)
(let ((init-channel (init (car channels) components));;通过信道起始端元件计算出信道下一个网络状态的 irdy 和 data。
(target-channel (target (car channels) components)));;通过信道的终端元件计算出信道下一个网络状态的 trdy。
(if (endp channels);;判断所有信道是否都计算完。
nil
(append (set-channel init-channel target-channel)
(channel-calculate (cdr channels) components)))));;重新计算出每个信道下一个网络状态的信道值。
```

注：set-channel 函数把 init-channel 和 target-channel 合并在一起，计算出信道的所有信号值。init 和 target 函数调用了 xMAS 元件的基本函数来计算信道的信号值，这也是 xMAS 元件与 xMAS 网络之间有联系的地方。

在介绍 data-transfer 算法之前，有必要说明 xMAS 网络时序逻辑部分进行数据更新的条件：sequential-network = ⟨init target⟩，当 init 和 target 之间的组合逻辑部分信道连通时，init 数据输出，target 数据输入。

组合逻辑部分是由 function、fork、join、switch 和 merge 等 5 个时序元件按照一定的顺序连接而成的网络。由于已经证明这些元件存在这样的元件信道连通同步性质：输入连通，输出必连通；输出连通，输入也必连通。那么要保证整个组合逻辑网络连通，只需保证网络的所有输入和输出信道连通即可。因为如果网络的输入和输出端信道连通，通过元件信道连通同步性质，可以推导出整个网络都是连通的。

下面阐述 data-transer 的主要算法：

```
(defun data-transfer(sequential-networks channels components)
(cond ((endp sequential-networks) channels);计算结束后，返回信道集合。
  ((connect (car sequential-networks))
(data-transfer (cdr sequential-networks)(update-channels
(car sequential-networks) channels components) components));当某个组合逻辑网络连通时，update-channels 函数会进行队列中数据的输入和输出操作。
(t (data-transfer (cdr seqnential-networks) channels components))));;当不满足条件时，递归计算下一个，直到计算完成为止。
```

在 channel-calculate 和 data-transfer 的基础上，可以在 xMAS 网络上验证 xMAS 模型的属性。

5 SpaceWire 信誉逻辑的形式化建模

SpaceWire 总线由波特率选择模块、恢复模块、信誉模块、时间模块、检错模块、控制模块、发送模块和接收模块 8 个模块组成。其接口设计框图如图 3 所示。

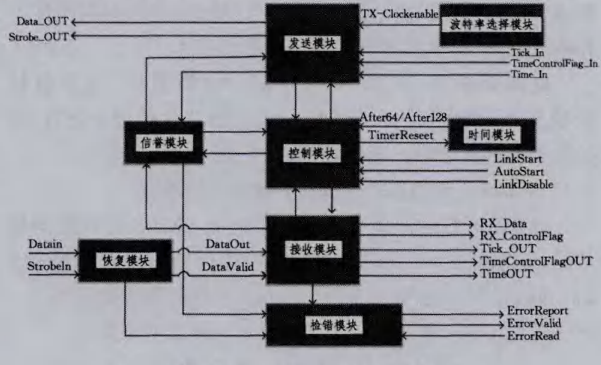

图 3 SpaceWire 接口设计框图

本文对信誉逻辑进行验证，验证其 VHDL 程序与规范描述是否一致。信誉模块的主要功能是控制接收器接收到的常字符(NChar)的个数。按照该苛刻环境高速总线协议中的要求，接收器的缓存最多只能接收 56 个常字符，当主机发送器发送一个 FCT 控制符，说明其相应的接收器还可以再接收 8 个常字符。因此，在发送器没有发送完任何 8 个常字符的情况下，此端接收到的 FCT 的个数不能超过 7 个，否则将会发生信誉错误。根据信誉模块的情况可以验证的属性有以下 3 个。

可发送性：信誉模块根据当前的 FCT 的数量，控制发送器是否可以发送数据包。当信誉标志 FCT 数量不为零时，发送器可以发送数据包。

可接收性：接收器每消耗一个数据包，信誉模块中 FCT 数量就增加一个。信誉模块根据当前 FCT 的数量，通知接收器是否还有空间接收 FCT。

数据一致性：发送器发送的数据包与接收器接收到的数据包数据一致。

由图 3 可以看到信誉逻辑不是单独存在的，它与发送模块和接收模块关系非常密切。因此，必须将三者统一起来，才会形成一个相对完整的系统。发送器负责发送数据包和控制字符 FCT。当本方接收器收到信誉模块的许可时，说明对方接收器有空间接收数据包，这时本方发送器才可以发送数据包。当本方接收器有空间接收数据时，会让本方的发送器发送相应数量的 FCT。接收器负责接收数据包和控制字符 FCT。当本方接收器有存储空间时，就可以接收对方发送的数据包和 FCT。接收器每消耗一个数据包，就多一个存储空间，就会让本方发送器发送一个 FCT。信誉模块主要控制接收器接收数据包的个数，控制接收器接收 FCT 的个数，控制发送器发送数据包。当 FCT 的数量达到最大时，再收到 FCT 就会发生信誉错误。

信誉逻辑是一种在分布式系统中流控制和资源分配的常见设计形式。图 4 中 chan_0～chan_21 代表 id 为 0～21 的信道，com_0～com_19 代表 id 为 0～19 的元件，这样标记的目的是为了后面验证属性时方便描述，增强可读性。在这个系统中，com_7 和 com_17 分别代表左右两端的接收器的存储信息队列；com_3 和 com_13 分别代表左右两端信誉模块中储存信誉的队列，此队列中的信誉是要发送给另一端的；com_8 和 com_18 分别代表左右两端接收信誉的队列。起始条件下，com_7 和 com_17 队列为空，表示双方接收器的存储空间为空，com_3 和 com_13 队列为 k（k 为所有队列的容量），根据文献[2]中流的不变量理论，可以推导出以下不变量等式：num(com_3)＋num(com_18)＋num(com_7)＝k；num(com_13)＋num(com_8)＋num(com_17)＝k；从等式中很容易看出，如果 com_8 不为空，那么 com_17 队列一定不满。也就是说如果左端接收到右端发送的信誉的话，右端必然会有存储空间接收左端发送的信息，可以避免链路接收端的输入缓冲器溢出。

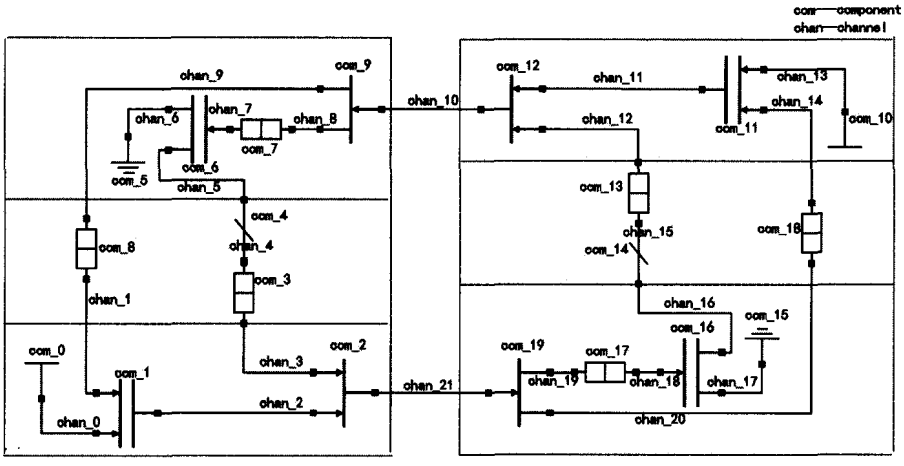

图 4　基于信誉逻辑的链路两端通信的 xMAS 模型

6　验证和分析

本文实验环境：硬件是因特尔 i5-3450 处理器；软件是 Windows 7 专业版，定理证明工具 ACL2 3.6。针对信誉逻辑提取的 3 个关键属性，采用 ACL2 定理证明器对信誉逻辑的功能正确性进行验证。

属性 3(可发送性)　通信系统发送器接收信誉标志 FCT 时，可以发送数据包。ACL2 实现如下：

```
(defthm send-property
(let * ((channels (xmasnetwork-channels * ntk * )
  (components (xmasnetwork-components * ntk * )
  (sequential-networks (xmasnetwork-sequential-networks * ntk *
)))
(implies (and (xmasnetworkp * ntk * )
    (get-source-oracle 0 components)
    (not (endp (get-queue-data 8 channels))))
    (get-channel-irdy 2 (run-netwrok-n channels components
sequential-networks 2)))))
```

在 ACL2 中，* ntk * 表示 xMAS 网络的结构。其中函数 run-network-n 表示计算 n 次之后网络状态的值，此函数主要调用了 channel-calculate 函数和 data-transfer 函数来实现，把当前网络状态值作为输入参数，输出计算 n 次之后的网络状态值。函数 get-source-oracle 表示获得 source 的 oracle 值，通过该值可以判断出 source 是否发送数据。函数 get-queue-data 表示获得整个队列中的数据。

属性 4(可接收性)　当每消耗一个数据包时，接收器就多一个数据空间，发送器就会发送一个 FCT。ACL2 实现如下：

```
(defthm receive-property
(let * ((channels (xmasnetwork-channels * ntk * )
  (components (xmasnetwork-components * ntk * )
(sequential-networks (xmasnetwork-sequential-networks * ntk * ))
(queue7-num (len (get-queue-data 7 channels)))
(queue3-num (len (get-queu-data 3 channels))))
(implies (and (xmasnetworkp * ntk * )
   (get-sink-oracle 5 components)
    (not (endp (get-queue-data 7 channels)))
    (not (equal queue3-num (get-queue-length 3 components))))
(and (data-equal 'FCT (first (get-queue-data 3 (run-network-n chan-
nels components sequential-networks 3))))
  (data-equal (1-queue7-len) (len (get-queue-data 7(run-network-n
channels components sequential-networks 3))))))))
```

属性 5(数据一致性)　发送器发送的数据包与接收器接收到的数据包数据一致。ACL2 实现如下：

```
(defthm data-consistency-property
(let * ((channels (xmasnetwork-channels * ntk * )
  (components (xmasnetwork-components * ntk * )
```

（下转第 134 页）

[9] Jafar S A, Fakhereddin M J. Degrees of freedom for the MIMO interference channel [J]. IEEE Transactions on Information Theory, 2007, 53(7): 2637-2642

[10] Lee N, Shin W, Heath R W, et al. Interference alignment with limited feedback for two-cell interfering MIMO-MAC[C]// 2012 International Symposium on Wireless Communication Systems (ISWCS). IEEE, 2012: 566-570

[11] Kim J, Park S H, Sung H, et al. Sum rate analysis of two-cell MIMO broadcast channels: spatial multiplexing gain[C]// IEEE International Conference on Communications (ICC 2010). IEEE, 2010: 1-5

[12] Shin W, Lee N, Lim J B, et al. On the design of interference alignment scheme for two-cell MIMO interfering broadcast channels[J]. IEEE Transactions on Wireless Communications, 2011, 10(2): 437-442

[13] Jafar S A, Shamai S. Degrees of freedom region of the MIMO X channel[J]. IEEE Transactions on Information Theory, 2008, 54(1): 151-170

[14] Huang C, Jafar S A. Degrees of freedom of the MIMO interference channel with cooperation and cognition[J]. IEEE Transactions on Information Theory, 2009, 55(9): 4211-4220

[15] Yu W, Cioffi J M. Sum capacity of Gaussian vector broadcast channels[J]. IEEE Transactions on Information Theory, 2004, 50(9): 1875-1892

(上接第 117 页)

```
(sequential-networks (xmasnetwork-sequential-networks * ntk * ))
(data (get-source-data 0 components))
(queue17-num (len (get-queue-data 17 channels)))
(queue13-num (len (get-queue-data 13 channels))))
(implies (and (xmasnetworkp * ntk * )
    (get-source-oracle 0 components)
    (get-sink-oracle 15 components)
    (not (endp (get-queue-data 8 channels)))
    (not (equal queue13-num (get-queue-length 13 components)))
    (not (equal queue17-num (get-queue-length 17 components))))
  (data-equal data (get-sink-data 15 (run-network-n channels components sequential-networks 6))
```

结束语 本文的工作是基于 xMAS 模型的 Spacewire 信誉逻辑的形式验证。对通信系统中信誉逻辑进行抽象建立 xMAS 模型，提取并成功验证了可发送性、可接收性和数据一致性等 3 个关键属性，证明其设计是正确的，为验证指导下的系统设计提供有效参考。另外本文用 ACL2 对 xMAS 模型进行形式化，建立的 xMAS 模型库可以重复利用，以提高模型验证工作的效率。

现有工作仅将信誉模块、发送模块和接收模块组合起来进行验证，在以后的工作中要对 SpaceWire 系统所有的模块建立规模更大的 xMAS 模型，提取更高层的属性，做到对 SpaceWire 规范的完备性验证，避免 SpaceWire 设计在未来的应用中可能造成的严重后果。此外在不影响原有 xMAS 元件的基础上，根据实际需要还可以对 xMAS 元件进行必要的扩展，让 xMAS 模型更加完善。

参 考 文 献

[1] ECSS. Space Engineering: SpaceWire-Links, nodes, routers, and networks(ECSS-E-50-12A)[S]. The Netherlands: ESA Publications Division ESTEC, 2003

[2] Chatterjee S, Kishinevsky M. Automatic Generation of Inductive Invariants from High-Level Microarchitectu-ral Models of Communication Fabrics [J]. Formal Methods in System Design, 2012, 40(2): 147-169

[3] Verbeek F, Schmaltz J. Hunting deadlocks efficiently in microarchitectural models of communication fabrics[C]// Formal Methods in Computer-Aided Design(FMCAD). 2011: 223-231

[4] Chatterjee S, Kishinevsky M O. Quick formal modeling of communication fabrics to enable verification[C]// High Level Design Validation and TestWorkshop (HLDVT). IEEE, 2010: 42-49

[5] Gotmanov A, Chatterjee S, Kishinevsky M. Verifying deadlock-freedom of communication fabrics [C] // Verification, Model Checking, and Interpretation (VMCAI). 2011: 214-231

[6] Ray S, Brayton R K. Scalable Progress Verification in Credit-Based Flow-Control Systems[C]// Design, Automation & Test in Europe Conference & Exhibition(DATE). 2012: 905-910

[7] Borrione D, Helmy A, Pierre L, et al. A generic model for formally verifying NoC communication architectures: a case study [C]// Networks-on-Chip (NOCS). 2007: 127-136

[8] Verbeek F. Formal Verification of on-Chip Communication Fabrics[M]. Radboud University Nijmegen, 2013

[9] van Gastel B, Schmaltz J. A formalisation of XMAS[C]// Electronic Proceeding in Theoretical Computer Science (EPTCS). 2013: 111-126

[10] Kaufmann M, Moore J S, Manolios P. Computer-Aided Reasoning: An Approach[M]. Norwell, MA: Kluwer Academic Publishers, 2000

[11] Kaufmann M, Moore J S, Manolios P. Computer-Aided Reasoning: ACL2 Case Studies[M]. Norwell, MA: Kluwer Academic Publishers, 2000

[12] Joosten S J C, Verbeek F, Schmaltz J. WickedXmas: Designing and Verifying on-chip Communication Fabrics[C]// Design and Implementation of Formal Tools and Systems(DIFTS). 2014: 1-8

[13] Kishinevsky M, Gotmanov A, Viktorov Y. Challenges in Verifying Communication Fabrics[C]// Interactive theorem proving (ITP). 2011: 18-21

[14] Zhang Yu-peng, Shi Zhi-ping, Guan Yong, et al. Formal Verification of SpaceWire Decoding Circuit in HOL4[J]. Journal of Chinese Systems, 2013, 8: 1959-1963(in Chinese)
张玉鹏，施智平，关永，等. SpaceWire 译码电路在 HOL4 中的形式化[J]. 小型微型计算机系统，2013，8：1959-1963

[15] Dong Ling-ling, Guan Yong, Li Xiao-juan, et al. Verification for SpaceWire error detection mechanism by LTL model checking [J]. Computer Engineering and Applications, 2012, 48(22): 88-94(in Chinese)
董玲玲，关永，李晓娟，等. 用 LTL 模型检验的方法验证 SpaceWire 检错机制[J]. 计算机工程与应用，2012，48(22)：88-94