文章编号: 1001 - 9081(2011) 10 - 2630 - 04 doi: 10. 3724/SP. J. 1087. 2011. 02630

基于拥塞控制的片上网络多播路由算法

袁景凌,刘 华,谢 威,蒋 幸
(武汉理工大学 计算机科学与技术学院,武汉 430070)
(yuanjingling@ 126. com)

摘 要: 为了满足片上网络日益丰富的应用要求,多播路由机制被应用到片上网络,以弥补传统单播通信方式的不足。以 Mesh 和 Torus 类的片上网络为例,分析了基于路径的 3 种多播路由算法(即 XY 路由、UpDown 路由和 SubPartition 路由算法) ,并研究了相应的拥塞控制策略。通过模拟实验表明,多播较单播通信具有更小的平均传输延时和更高的网络吞吐量,且负载分配均匀; 特别是 SubPartition 路由算法随着规模增大效果更加明显; 提出的多播拥塞控制机制,能更有效地利用多播通信,提高片上网络的性能。

关键词: 片上网络; 多播机制; 拥塞控制; 路由算法
中图分类号: TP393. 03 **文献标志码:** A

Multicast routing algorithm based on congestion control for NoC

YUAN Jing-ling, LIU Hua, XIE Wei, JIANG Xing
(*School of Computer Science and Technology*, *Wuhan University of Technology*, *Wuhan Hubei* 430070, *China*)

Abstract: The multicast routing method has been applied into the Network on Chip (NoC) since traditional unicast communication cannot meet the increasingly rich application requirements of NoC. Three kinds of path-based multicast routing algorithms including XY routing, UpDown routing and SubPartition routing algorithms were applied to 2D Mesh or Torus NoC. The congestion control strategy was proposed. The simulation results show multicast routing algorithms have shorter average latency and higher throughput and balanced applied load compared with unicast routing algorithms. SubPartition routing algorithm was confirmed to have a more stable and better performance as the network size increases. Finally, multicast congestion control techniques for NoC were employed to make multicast communications more efficient and enhance the NoC performance.

Key words: Network on Chip (NoC); multicast mechanism; congestion control; routing algorithm

0 引言

随着片上系统(System on Chip, SoC) 技术的发展,单个芯片集成度的增大增加了全局同步设计的难度,于是出现了片上网络(Network on Chip, NoC) 的概念。NoC 是 SoC 的一种新的设计方法,其核心思想是把网络设计的思想移植到芯片设计中,NoC 带来了一种全新的片上通信方式,代替了传统的总线通信方式,显著改善了片上系统的性能。尽管片上网络允许多个并发事务,但是它不直接支持多播,而很多片上系统的应用需要多播支持,比如传递全局状态,管理和配置网络,执行缓存一致性协议,特别是具有实时限制、吞吐量为导向的嵌入式多媒体处理中需要一个有效执行多播的方法[1]。在分布式共享内存的模式中,多播能有效地支持共享数据的更新[2],有效地执行几个并行应用,如搜索算法、矩阵运算(转置、相乘) 等也需要多播的支持[3]。NoC 上常规的单播通信方式已经不能满足应用需求,而多播通信方式能缓解资源负担,能更有效地利用带宽。当网络中需要传送多播消息时,传统的单播通信方式会将多播分解为多个单播,网络中会同时出现多个不必要的消息,这将迅速降低网络的性能,严重时还会引起网络拥塞。因此,研究 NoC 的多播通信方式对改善网络性能具有重要的意义。

1 相关研究

由于多播通信具有多个目的地,网络更有可能产生拥塞,因此需要选择合适的方法进行拥塞控制,来提高片上网络的性能。当节点产生的流量大于整个网络负载时,拥塞就会产生,拥塞是网络性能下降的一个重要因素。在虫孔交换片上网络中,传递的信息单元为片,并由头片来做路由决策。当头片被阻塞时,定时器开始计时,一定时间后在网络中产生并扩散限制信号,限制产生新的数据包注入网络[4]。文献[5]在基于动态 XY 路由的基础上提出了一种新的适应性路由方法,这是通过在两个核之间的 X 方向和 Y 方向增加两根拥塞控制线实现的,这种方法在一定程度上能够控制拥塞,但是增加了连接线,因此增加了核之间连接的复杂性,同时增加了片上网络的面积。文献[6]采用全局反馈来控制拥塞,这种方法考虑了全局的流量。通过反馈相邻节点之间的拥塞信息来反映全局的流量,这种拥塞信息逐步反馈到源节点,因此源节点能够调整数据注入速率。这种方法在每个节点都增加一个反馈控制器和缓存,增加了硬件成本。文献[7]采用了新的检测标准来检测拥塞,即被占用的源缓存区长度,拥塞信息能

收稿日期: 2011 - 03 - 01; 修回日期: 2011 - 09 - 14。
基金项目: 武汉理工大学自主创新研究基金资助项目(2010-ZY-JS-026; 2011-IV-086) 。
作者简介: 袁景凌(1975 -) ,女,湖北武汉人,副教授,博士,CCF 会员,主要研究方向: 智能方法、多核分析; 刘华(1985 -) ,男,湖北恩施人,硕士研究生,主要研究方向: 片上网络; 谢威(1982 -) ,男,湖北襄阳人,硕士研究生,主要研究方向: 云计算; 蒋幸(1987 -) ,男,湖北天门人,主要研究方向: 体系结构。

够直接在节点内获得,因此这些方法是分布式的且不需要全局信息。这种方法用很小的硬件成本实现了拥塞控制,但逻辑控制稍显复杂。文献[8]提出了根据空间分布特征预测最有可能出现拥塞区域的全局拥塞控制算法,该全局拥塞控制算法考虑了重叠的程度和最大空闲空间两个优化标准。直观地说,一个良好的拥塞控制算法应尽量减少重叠的程度,同时最大化连续的空白空间。片上网络多播通信能显著提高其通信效率,但没有涉及到拥塞控制问题[9]。本文根据片上网络的特点设计了多播拥塞控制算法来解决片上网络的拥塞问题,更好地提高片上网络的通信性能。

2　基于路径的多播路由算法

2.1　片上网络多播路由模型

定义 1　具有 2D mesh 拓扑结构的 $m \times n$ NoC 模型可以表示为图 $G = (V,E)$,V 是图 G 的节点集,$V(G) = \{(x,y) \mid 0 \leqslant x < n, 0 \leqslant y < m\}$,$V$ 中的每个节点代表 NoC 中的一个路由器;E 是图的边集,$E(G) = \{[(x_i,y_i),(x_j,y_j)] \mid (x_i,y_i),(x_j,y_j) \in V(G)\}$,$E$ 中每条边对应路由器之间的一条通道,$N = m \times n$ 为 NoC 的节点数。

定义 2　多播模型可以表示为 $M = \{s,d_1,d_2,d_3,\cdots,d_k\}$,其中,$s$ 代表源点集;$\{d_1,d_2,d_3,\cdots,d_k\}$ 代表目的集;k 为目的地的个数,如果 $k = 1$ 则多播模型转化为了单播,若 $k = N$ 则多播模型转化为了广播。

定义 3　令多播的源节点为 U,目的集为 V,则多播通信为寻找合适的路由算法 R 使得 $R(U,V) = W$,其中 W 为 U 到 V 路径上的中间节点。

2.2　基于路径的路由算法

本文讨论 3 种多播路由算法,分别是 Original、UpDown、SubPartition。2D mesh 拓扑结构中 Original 算法允许数据包往链路的各个方向转发; UpDown 算法根据源节点的标号,将目标节点划分为两个集合,一个集合比源节点的标号大,而另一个集合则比源节点的标号小,UpDown 算法只允许数据包往两个集合其中的一个转发; SubPartition 则根据一定的划分规则将网络划分为 4 个子网,在各个子网上执行各自的路由算法,3 种算法数据包转发方向如图 1 所示。

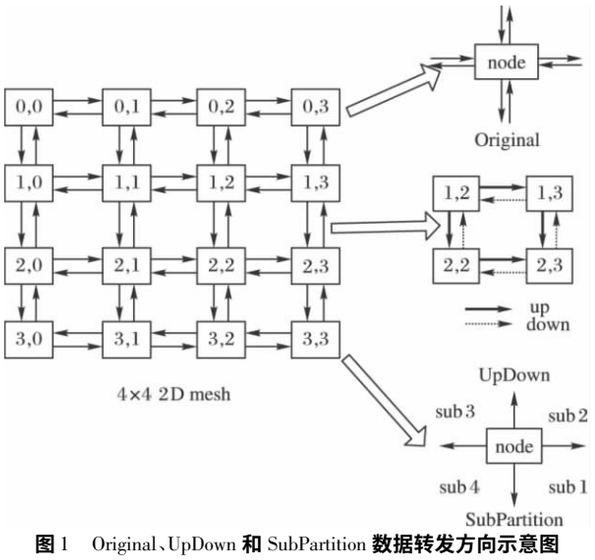

图 1　Original、UpDown 和 SubPartition 数据转发方向示意图

2.2.1　原始 XY 路由算法

在该算法下,数据包可以沿着链路的各个方向转发,多播数据包采用复制转发的方式。数据包首先沿着 X 方向然后沿着 Y 方向到达目的地[2]。

2.2.2　UpDown 算法

UpDown 多播路由将网络分为两个子网络,具体实现方法是将每个路由器从 0 到 $m \times n - 1$ 编号,假设网络的规模为 $m \times n$,则标号 $L(x,y)$ 与各个节点坐标 (x,y) 的关系为 $L(x,y) = x \times n + y$,根据标号可以将网络分为两个子集,一个是比源节点标号大的子网络集,即 D_h; 另一个是比源节点标号小的子网络集,即 D_l。将多播目的集合也划为两个子集: 假设源节点为 (x_0,y_0),集合 D 为目的集,(x,y) 为本地节点,根据以下规则将目的集划分为两个子集:

$$D_{up} = \{(x,y) \mid (x,y) \in D, L(x,y) > L(x_0,y_0)\}$$

$$D_{down} = \{(x,y) \mid (x,y) \in D, L(x,y) < L(x_0,y_0)\}$$

将目的集合 D_{up} 与 D_{down} 分别映射到网络 D_h 和 D_l 子网中,在各个子网上实现其多播路由算法。

2.2.3　SubPartition 算法

SubPartition 多播路由将网络划分为 4 个子网,在 4 个子网上只能向特定的方向上转发数据,例如子网 1 向东和南两个方向转发,子网 2 向东和北两个方向转发,子网 3 向西和北两个方向转发,子网 4 向西和南两个方向转发。假设 (x,y) 为任意节点,根据以下规则将网络划分为 4 个子网:

$$subnetwork_1 = \{(x,y) \mid [(x,y),(x+1,y)] \text{ and } [(x,y),(x,y-1)]\}$$

$$subnetwork_2 = \{(x,y) \mid [(x,y),(x+1,y)] \text{ and } [(x,y),(x,y+1)]\}$$

$$subnetwork_3 = \{(x,y) \mid [(x,y),(x-1,y)] \text{ and } [(x,y),(x,y+1)]\}$$

$$subnetwork_4 = \{(x,y) \mid [(x,y),(x-1,y)] \text{ and } [(x,y),(x,y-1)]\}$$

根据同样的方法将目的集划分为 4 个子集 D_1、D_2、D_3、D_4,将这 4 个子集分别映射到 $subnetwork_1$、$subnetwork_2$、$subnetwork_3$、$subnetwork_4$ 子网上,在各个子网上实现其多播路由算法。

3　多播拥塞控制

根据片上网络多播通信的特点,本文提出了基于数据包的拥塞控制(Packet Congestion Control, PCC),基于数据片的拥塞控制(Flit Congestion Control, FCC),基于数据包和数据片两者结合的拥塞控制(Packet_Flit Congestion Control, PFCC)的分级拥塞控制算法。分级体现在当 PCC 或 FCC 不能完全解决网络的拥塞,那么在前者的基础上再启用 PFCC,从而解决片上网络多播拥塞问题,流程如图 2 所示。FCC 和 PCC 算法分别从 flit 级和 packet 级来控制网络的拥塞,而 PFCC 是两种算法的结合,从 flit 和 packet 两级来控制拥塞,只有当 FCC 或 PCC 不能完全解决网络的拥塞时,那么进而采用 PFCC 解决网络的拥塞问题。首先根据缓存占用率来探测是否存在拥塞,若存在拥塞,则启动拥塞控制单元,拥塞控制单元包括 PCC,FCC 其中之一或者是 PCC 和 PFCC、FCC 和 PFCC 两者的组合,经过拥塞控制单元的处理后,启动决策单元,对网络采取相应的措施确保消除或减轻网络拥塞。

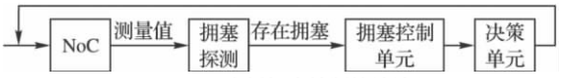

图 2　NoC 多播拥塞控制算法流程

拥塞控制流程如图 3 所示,在此采用了两级拥塞控制机

制,当探测到拥塞时,首先启动 FCC,若经过 FCC 运行后,拥塞问题得到解决,那么继续监测网络的运行情况;如果运行 FCC 仍未解决拥塞问题,那么启动 PFCC,若采用此机制后拥塞问题完全解决,那么进入下一周期探测,如果运行 PFCC 之后仍不能解决,则重复前面的步骤。

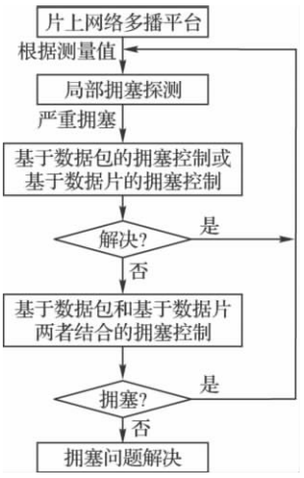

图 3 拥塞控制流程

FCC 算法是从 flit 级对网络进行控制。FCC 算法如下:

1) 每隔一定时间 T_d 测量缓存占用率 N_m。

2) 比较 N_m 与设置的阈值 N_t。

3) 若 $N_m > N_t$,则 $C_i = 1$。

4) 若 $C_i = 1$,则暂停数据注入。

5) 重复前 3 步,直至 $C_i = 0$ 时,重新注入数据。

PCC 算法是从 packet 级对网络进行控制。PCC 算法如下:

1) 每隔一定时间 T_d 测量 buffer 中存在的整包 T。

2) 比较 T 与设置的阈值 t。

3) 若 $T > t$,则等待下一个发包周期。

4) 若 $T \leqslant t$,则产生一个新的数据包。

PFCC 算法是 PCC 算法和 FCC 算法两者的结合,从 flit 和 packet 两级来控制拥塞,PFCC 算法流程如下所示:

1) 每隔一定时间 T_d 测量缓存占用率 N_m 和 buffer 中存在的整包 T。

2) 比较 N_m 与设置的阈值 N_t,同时比较 T 与设置的阈值 t。

① 若 $N_m > N_t$ 或 $T > t$,资源节点暂停产生数据包,同时 router 不转发新的数据包。

② 当 $N_m \leqslant N_t$ 或 $T \leqslant t$ 时,资源节点重新产生数据包,同时 router 开始转发新的数据包。

下面的实验中本文应用了 FCC、PCC 以及 PFCC 的方法来解决冲突。FCC、PCC 以及 PFCC 属于局部拥塞控制机制,因为它只关系到单独的一个节点。多播拥塞控制方法减少了因拥塞而产生的延时,提高了整个网络的性能。

4 实验分析

4.1 模拟实验方法

本文基于C++语言分别构建了几种(如 4×4,5×6,8×8) 双通道 2D mesh 拓扑结构的 NoC 单播和多播模拟器,模拟器中采用虫孔交换机制,模拟环境中数据的最小单元是 flit,每个物理通道分为 4 个虚通道,模拟器的各个参数配置如表 1所示。模拟器参数的配置首先是根据模拟器本身的结构和仿真实验的需要来设置的,其次是根据对大量仿真结果数据的分析,从而得到最佳的参数配置;比如网络拓扑和路由算法这两个参数的设置就是根据模拟器本身的结构而设置的。另外比如模拟周期,如果设置的周期过于小,那么将不能产生足够的数据包,从而不能准确统计整个网络的各个性能参数。

表 1 模拟器参数配置

参数	值
网络拓扑	4×4/5×6/8×8
虚通道个数	4
片的长度/b	16/64
流量模式	Uniform
路由算法	Original/UpDown/SubPartition
模拟周期	80 000/120 000(10% warm-up)

本文实验分为 3 个部分:1) 在网络规模为 4×4 时,比较了在相同地址序列情况下多播和单播的平均延时、平均跳数、带宽利用率和平均负载;2) 在网络规模为 5×6 时,比较了 FCC、PFCC 以及无拥塞控制情况下多播的平均延时、平均跳数、带宽利用率和平均负载;3) 在网络规模为 5×6 时,比较了 PCC、PFCC 以及无拥塞控制情况下各参数对网络性能的影响。

4.2 多播与单播性能比较

本实验在没有拥塞的情况下比较了单播和多播的各种性能参数,包括平均延时(周期)、平均跳数(跳)、运行负载(帧/周期/路由)和带宽利用率。在实验中目的地址是模拟器随机产生,并将多播和单播模拟器分别设置相同的地址,flit 长度为 16 b,图 4 分别比较了多播和单播的性能,从图中可看出在不同的目标节点下,总体上多播较单播具有更好的性能。

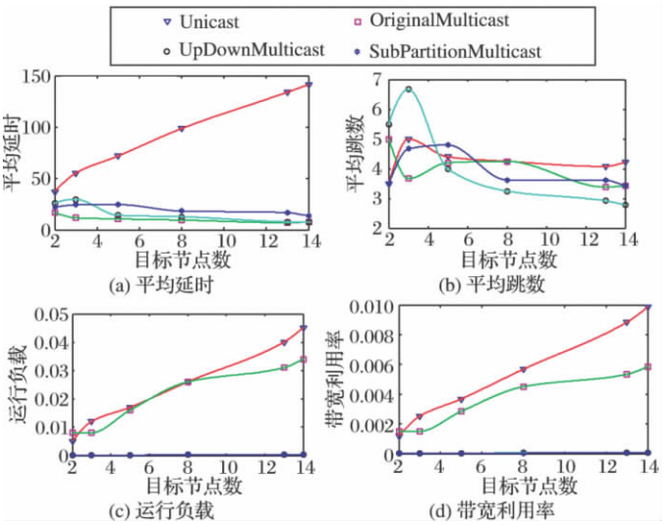

图 4 单播和多播性能比较(4×4 2D Mesh)

从图 4 可看出,在相同地址的情况下,总体上多播比单播具有更小的平均延时和平均跳数。当目标节点数很少时,单播的性能可能优于多播。但是随着目标节点数的增多,多播通信的优势越来越明显,当目标节点数为 2 时,单播和多播的延时区别不大,当目标节点数为 8 时,多播的平均延时只有单播的 25%,平均跳数只有单播的 77%。

多播比单播具有更小带宽利用率和单位负载。特别是 UpDown、SubPartition 带宽利用率和负载非常低,当目标节点为 8 时,单播的带宽利用率是两者的 130 倍,且这种趋势随着节点数的增多变得明显。图 4 中目标节点为 3 时基于中

UpDown 的多播的平均跳数大于单播的平均跳数，这是因为 UpDown 只能向 X 和一个 $Y2$ 个链路方向上转发数据，而单播可以向 4 个链路方向上转发。

4.3　FCC 和 PFCC 机制对网络性能的影响

本实验在网络规模为 5 ×6 时，比较 FCC 和 PFCC 机制对网络性能的影响，实验采用的路由算法为 SubPartition 算法，flit 个数为 32，注入包的数目为 45 000，T_d = 300，N_t = 0.2。图 5显示了 FCC 和 PFCC 机制对网络性能的影响，从图 5 可看出网络拥塞的情况得到解决，性能参数得到改善。

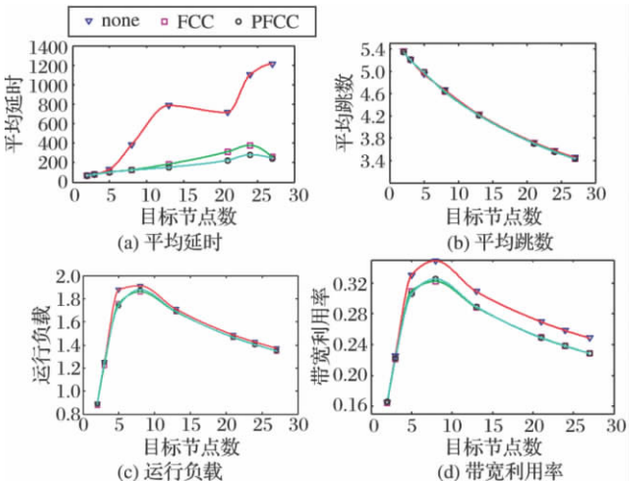

图 5　FCC 和 PFCC 机制对网络性能的影响(5 ×6 2D Mesh)

从图 5 可看出，FCC 和 PFCC 机制下多播的平均延时、跳数、负载、带宽利用率呈下降趋势，特别是延时效果显著，当目标节点较少时效果不明显，因为网络未产生拥塞。但是随着目标节点数的增多，网络出现拥塞，拥塞控制机制的优势显现出来。当节点数为 21 时，没有拥塞控制的情况下，网络中出现大量丢包，当加上拥塞控制机制时，网络中基本上不会存在丢包现象。FCC 的延时只有不加拥塞控制时的 43%，平均跳数只有无拥塞机制时的 93%，负载只有无拥塞机制时的 91%，带宽利用率为只有无拥塞机制时的 85%。

4.4　PCC 和 PFCC 机制对网络性能的影响

本实验在网络规模为 5 ×6 时，比较 PCC 和 PFCC 机制对网络性能的影响，实验采用的路由算法为 SubPartition 算法，flit 个数为 32，注入包的数目为 45 000，T_d = 300，t = 10。图 6 显示了 PCC 和 PFCC 机制对网络性能的影响，从图 6 可看出网络拥塞控制的效果，网络拥塞的情况得到了有效的解决。

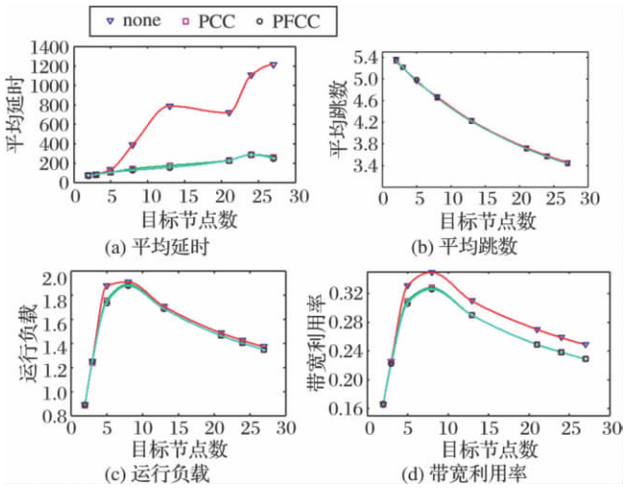

图 6　PCC 和 PFCC 机制对网络性能的影响(5 ×6 2D Mesh)

从图 6 可看出，PCC 和 PFCC 机制下多播的平均延时、跳数、负载、带宽利用率呈下降趋势，特别是延时效果显著，当目标节点较少时效果不明显，因为网络未产生拥塞; 但是随着目标节点数的增多，网络中出现拥塞，拥塞控制机制的优势显现出来，当节点数为 24 时，没有拥塞控制的情况下，网络中出现大量丢包，当加上拥塞控制机制时，网络中基本上不存在丢包现象。FCC 的延时只有不加拥塞控制时的 26%，平均跳数只有无拥塞机制时的 95%，负载只有无拥塞机制时的 90%，带宽利用率为只有无拥塞机制时的 83%。同时可看出，随着目标节点数的增多，SubPartition 更适合多播通信。

4.5　性能分析

从以上的实验，本文可以得到以下性能分析:

1) 在相同固定地址情况下，多播比单播具有更好的性能。

2) 在网络未死锁或拥塞的时候，3 种算法的延时等性能参数都呈下降趋势，UpDown 和 SubPartition 两种算法随着节点数增多更适合多播通信。

3) 目标节点数增大时，Original 算法更容易产生拥塞，导致延时增大，使性能急剧下降。

4) UpDown 其纵向带宽利用率低，参数变化范围较大，SubPartition 算法在每个子网上数据只能向两个链路上转发，因此变化平稳。

5) 在一定范围内(当目标节点占总节点数比率较小，约 40%) ，Original 算法性能较好; SubPartition 算法更适合大规模且目标节点数多时的情况; UpDown 算法有时性能较好，但不稳定。

6) FCC、PCC 以及 PFCC 拥塞控制机制效果明显。当没有采取拥塞控制时，网络极易出现拥塞以及大量丢包，网络通信的性能急剧下降。当采用拥塞控制机制后，网络拥塞的情况消除或有所改进，更能发挥多播通信的优势。

5　结语

从仿真实验结果看，片上网络的多播通信比单播具有更小的网络延时和平均跳数且负载分配均匀，随着规模增大有更明显的效果。具有死锁避免功能的 UpDown 和分块路由比传统 XY 路由算法，总体上显示出更好的性能，特别是，分块路由对于规模较大且目标节点数较多时，具有更好的效果。应用拥塞控制策略，能更有效地利用多播通信，提高片上网络的性能。

参考文献:

[1] LU ZHONGHAI, YIN BEI, JANTSCH A. Connection-oriented multicasting in wormhole-switched networks on chip [C]// Proceedings of 2006 Emerging VLSI Technologies and Architectures. Karlsruhe: IEEE Computer Society, 2006: 6 - 11.

[2] LIN XIAOLA, MCKINLEY P K, NI L M. Deadlock-free multicast wormhole routing in 2-D mesh multicomputers [J]. IEEE Transactions on Parallel and Distributed Systems, 1994, 5(8): 793 - 804.

[3] CARARA E A, MORAS F G. Deadlock-free multicast routing algorithm for wormhole-switched mesh networks-on-chip [C]// IEEE Computer Society Annual Symposium on VLSI. Washington, DC: IEEE Computer Society, 2008: 341 - 346.

[4] JAGANNATHAN S, ALMEROTH K C. Using tree topology for multicast congestion control [C]// Proceedings of 2001 International Conference on Parallel Processing. Washington, DC: IEEE Computer Society, 2001: 313 - 321.

(下转第 2637 页)

下面对 6 个不同的实例进行相应的数值计算,通过对比数值结果和解析结果来验证上述解析表达式的正确性。在数值计算中,我们采用的参数取值为 $BSS = 2$ b, $C_o = 113$ b。

实例 1　选择文献 [4] 中的实例。FlexRay 总线网络中共有 6 个节点,有 21 个信号在静态段中传输,各个信号的字节长度分别为 4,8,8,3,6,8,16,3,8,6,7,3,1,8,5,2,11,1,5,8,1。

实例 2　FlexRay 总线网络中共有 4 个节点,有 21 个信号需要在静态段中传输,各个信号的字节长度分别为 50,65,7,14,6,80,33,2,6,8,9,2,1,3,6,80,5,3,5,7,5。

实例 3　FlexRay 总线网络中共有 8 个节点,有 34 个信号需要在静态段中传输,各个信号的字节长度分别为 10,6,70,18,6,20,35,12,66,8,9,2,11,3,6,80,5,4,7,7,6,22,35,7,10,3,12,5,7,6,22,6,4,8。

实例 4　FlexRay 总线网络中共有 6 个节点,有 24 个信号需要在静态段中传输,各个信号的字节长度分别为 10,24,12,18,22,10,32,12,16,8,10,30,18,22,16,20,3,10,6,6,8,25,20,18。

实例 5　FlexRay 总线网络中共有 10 个节点,有 40 个信号需要在静态段中传输,各个信号的字节长度分别为 8,14,12,18,6,10,2,12,16,8,10,20,9,12,16,10,3,10,6,6,8,8,10,18,4,6,2,6,18,5,13,4,15,10,13,17,15,13,12,10。

实例 6　FlexRay 网络中共有 10 个节点,有 32 个信号在静态段中传输,各个信号的字节长度分别为 4,4,4,4,1,2,1,4,4,2,1,4,4,2,4,4,4,4,2,2,2,2,2,2,2,1,2,2,1,2,2,1。

从表 1 可清楚地看到,6 个实例的解析结果都与数值结果吻合得非常好。这就进一步验证了所得解析结果的正确性。一般来说,一个 FlexRay 网络中静态段的时间长度可以超过其整个通信周期长度的 70%。因此本文得到的静态段最大带宽利用率就可以近似地认为是整个网络带宽利用率。

表 1　6 个实例的解析结果与数值结果

实例序列号	解析结果($U_{\max}$)	数值结果($U_{\max}$)
1	0.220 2	0.220 2
2	0.301 0	0.301 0
3	0.315 7	0.315 7
4	0.320 8	0.320 8
5	0.271 9	0.271 9
6	0.153 9	0.153 9

4　结语

带宽利用率是描述一个 FlexRay 网络性能的重要指标。本文在现有 FlexRay 时间优化模型的基础上,通过运用一定的数学近似方法,推导出了 FlexRay 静态帧净荷段最优长度和最大带宽利用率的解析表达式。结果表明,FlexRay 静态帧净荷段最优长度与所有静态信号的平均字节长度的平方根成正比。数值实验进一步验证了所得解析表达式的正确性,为今后 FlexRay 网络的性能分析和设计工作提供了理论参考。

参考文献:

[1] 王婧, 张欣. 汽车网络通信协议 TTP/C 和 FlexRay 的研究分析[J]. 北京汽车, 2006, 10(6): 40 – 43.

[2] 王锴, 王宏, 徐皑冬. 下一代车载网络 FlexRay 及其应用研究[J]. 计算机工程与应用, 2008, 44(20): 77 – 79,98.

[3] POP T, POP P, ELES P, *et al*. Timing analysis of the FlexRay communication protocol [J]. Real-Time Systems, 2008, 39(1): 205 – 235.

[4] PARK I, SUNWOO M. FlexRay network parameter optimization method for automotive applications [J]. IEEE Transactions on Industrial Electronics, 2011, 58(4): 1449 – 1459.

[5] ZENG HAIBO, NATALE M D, GHOSAL A, *et al*. Schedule optimization of time-triggered systems communicating over the FlexRay static segment [J]. IEEE Transactions on Industrial Informatics, 2011, 7(1): 1 – 17.

[6] KANAJAN S, ABELL J. Sensitivity analysis on Flexray dynamic segment design parameters [C]// Proceedings of the Fourth International Conference on Systems and Networks Communications. Piscataway, NJ: IEEE Press, 2009: 12 – 18.

[7] SCHMIDT E G, SCHMIDT K. Message scheduling for the FlexRay protocol: the dynamic segment [J]. IEEE Transactions on Vehicular Technology, 2008, 58(5): 2160 – 2169.

[8] SCHMIDT K, SCHMIDT E G. Message scheduling for the FlexRay protocol: The static segment [J]. IEEE Transactions on Vehicular Technology, 2008, 58(5): 2170 – 2179.

[9] 赵睿, 秦贵和, 范铁虎. FlexRay 通信协议的总线周期优化[J]. 计算机应用研究, 2010, 27(10): 3847 – 3850.

[10] 李佳, 田光宇. FlexRay 网络通信延迟时间分析[J]. 清华大学学报: 自然科学版, 2007, 47(8): 1343 – 1346.

[11] HWANG C L, CHANG L J, YU Y S. Network-based fuzzy decentralized sliding-mode control for car-like mobile robots [J]. IEEE Transactions on Industrial Electronics, 2007, 54(1): 574 – 585.

[12] LI HONGBO, CHOW MOYUEN, SUN ZENGQI. EDA-based speed control of a networked DC motor system with time delays and packet losses [J]. IEEE Transactions on Industrial Electronics, 2009, 56(5): 1727 – 1735.

[13] MINSUK S, MYOUNGBO S. Optimal period and priority assignment for a networked control system scheduled by a fixed priority scheduling system [J]. International Journal of Automotive Technology, 2007, 8(1): 39 – 48.

(上接第 2633 页)

[5] LOTFI-KAMRAN P, RAHMANI A M, DANESHTALAB M, *et al*. EDXY — A low cost congestion-aware routing algorithm for network-on-chips [J]. Journal of Systems Architecture, 2010, 56(7): 256 – 264.

[6] WANG XU, GAN GE, FAN DONGRUI, *et al*. GFFC: The global feedback based flow control in the NoC design for many-core processor [C]// Proceedings of the Sixth IFIP International Conference on Network and Parallel Computing. Washington, DC: IEEE Computer Society, 2009: 227 – 232.

[7] GU HUAXI, XU JIANG, WANG KUN. A new distributed congestion control mechanism for networks on chip [J]. Telecommunication Systems, 2009, 44(3): 321 – 331.

[8] YUAN JINGLING, HE JINGJING, ZHONG LUO. GA based congestion aware algorithm for application in NoC [C]// International Conference on Computational Intelligence and Software Engineering. Washington, DC: IEEE Computer Society, 2009: 1 – 5.

[9] YUAN JINGLING, LIU HUA, JIANG XING, *et al*. Key techniques of multicast communication for network on chip [C]// International Conference on Internet Technology and Applications. Washington, DC: IEEE Conputer Society, 2010: 1 – 5.