

一种基于双缓冲和 XYX 路由的片上网络容错机制设计

周 芳, 吴 宁

(南京航空航天大学 电子信息工程学院, 江苏 南京 210016)

摘 要: 提出了一种基于双缓冲和 XYX 路由的容错机制. 该机制通过使用双缓冲存储数据包来减少数据发送的等待延时, 并采用 XYX 路由算法缩短路由路径, 达到通信延时和功耗优化的目的, 实现数据的高可靠高效率传输. 通过 System C 所搭建的 4×4 的 2D-Mesh 网络平台验证了该机制的有效性, 实验结果表明, 基于双缓冲和 XYX 路由的容错机制相对端到端随机路由算法具有更低的延时和功耗, 当失效率小于 2% 时, 数据的传输延时和功耗开销可分别降低约 50% 和 20%.

关键词: 片上网络; 瞬时故障; 容错; XYX 路由算法; 双缓冲

中图分类号: TP301

文献标识码: A

文章编号: 1000-7180(2015)04-0025-06

A Fault-tolerant Mechanism Based on Dual-buffer and XYX Routing for NoC

ZHOU Fang, WU Ning

(College of Electronic and Information Engineering, Nanjing University of
Aeronautics and Astronautics, Nanjing 210016, China)

Abstract: This paper proposes a fault-tolerant mechanism based on dual-buffer and XYX routing. The mechanism reduces the delay by dual buffer and optimizes the power by XYX routing strategy, which can achieve better performance in the condition of high reliability. System C is used to program the NoC router and build the 4×4 2D mesh NoC platform for verifying the validity of the fault-tolerant mechanism. The results show that the data transmission delay and power consumption overhead of this mechanism were reduced by approximately 50% and 20% compared to end-to-end random routing algorithm when the failure rate is less than 2%.

Key words: network on chip; transient faults; fault-tolerant; XYX routing; dual-buffer

1 引言

目前, 解决路由瞬时故障^[1-2]的主要方法有基于随机通信的容错路由^[3]以及基于检错纠错码的重传机制^[4]. 在随机通信机制中, 路由节点以一定的概率向它的相邻节点以广播的形式发送数据包, 直到目的节点收到数据包. 目的节点从它的相邻节点获取多份数据包拷贝, 并从中接收一份正确的数据包^[5]. 该机制的优点在于实现简单且没有任何硬件的额外开销, 但该机制在转发数据包时会使网络中充斥大量的冗余包, 从而可能导致网络拥塞. 因此, 更多的

研究者倾向于采用基于检错纠错码的重传机制来解决瞬时故障. 其中, 文献[6]中提出的重传机制中, 故障控制模式采用的是停止/等待模式, 该模式在数据包发送完毕后需要等待 ACK 反馈的到达, 这样会导致较大的等待延时. 文献[7]中提出了一种端到端随机路由算法, 该算法结合了端到端的重传机制和随机通信机制来实现数据的容错传输. 但由于通过路由查找表存储路由向量, 需要大量的路由向量存储单元, 所以增加了面积和功耗开销. 文献[8]中提出了一种 XYX 容错路由算法, 该算法通过采用 XY 路由和 YX 路由交替传递以降低数据包错误传

收稿日期: 2014-05-28; 修回日期: 2014-07-23

基金项目: 国家自然科学基金资助项目(61376025); 江苏省产学研联合创新基金(BY2013003-11)

输的概率,但无法保证 100%容错.文献[9]中采用多路径传输的方法来解决瞬时故障问题,但多路径传输使用的不一定是最短路径,容易造成拥塞和延时以及功耗的浪费.

针对上述算法的缺点,本文借鉴端到端随机路由算法的思想,并结合原有检错重传方案的优点,提出了一种基于双缓冲和 XYX 路由的容错机制.该机制采用双缓冲存储器来减少数据包检错重传的等待延时,并使用 XYX 路由代替随机路由,缩短路由路径,达到减少延时和功耗的目的.从而在保证容错性能的前提下,优化了系统延时和功耗.

2 基于双缓冲和 XYX 路由的容错设计方法

2.1 基于检错重传的双缓冲发送机制

传统的检错重传机制在发送完整个微片集后,如果没有收到对应的 ACK 反馈包,发送端将继续重新发送之前发送的微片集.这样虽然充分利用了发送端等待接收 ACK 反馈包的时间,但算法仍存在较大的延时开销.如果在发送端正确发送完整个微片集之后,继续发送新的数据微片,而不是冗余微片,就能充分利用发送端发送冗余微片的延时.或者发送端在收到 ACK 后,待发送的微片已经存储在数据缓冲区中,发送端不需要等待数据缓冲区的刷新,这样也将减少传输延时.基于上述考虑,本文采用基于检错重传的双缓冲循环机制来发送微片.

在发送端,将两个微片集分别存储在两个数据缓冲区中.发送端算法的处理流程如图 1 所示.

图 1 中,发送端首先发送缓冲区 A 中的数据,当缓冲区 A 中的数据发送完毕后,接着发送缓冲区 B 中的数据,由于此后随时有可能收到来自接收端的对应缓冲区 A 中的 ACK 反馈包,所以发送端在发送缓冲区 B 中的每一个微片时,需要判断是否有对应缓冲区 A 的 ACK 反馈包到达.如果发送端接收到对应缓冲区 A 的 ACK 反馈包,则立即清空缓冲区 A,并将下一个新的微片集放入缓冲区 A 中,当缓冲区 B 中的数据发送完毕后,又开始发送缓冲区 A 中的数据,同时也需要判断缓冲区 B 中的 ACK 反馈包是否到达.

如果缓冲区 B 中的数据发送完毕之后,仍然没有收到来自接收端的对应缓冲区 A 的 ACK 反馈包,则重新开始发送缓冲区 A 中的数据,并判断是否有 ACK 反馈包到达,如果此时发送端接收到对应缓冲区 A 的 ACK 反馈包,则发送端停止发送缓冲区 A 中的数据,转而发送缓冲区 B 中的数据,并

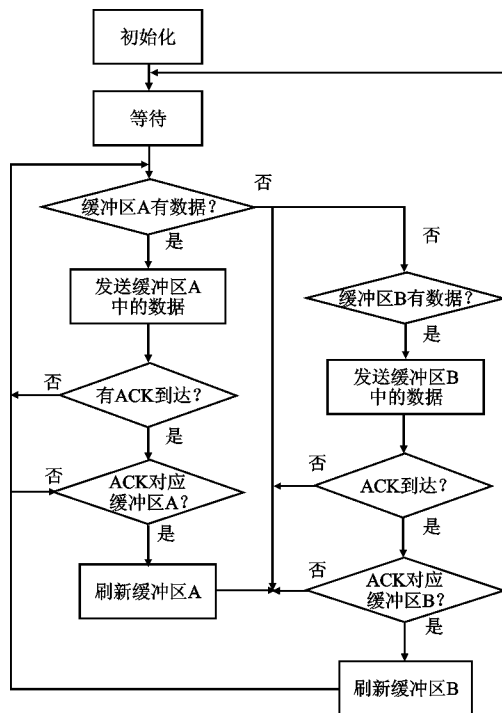


图 1 容错机制发送端算法流程图

立即清空缓冲区 A,将下一个新的微片集放入缓冲区 A 中,如果此时接收到对应缓冲区 B 的 ACK 反馈包,则立即清空缓冲区 B,并将下一个新的微片集放入缓冲区 B 中.

在接收端,其算法处理流程如图 2 所示.

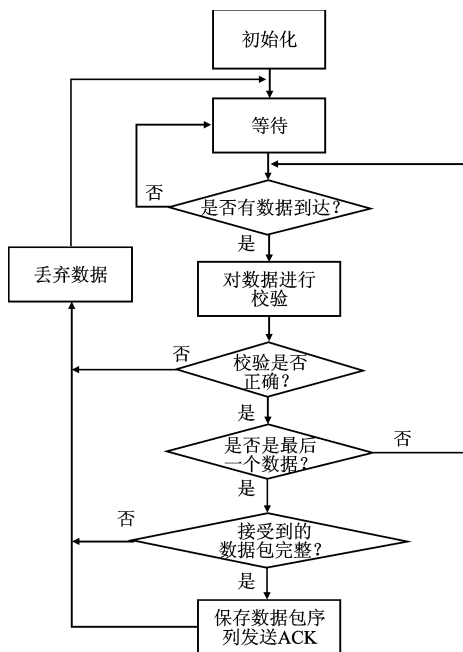


图 2 容错机制接收端算法流程图

图 2 中,当微片到达接收端之后,接收端首先对收到的微片进行校验,判断其是否正确,如果是正确

的,再判断是否是顺序到达的微片,如果是错误的或者乱序的微片,则直接丢弃该微片,并丢弃随后接收到的属于同一微片集中的所有微片,直到正确接收到新的微片集.当接收端收到一个完整正确的微片集后,接收端将立即发送对应微片集的ACK反馈包.为了提高网络通信的可靠性,算法采用三模冗余模型发送ACK反馈包.

使用双缓冲机制的数据发送过程中,如果出现了传输错误,则检错重传过程如图3所示.

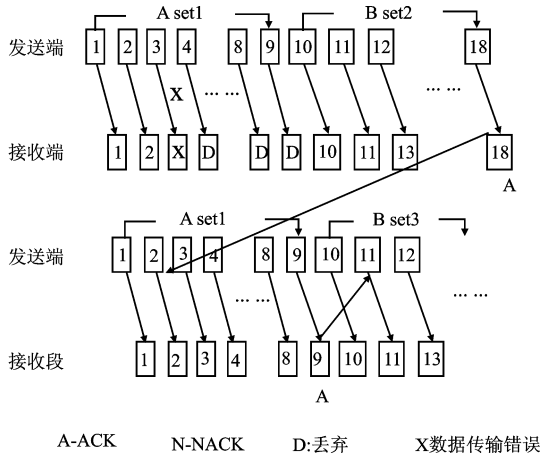


图3 双缓冲机制检错重传示意图

图3中,发送缓冲区A中的微片集set1, flit3传输出错,接收端检测到微片出错后丢弃该微片以及之后的set1中的其他微片;发送端发送完set1后继续发送缓冲区B中的微片集set2,当整个微片集都被正确接收后,接收端发送ACK给发送端,发送端接收到ACK后,清空缓冲区B,并重新输入新的微片集;当发送端发送完缓冲区B中的微片集后,由于缓冲区A中的数据传输错误,所以又继续发送缓冲区A中的微片,如此循环,直到数据发送完毕.

2.2 XYX路由策略

为了降低数据重传的次數并减轻网络带宽负载,本文设计发送端在发送缓冲区中的微片时,采用XYX路由算法.XYX路由算法是指发送端首先采用XY路由策略发送缓冲区中的数据,一段时间之后,如果没有收到对应的ACK反馈包,则采用YX路由策略重新发送缓冲区中的微片.一段时间之后,如果还是没有收到ACK反馈包,则再采用XY路由策略重新发送缓冲区中的微片.如此循环,直到收到来自接收端的对应缓冲区的ACK反馈包为止.发送端接收到ACK反馈包后,立即清空缓冲区并存入新的数据微片.XYX路由算法流程图如图4所示.

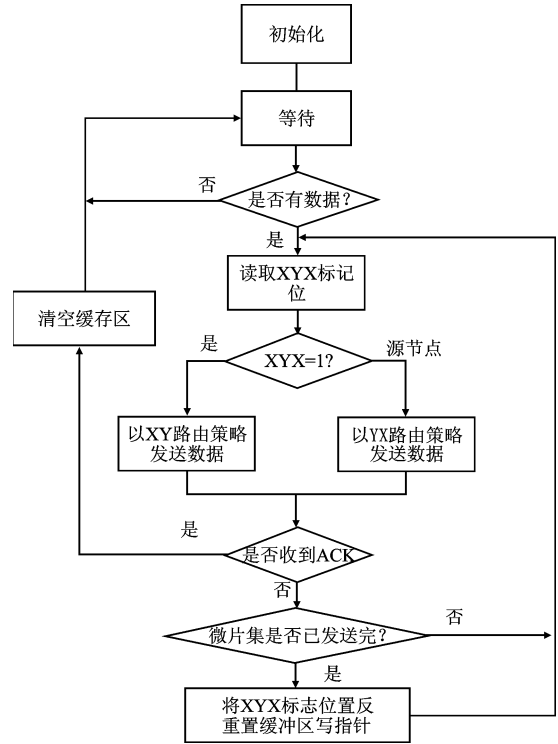


图4 XYX路由算法流程图

3 容错机制性能分析

假设每个数据包都被拆分为 N_f 个微片,一个包头微片从源节点到达目的节点所需时间为 RTT ,在这段时间内,源节点可以连续发送 N_t 个微片,失效率用 E_r 来表示,定义出错的微片的平均值为 $N_r = E_r \times N_f$.在此假设下,本节将对本文所提出的容错方法的延时和功耗开销进行分析,并与端到端随机路由算法进行比较.

3.1 延时性能分析

数据包的发送延时由包头微片发送延时、数据微片发送延时以及等待延时三部分构成.为方便分析比较,每个数据包的包头微片的传输延时都用所有包头微片的平均传输延时 L_{head} 来表示,每个数据包的数据微片的传输延时也都用所有数据微片的平均传输延时 L_{data} 来表示,因此数据包发送的平均延时可表示为

$$L_{average} = \frac{\sum_{i=1}^N [L_{head} + L_{data} \times (n_i - 1) + L_{wait}(i)]}{N} \quad (1)$$

由式(1)可以看出,算法的平均延时的比较可以归结为在发送相同有效数据的情况下,发送的微片数目 n_i 以及数据包的等待延时 $L_{wait}(i)$ 的比较.

(1) 基于双缓冲和 XYX 路由的容错算法

当 $E_r = 0$ 时,若发送一个数据包,有 $n_1 = N_f + N_t, L_{\text{wait}} = 0$,由于数据包是连续发送的,即当发送完第一个数据包后,如果没有等到 ACK,发送端会继续发送第二个数据包。所以当发送 N 个数据包时,有:

$$L_{\text{average}} = \frac{L_{\text{head}} + L_{\text{data}} \times (N \times N_f + N_t - 1)}{N} \quad (2)$$

当 $E_r \neq 0$ 时,考虑算法的最坏情况,即出错的微片不重叠,若发送一个数据包,由于当数据包中有微片出错时,该微片集的所有微片都将被重新发送,所以有 $n_1 = N_f \times (N_r + 1) + N_t, L_{\text{wait}} = 0$,因此,如果发送 N 个数据包,有:

$$L_{\text{average}} = \frac{L_{\text{head}} + L_{\text{data}} \times [N_f \times (N_r + 1) \times N + N_t - 1]}{N} \quad (3)$$

(2) 端到端随机路由算法

如果采用端到端随机路由算法,则当 $E_r = 0$ 时,若发送一个数据包,有 $n_1 = N_f + N_t, L_{\text{wait}} = 0$,由于数据包是单个发送,所以当发送 N 个数据包时,有:

$$L_{\text{average}} = L_{\text{head}} + L_{\text{data}} \times (N_f + N_t - 1) \quad (4)$$

当 $E_r \neq 0$ 时,考虑算法的最坏情况,即出错的微片不重叠,若发送一个数据包,有 $n_1 = N_f \times (N_r + 1) + N_t, L_{\text{wait}} = 0$,所以当发送 N 个数据包时,有:

$$L_{\text{average}} = L_{\text{head}} + L_{\text{data}} \times [N_f \times (N_r + 1) + N_t - 1] \quad (5)$$

因此,可以得到以下结论:

(1) 当 $E_r = 0$ 时,基于双缓冲和 XYX 路由的检错重传机制与端到端随机路由算法的平均延时的大小与 N 有关;当 $N = 1$ 时,端到端随机路由算法和基于双缓冲和 XYX 路由的检错重传机制的平均延时相等;随着 N 的增大,基于双缓冲和 XYX 路由的检错重传机制的平均延时小于端到端随机路由算法;

(2) 当 $E_r \neq 0$ 时,端到端随机路由算法和基于双缓冲和 XYX 路由的检错重传机制在最坏情况下的平均延时都随着 E_r 的增大而不断增大,但基于双缓冲和 XYX 路由的检错重传机制的平均延时仍小于端到端随机路由算法。由此可见,基于双缓冲和 XYX 路由的检错重传机制的延时更小。

3.2 功耗性能分析

在片上网络中,数据包的功耗主要由微片路由

计算功耗、数据微片路由功耗以及链路功耗三部分构成。为便于分析比较,每个数据包中的微片路由计算功耗都用微片路由计算功耗的平均值 E_{r_head} 来表示,每个数据包中的数据微片路由功耗都用数据微片路由功耗的平均值 E_{data} 来表示,链路功耗也用平均链路功耗 E_{link} 表示,因此可以定义片上网络的平均功耗为

$$E_{\text{average}} = \frac{\sum_{i=1}^N [E_{r_head} \times d'_i + E_{data} \times (n'_i - d'_i) + E_{link} \times n'_i]}{N} \quad (6)$$

由式(6)可知,算法的平均功耗比较归结于在发送相同有效数据的情况下,发送的微片总数 n'_i 和需要通过路由计算的微片数目 d'_i 的比较。

(1) 基于双缓冲和 XYX 路由的容错算法

当 $E_r = 0$ 时,发送端每发送一个数据包,需要发送一个包头微片,多个数据微片,当未接收到 ACK 时,会继续发送下一个数据包,而接收端在正确接收完整个微片集后,会发送一个 ACK 反馈包,所以当发送一个数据包时,有 $d'_1 = 3, n'_i = N_f + N_t + 1, n'_i - d'_1 = N_f + N_t - 2$,从而推出当发送 N 个数据包时,有:

$$E_{\text{average}} = \frac{E_{r_head} \times (1 + 2 \times N) + E_{data} \times [N \times (N_f - 1) + N_t - 1]}{N} + \frac{E_{link} \times [N \times (N_f + 1) + N_t]}{N} \quad (7)$$

当 $E_r \neq 0$ 时,考虑算法的最坏情况,即出错的微片不重叠,而且出错的都是包头微片,当发送一个数据包时,有 $d'_1 = N_r + 3, n'_1 = N_f \times (N_r + 1) + N_t + 1, n'_1 - d'_1 = (N_f - 1) \times (N_r + 1) + N_t - 1$,所以当发送 N 个数据包时,有:

$$E_{\text{average}} = \frac{E_{r_head} \times [N \times (N_r + 2) + 1] + E_{data} \times [N \times (N_f - 1) \times (N_r + 1) + N_t - 1]}{N} + \frac{E_{link} \times [N \times (N_f \times (N_r + 1) + 1) + N_t]}{N} \quad (8)$$

(2) 端到端随机路由算法

如果采用端到端随机路由算法,则当 $E_r = 0$ 时,发送端每发送一个数据包,需要发送一个包头微片和多个数据微片,当未接收到 ACK 时,会重新发送包头微片及随后的数据微片,而接收端在正确接收完整个微片集后,会发送一个 ACK 反馈包,所以当发送一个数据包时,有 $d'_1 = 3, n'_i = N_f + N_t + 1, n'_i - d'_1 = N_f + N_t - 2$,从而推出当发送 N 个数据包时,有:

$$E_{\text{average}} = 3 \times E_{r_head} + E_{data} \times (N_f + N_t - 2) +$$

$$E_{\text{link}} \times (N_f + N_t + 1) \quad (9)$$

当 $E_r \neq 0$ 时,考虑算法的最坏情况,即出错的微片不重叠,而且出错的都是包头微片,当发送一个数据包时,有 $d_{1'} = N_r + 3, n_{1'} = N_f \times (N_r + 1) + N_t + 1, n_{1'} - d_{1'} = (N_f - 1) \times (N_r + 1) + N_t - 1$,所以当发送 N 个数据包时,有:

$$E_{\text{average}} = E_{r_head} \times (N_r + 3) + E_{\text{data}} \times [(N_f - 1) \times (N_r + 1) + N_t - 1] + E_{\text{link}} \times [N_f \times (N_r + 1) + N_t + 1] \quad (10)$$

因此,可以得出以下结论.

(1) 当 $E_r = 0$ 时,基于双缓冲和XYX路由的检错重传机制与端到端随机路由算法的功耗开销与 N 有关:当 $N = 1$ 时,端到端随机路由算法和基于双缓冲和XYX路由的检错重传机制的功耗开销相等,随着 N 的增大,基于双缓冲和XYX路由的检错重传机制的功耗开销略小于端到端随机路由算法.

(2) 当 $E_r \neq 0$ 时,端到端随机路由算法和基于双缓冲和XYX路由的检错重传机制在最坏情况下的功耗开销都随着 E_r 的增大而增大,但基于双缓冲和XYX路由的检错重传机制的功耗开销略小于端到端随机路由算法.

通过上述分析得出,基于双缓冲和XYX路由的检错重传机制通过在本地输入端口设置双缓冲减少了数据包的发送等待时间,相对端到端随机路由算法,减少了延时时间.同时采用XYX路由代替随机路由,减少了冗余包的发送数目,从而降低了网络的功耗开销.

4 实验结果及分析

为了验证所提出的基于双缓冲和XYX路由的检错重传机制,本文使用System C系统设计语言设计了一个双通道的路由器系统级模型,以实现该检错重传机制,并利用该路由器模型实例化16个路由节点,为每个路由节点分配地址,构建了一个 4×4 的2D-Mesh网络平台来进行仿真实验.

在片上网络中,产生瞬时故障的概率一般为 $10^{-9} \sim 10^{-20}$ ^[10],由于该概率太小,所以在实验分析过程中仿真失效率设为1%~5%区间,又因为瞬时故障的产生具有随机性和偶然性,所以设置当失效率为1%时,发送100个微片中随机有一个微片出错;当失效率为5%时,发送100个微片中随机有5个微片出错.

4.1 容错性能分析

基于双缓冲和XYX路由的检错重传机制和端

到端随机路由算法都必须进行反馈确认,如果接收端没有收到正确的数据,算法将不会停止进行.因此,基于双缓冲和XYX路由的检错重传机制和端到端随机路由算法的容错性能都几乎为100%.

4.2 延时分析

假设一个数据包被拆分为15个微片,每个微片大小为32比特,设置注入率为0.3微片·周期⁻¹·节点⁻¹,失效率为2%,我们随机选取1个节点(2,2)为源节点传送数据,进行仿真实验,验证其延时性能,实验结果如图5所示.

从图中可以看出,缓冲区A在发送第2和第7个数据包时的传输延时比较大,而缓冲区B在发送第3和第8个数据包时,传输延时比较大.这是由于网络中存在2%的失效率,即每发送100个微片就有一个微片出现故障,这个故障随机的出现在了缓冲区A发送第2和第7个数据包及缓冲区B发送第3和第8个数据包的时候.由于基于双缓冲和XYX路由的检错重传机制必须进行反馈确认,如果接收端没有收到正确的数据,发送端将继续发送该数据直到收到对应的ACK反馈包为止,因此传输延时会大大增加.

由于基于双缓冲和XYX路由的检错重传机制的延时性能与失效率有关,所以实验在注入率为0.5微片·周期⁻¹·节点⁻¹下,通过在不同失效率条件下,对基于双缓冲和XYX路由的检错重传机制和端到端随机路由算法进行延时性能分析.不同失效率下算法的平均延时如图6所示.

从图6中可以看出,基于双缓冲和XYX路由的检错重传机制和端到端随机路由算法的平均延时随着失效率的增加而不断增加.当失效率小于4%时,基于双缓冲和XYX路由的检错重传机制的延时开销远小于端到端随机路由算法,特别当失效率小于2%时,基于双缓冲和XYX路由的检错重传机制的转发延时比端到端随机路由减少将近50%.而当失效率大于4%时,基于双缓冲的检错重传机制的延时曲线几乎与端到端随机路由算法的曲线重合,即两者延时开销基本相等.

4.3 功耗分析

在片上网络中,一般只考虑数据包的转发功耗,而数据包的转发功耗取决于转发数据包的数目,所以可以通过统计从源节点向目的节点发送100个数据包时,基于双缓冲和XYX的检错重传机制和端到端随机路由算法中转发数据包的大小来比较功耗大小.不同失效率下算法发送的平均微片数目如图7所示.

在传送数据包时,随着失效率的增大,基于双缓冲和 XYX 的检错重传机制与端到端随机路由算法的平均发送微片数目也随之增大.当失效率小于 2% 时,基于双缓冲和 XYX 路由的检错重传机制转发的平均微片数目比端到端随机路由算法小约 30%,当失效率大于 2% 时,基于双缓冲和 XYX 路由的检错重传机制中转发的总微片数的平均值约等于端到端随机路由算法.

5 结束语

本文设计了一种基于双缓冲和 XYX 路由的容错机制.该机制通过采用双缓冲机制和 XYX 路由策略来循环交替发送数据包,从而减少等待延时和冗余包发送数目,在保证容错性能的同时,达到了延时和功耗优化的目的.通过使用 SystemC 设计的路由器模型构建的 4×4 的 2D-Mesh 网络平台,进行了多组仿真实验,实验结果表明,基于双缓冲和 XYX 路由的检错重传机制具有接近 100% 的容错性能,而且在失效率小于 2% 时,数据的传输延时和功耗开销相对端到端随机路由算法分别降低约 50% 和 20%,从而保证了高可靠高效率的数据传输.

参考文献:

- [1] Simn Ogg, Bashir Al-Hashimi, Alex Yakovlev. Asynchronous transient resilient links for NoC [C]// the 6th IEEE/ACM/IFIP international conference on Hardware/Software codesign and system synthesis. USA: New York, 2008: 1-6.
- [2] Muhammad Ali, Michael Welzl, Sven Hessler. A fault tolerant mechanism for handling permanent and transient failures in a network on chip [C]// the 4th international information technology conference. USA: Los Vegas, 2007: 1027-1032.
- [3] Dumitras T, Kemer S, Marculescu R. Towards on-chip fault-tolerant communication [C]// Proc. ASP-DAC. Kitakyushu, Japan: IEEE, 2003: 225-232.
- [4] Muralli S, Theocharides T, Vijaykrishnan N, et al. Analysis of error recovery schemes for networks on chips [J]. IEEE Design & Test of Computers, 2005, 22(5): 434-442.
- [5] Paul Bogdan, Tudor Dumitras, Radu Marculescu. Stochastic communication: a new paradigm for fault-tolerant networks networks-on-chip [C]// VLSI Design 2007. Bangalore, 2007: 1-17.
- [6] Muhammad Ali, Michael Welzl, Sven Hessler. An efficient fault tolerant mechanism to deal with permanent and transient failures in a network on chip [J]. High Performance System Architecture, 2007, 1(2): 113-123.
- [7] 张磊,李华伟,李晓维. 用于片上网络的容错通信算法 [J]. 计算机辅助设计与图形学学报, 2007, 4(4): 904-912.
- [8] Ahmad Patooghy, Seyed Ghassem Miremadi. XYX: a power & performance efficient fault-tolerant routing algorithm for network on chip [C]// Proceedings of the 2009 17th international Conference on Parallel, Distributed and Network-based Processing. Germany: Weimar, 2009: 245-251.
- [9] Karp R. Randomized rumor spreading [C]// Proc. of the 41th Annual Symposium on Foundations of Computer Science. San Diego, USA: IEEE Press, 2000: 565-574.
- [10] 罗莎莎. 片上网络瞬时故障容忍算法研究 [D]. 长沙: 湖南大学, 2009.

作者简介:

周 芳 女, (1979-), 硕士, 讲师. 研究方向为数字系统设计与计算机应用、专用集成电路设计. E-mail: zfnuaa@nuaa.edu.cn.

吴 宁 女, (1956-), 教授, 博士生导师. 研究方向为数字系统理论与技术、电子系统集成与专用集成电路设计.

(上接第 24 页)

- [7] 蔡宇东, 沈海斌, 严晓浪. AES 算法的高速实现 [J]. 微电子学与计算机, 2004, 21(1): 83-85.
- [8] Chen Jun Hong, Shieh Ming Der, Lin Wen Ching. A high performance unified-field reconfigurable cryptography processor [J]. IEEE Transaction on VLSI, 2010, 18(8): 1145-1158.
- [9] Chang Peichann, Huan Weihsuig. A block mining and recombination enhanced genetic algorithm for the permutation flow shop scheduling problem [J]. Int. J.

Production Economics, 2013, 141(1): 45-55.

作者简介:

常忠祥 男, (1985-), 博士研究生. 研究方向为专用集成电路设计. E-mail: changzhongxiang0@126.com.

戴紫彬 男, (1966-), 教授, 博士生导师. 研究方向为密码微系统设计.

李 伟 男, (1983-), 讲师. 研究方向为密码微系统设计.

龚海宁 男, (1980-), 工程师. 研究方向为软件工程.