面向千兆以太网的动态RDMA通信方法

李龙飞，史阳春，王剑峰，贺占庄

(西安微电子技术研究所集成电路设计部 西安 710065)

【摘要】针对RDMA传输在千兆以太网中连接建立等待时间过长造成传输效率降低的问题，提出一种动态RDMA通信方法，并进行建模和实验。该方法在保持对传统通信方法兼容的基础上，提出间接传输方法及对应的动态选择策略。通过扩展NIC的硬件逻辑并预留相应的缓存资源，动态RDMA通信方法可以在连接未建立成功的情况下进行数据传输，同时继续保持连接的建立。该过程实现了数据传输与连接建立的并行进行，且对上层协议透明。以千兆以太网NIC为原型构建端系统模型，并搭建测试平台进行实验。实验结果表明，动态RDMA通信方法以很小的额外硬件开销解决了连接建立等待时间过长的问题，提供了优于传统RDMA通信方法的传输性能。

关 键 词 动态RDMA； 千兆以太网； 硬件加速； 间接传输

中图分类号 TP393 **文献标志码** A doi:10.3969/j.issn.1001-0548.2018.05.006

Hardware Accelerated Dynamic RDMA Method for Gigabit Ethernet

LI Long-fei, SHI Yang-chun, WANG Jian-feng, and HE Zhan-zhuang

(Department of Integrated Circuit Design, Xi'an Microelectronics Technology Institute Xi'an 710065)

Abstract Waiting for connection establishment can be inefficient during remote direct memory access (RDMA) transport for Gigabit Ethernet. Aiming at solving this problem, a hardware accelerated dynamic RDMA method is proposed in this paper. This method allows starting RDMA transport at the same time with connection establishment by sending packages first to indirection buffer in network interface card (NIC) and copying them later to host memory. The terminal models of dynamic RDMA method are built and the simulation platforms for perform experiments are developed. Experimental results show that the dynamic RDMA method not only can solve the long waiting problem of connection establishment with little extra hardware cost, but also can provide higher transport performance compared with traditional RDMA methods.

Key words dynamic RDMA; Gigabit Ethernet; hardware acceleration; indirection transport

根据吉尔德定律(Gilder's law)和摩尔定律(Moore's law)，网络带宽的增长速度至少是CPU运算性能增长速度的3倍(即每6个月增长1倍)[1]。因此，近年来影响系统性能的瓶颈已经逐渐从网络带宽转变为了CPU的运算性能。减小网络数据对CPU资源的消耗是提高系统性能的有效手段。作为InfiniBand中的一项重要技术，RDMA从被提出就得到了业界的广泛关注。RDMA操作可以实现对远程应用程序内存的直接读写操作，且整个过程不需要远程节点CPU的参与，实现“零复制”操作，因此没有消耗任何CPU资源，为InfiniBand带来了传统TCP/IP完全无法实现的高速数据传输特性。

千兆以太网的普及使得RDMA不再是InfiniBand独享的技术。然而由于千兆以太网是不可靠网络，因此必须采取一定的技术手段来保证RDMA的可靠传输。InfiniBand提供硬件级的端到端可靠传输服务，于是以RoCE(RDMA over Converged Ethernet)为代表的RDMA技术提出采用硬件设备保证数据链路层的无损传输[2]。但该方案需要改变网络拓扑，代价较高，不适宜现有网络环境的改造。另一种方案是采用面向连接的传输，通过上层协议来保证RDMA传输的可靠性，如iWARP[3]。除了需要上层协议支持外，该方案还需要支持RDMA的NIC(简称为RNIC)才能实现，但比起前者，该方案在硬件方面的改动很小[4]，且对现有的网络和设备兼容，因此其逐渐成为了国内外诸多学者关注的重点和研究的方向。

在采用面向连接的传输方式中，文献[5]通过对

收稿日期：2017－05－16；修回日期：2018－05－28
基金项目：部级基金
作者简介：李龙飞(1988－)，男，博士生，主要从事计算机网络硬件加速方面的研究.

流量控制、重传控制进行优化，减小了在发生数据丢失时系统的重传开销，实现了快速RDMA重传；文献[6]提出一种轻量级的RDMA引擎，通过RNIC硬件辅助来实现RDMA技术在虚拟机中的应用；文献[7]重点关注RDMA过程中的内存注册，提出基于动态链表的注册内存池技术，减小内存注册操作的使用频率；文献[8]面向RNIC的加速设计，提出批量转发机制来提高RDMA传输在数据帧较小时的吞吐量；文献[9]针对RDMA报文乱序到达、重路由等问题，提出动态连接的解决方案。在面向连接的RDMA传输中，连接的建立需要采用握手机制实现。若传输发起端等待应答端的应答时间过长，那么RDMA所带来的性能优势就会被削弱。上述文献均未考虑过长的连接建立时间对RDMA性能造成的影响，而在实际应用中，以太网的不可靠传输所造成的链路丢帧以及内存注册失败等原因均会造成发起端的长时间等待，从而引起传输性能下降[10-11]。

针对上述问题，本文面向千兆以太网提出一种硬件支持的动态RDMA通信方法。该方法保持对传统RDMA通信方法的兼容，在此基础上提出了间接RDMA传输，即通过扩展RNIC的硬件逻辑并预留相应的缓存资源，从而使发起端在连接未建立成功的情况下进行RDMA传输。本文同时给出了直接传输、间接传输两种模式的动态切换方法并建立端系统模型进行仿真实验。与其他研究相比，本文的主要创新点如下：

1) 首次提出间接RDMA传输及相应的传输模式动态切换方法，即动态RDMA通信方法。

2) 以千兆以太网NIC为原型构建端系统模型对动态RDMA通信方法和传统RDMA通信方法的传输性能进行对比分析。

1　动态RDMA通信方法及端系统模型

动态RDMA通信方法的核心思想是根据在连接建立过程中应答端的应答状态以及发起端的请求状态，在直接传输与间接传输两种模式中进行动态选择，以避免过长的请求应答等待时间，从而降低单次RDMA传输耗时，提高传输效率。

1.1　传输模式

1.1.1　直接RDMA传输

直接传输即传统的RDMA传输，是指发起端和应答端在数据传输之前先建立连接，只有当连接建立完成后才开始RDMA传输[12]。在整个传输过程中，采用滑动窗口等确认机制对传输的可靠性进行保证。RDMA传输所需要的必要参数如表1所示。

表1　RDMA的参数化描述

参数	说明
TID	本次RDMA传输的ID号
TYPE	传输类型(读或写)
MODE	传输模式(直接或间接)
SBA	本次传输中发起端内存区域的基地址
DBA	本次传输中应答端内存区域的基地址
LEN	RDMA传输数据长度(字节)
SN	RDMA数据帧传输序列号
ACKN	RDMA数据帧传输确认号

RDMA的连接建立过程是由传输发起端的上层协议发起的，在连接建立请求中会将TID、TYPE、MODE、SBA、DBA、LEN等信息告知应答端。应答端根据DBA及LEN等信息对内存区域进行注册。若内存注册成功，则返回请求应答，称之为有效请求应答；若内存注册失败，同样会返回请求应答，发起端收到该应答后会重置超时定时器，等待重新请求，称这种应答为无效请求应答。

连接建立完成后，由发起端和应答端的RNIC负责RDMA数据的传输[13]。上层协议和CPU在连接建立后不再需要参与到RDMA的传输中。以RDMA写过程为例，传输发起端RNIC根据连接信息直接从对应的内存区域中获取数据并封装为RDMA帧发送至链路；传输应答端RNIC在接收过程中进行过滤，在识别到RDMA帧后，根据其帧头的DBA、LEN、SN等信息确定该数据在内存区域中的地址，然后直接将数据放入对应的内存区域[14]。数据传输过程中的应答确认、超时重传等可靠性机制均在RNIC内部硬件实现。当传输完成后RNIC释放连接，并给上层协议产生标志信号。

1.1.2　间接RDMA传输

由于千兆以太网的不可靠性，因此在连接建立过程中，用于请求和应答的RDMA帧有可能发生丢失；另一方面，由于传输应答端无法确定何时会收到RDMA请求，因此在收到请求后，可能存在传输需求内存区域未释放等情况，因此不能完成内存注册。上述两种情况均会引起连接建立过程耗时的增大，造成RDMA传输的等待，从而影响RDMA传输性能[15]。

间接RDMA传输，是指在连接未建立成功的状态下通过向上层协议产生模拟应答，使发起端和应答端开始RDMA数据传输，同时在数据传输的过程中完成连接建立。间接传输通过扩展RNIC硬件逻辑

和预留缓存资源，同时在传输中保持对连接建立信息的记录，使RDMA数据能最终到达指定内存区域。

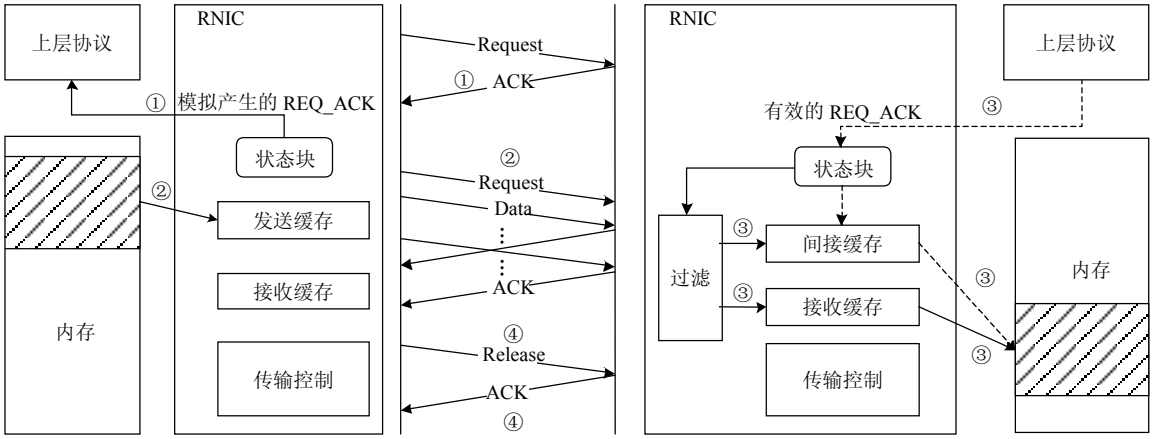

图1 间接RDMA传输示意图

结合图1对间接RDMA传输中的写操作进行说明，其传输过程可分为以下4个步骤：

1) 传输发起端的上层协议发起RDMA请求，RNIC会在未收到应答端请求应答的情况下由其自身模拟产生一个应答帧，并传递至上层协议。

2) 发起端上层协议把RDMA数据传输的控制权交至RNIC。RNIC记录此次RDMA传输的各项参数至连接状态块，并开始进行RDMA数据写操作。

3) 应答端RNIC接收来自发起端的数据，并根据实际的状态选择直接或间接处理方式。直接处理方式将数据在本地进行缓存后立即传输至内存指定区域；间接处理方式将数据缓存至RNIC内部提前预留的间接缓存中，并等待上层协议发出请求应答。当检测到请求应答已发出时，RNIC会将间接缓存中与之对应的数据存储至指定的内存区域。

4) 发起端RNIC在连接建立完成且数据传输完成时，发出RDMA连接释放请求。应答端RNIC在接收到发起端的连接释放请求后，释放该RDMA连接。

以上4个步骤中，步骤2)和3)是间接传输的关键，这里需要进一步进行说明。在步骤2)中，由于请求帧在链路传输中也存在丢帧的情况，因此发起端无法判断出应答端是否接收到请求帧。在这种情况下，RNIC在进行数据传输的过程中仍会进行请求的发送。在写操作进行的过程中，发起端会对收到的数据帧进行过滤，若发起端接收到真正的有效请求应答，则RNIC会在连接状态块中进行记录，表明真正的连接建立完成，同时将该请求应答抛弃。

在步骤3)中，应答端对链路丢帧和内存注册失败所造成的连接建立不成功的处理方式是不一样的。若应答端发出的有效请求应答丢失，则表明应答端已经完成了内存注册，因此应答端RNIC会按直接传输的方式对接收到的数据进行处理。除了这种情况外，RNIC均采用间接处理方式。这是因为对于应答端来说，无效请求应答丢失等效于内存注册失败。在这种情况下，应答端RNIC在连接状态块中查询不到对应的连接信息，只能将数据缓存至提前预留的间接缓存中。只有当RNIC在状态块中检测到上层协议产生的请求应答已发出时，间接缓存中的数据才可以被转移到指定的内存区域中。

间接传输的读操作是写操作的逆向过程，是由应答端向发起端进行数据传输。与写操作相反，应答端根据发起端的请求应答状态来决定是否发起间接传输。值得注意的是，发起端在发起读操作前，必定完成了本地内存区域的注册，因此读操作的间接传输一定是由有效请求应答丢失引起的，这意味着间接传输中的读操作不需要间接缓存的参与。读操作在传输开始后其传输机制与写操作相同，这里不再赘述。

1.2 动态选择方法及处理流程

在动态RDMA通信方法中，动态选择方法根据连接建立过程中发起端和应答端的状态，决定一次RDMA传输的传输模式。以写操作为例，当发起端收到有效请求应答后会发起直接传输；对于内存注册失败情况，在收到无效请求应答后会发起间接传输；对于链路丢帧情况，规定如果发起端在重传定时器第二次超时前仍未收到任何有效请求应答，则直接发起间接传输。对于每一个不同的RDMA传输，发起端根据上述方法动态地对传输模式进行选择。与写操作相反，读操作的传输模式由应答端进行动态选择，其方法与写操作一致。

结合上文对直接传输、间接传输的描述，动态RDMA通信方法写操作过程中发起端的伪代码描述如下：

```
send TID i request
if get REQ_ACK then
  if valid REQ_ACK then
  MODE = direct
  else MODE = indirect
  end if
else if resend_counter = 2 then
  MODE = indirect
  else
 resend TID i request
 resend_counter ++
 end if
end if
if MODE = direct then
TX buffer <= memory[SBA:SBA+LEN−1]
send data from TX buffer to link
else
generate simulate REQ_ACK to upper layer
TX buffer <= memory[SBA:SBA+LEN−1]
send data from TX buffer to link
  if get REQ_ACK and TID = i then
  update status block and abort REQ_ACK
  end if
end if
```

写操作过程中应答端的伪代码描述如下：

```
get request and add TID i to status block
if memory registration done then
send valid REQ_ACK
else
send invalid REQ_ACK
end if
if get RDMA data then
  if MODE = direct then
  receive data from link to RX buffer
 memory[DBA:DBA+LEN−1] <= RX buffer
 else
   if send valid REQ_ACK and TID = i then
   receive data from link to RX buffer
   memory[DBA:DBA+LEN−1]<= RX buffer
   else
   receive data from link to indirect buffer
   if get valid REQ_ACK and TID = i then
   memory[DBA:DBA+LEN−1]<= indirect buffer
   else
   wait until valid REQ_ACK
   end if
  end if
 end if
end if
```

1.3　端系统模型构建

端系统模型的主要功能是模拟动态RDMA通信方法中连接发起端和应答端的工作原理，以便通过实验平台对本文提出的动态RDMA通信方法进行仿真验证。端系统模型主要包括3方面内容：传输协议、系统主机以及RNIC。由于动态RDMA通信方法兼容传统RDMA通信方法，因此本节主要介绍动态RDMA通信方法中的端系统模型。

1.3.1　传输协议

基于以太网帧封装格式，模型中自定义一类动态RDMA通信报文，称之为DRP(dynamic RDMA protocol)报文，对应的传输协议为DRP协议。DRP报文格式如图2所示。

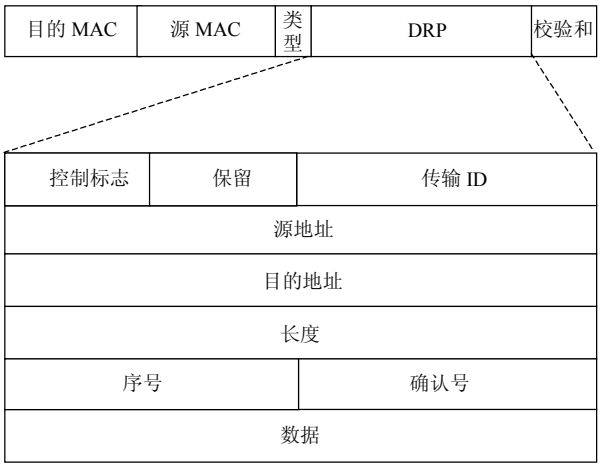

图2　DRP报文格式

定义DRP报文的类型号为0X01FF。因为以太网帧类型编号的0X0101到0X01FF区间为实验使用，因此DRP不会与其他协议类型发生冲突[16]。DRP首部中控制字段长为8位，包含DRP报文的各类标志信息，具体包括连接建立请求报文标志位、连接建立请求应答报文标志位、数据报文标志位、连接释放请求标志位、连接释放应答标志位、读写标志位以及直接、间接传输标志位。DRP首部其余字段包括

了RDMA传输所需要的基本信息，需要注意的是长度字段在连接建立过程中表示的是RDMA传输数据长度，在数据传输过程中表示该DRP报文中有效负载的长度，二者通过控制字段进行区分。

采用类TCP的窗口控制和重传控制机制对传输可靠性进行保障。考虑到硬件实现的复杂度，协议未涉及流控和拥塞控制机制。事实上，动态RDMA通信方法更关心不可靠网络中数据帧丢失对传输所造成的影响，因此这里不考虑流控和拥塞控制是合理的。窗口控制方面，模型中固定窗口大小为8，重传控制方面，因为端系统模型采用了全双工的千兆以太网NIC，意味着每秒大约有81 000个1 kB左右大小的数据帧流经网络[17]，因此数据帧传输采用了高速重传控制机制(fast retransmission)，使超时管理更有效。总之，为了降低端系统模型的复杂度，在保证可靠传输的基础上，本文尽可能精简传输协议。

1.3.2 RNIC及系统主机

BCM5718是博通公司(BroadCom)开发的一款千兆以太网控制电路，在市面上得到了广泛应用[18]。本文参考BCM5718的架构，采用system verilog语言对端系统中的RNIC(简称为动态RNIC)进行构建。动态RNIC架构如图3所示，其内部集成的PCI-E、GMAC模块采用Synopsys 公司的IP软核实现，其余模块均由自主设计。较BCM5718相比，动态RNIC主要增加了RDMA传输控制模块及间接缓存模块。RDMA传输控制模块主要对DRP协议进行处理，实现动态RDMA通信中的传输模式动态选择及可靠传输控制。接收通路中增加的间接缓存模块根据帧过滤模块的过滤结果，实现在间接传输中对数据的缓存。较动态RNIC，传统RNIC架构中不包含间接缓存模块，且RDMA传输控制模块的功能较为简单，只需要实现连接建立后的可靠传输控制。

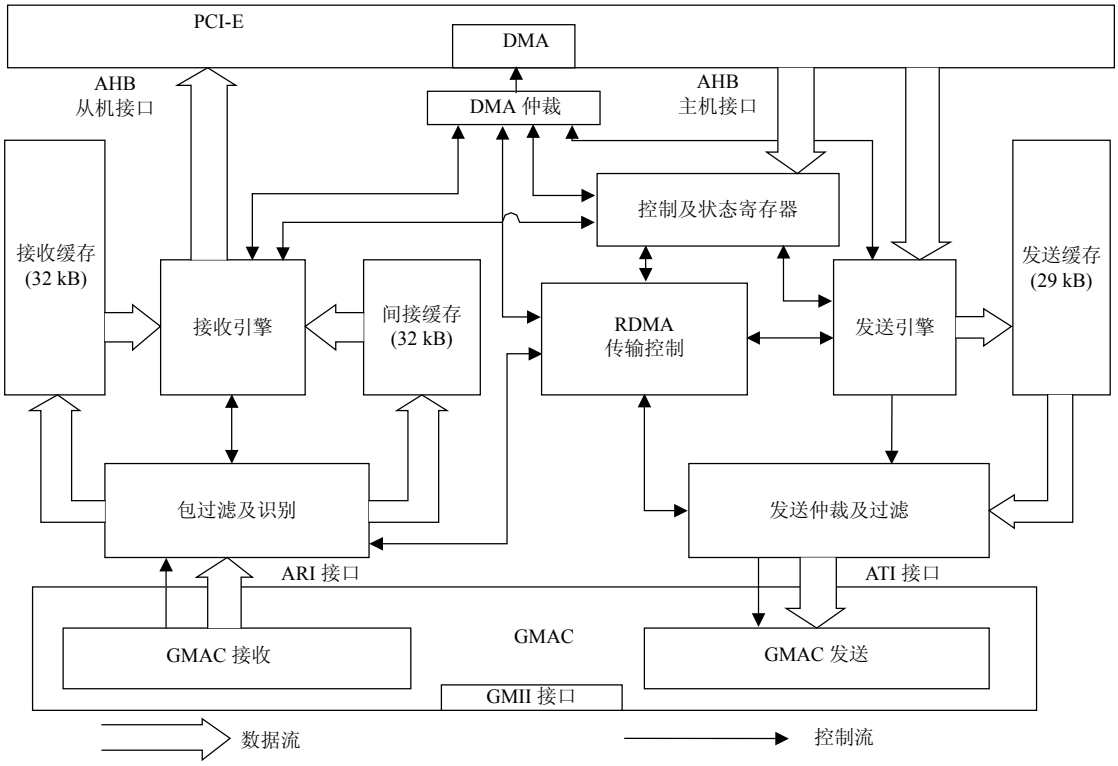

图3 动态RNIC架构

端系统模型中的主机部分为一个PCI-E根节点(root complex)，其主要功能是根据配置模拟上层协议对RDMA传输的处理，具体包括连接建立请求的发起、应答以及内存的锁定、注册等操作。即，主机部分会根据用户配置对RNIC产生相应的激励并做出相应的响应。模型中主机部分在仿真实验和实物实验中采用了不同的实现方式，将在实验章节中进行说明。

2 实验及结果

2.1 仿真实验

采用Cadence的NC-SIM搭建仿真平台对本文提出的动态RDMA通信方法和传统RDMA通信方法进行对比实验。端系统模型中的主机部分采用system verilog建模语言构建，并通过PCI-E的VIP(verification IP)与RNIC相连。仿真环境中采用点对

点的网络拓扑方式，系统时钟为125 MHz，且链路工作在千兆全双工模式。实验均以RDMA写操作为例对动态RDMA通信方法和传统RDMA通信方法进行比较。其中，每次RDMA传输的数据规模一律为128 kB，每个数据帧的有效负载均为1 024字节。各种实验环境下的传输次数均为800次，共进行5次，并取均值作为最终实验结果。所有实验数据和比较结果均基于本文提出的端系统模型获得。

为了有效验证动态RDMA通信方法的性能，针对内存注册失败及链路丢帧两大因素分别设计了相应的实验。

在网络无丢帧的环境下，比较内存注册失败对两种通信方法的影响。两种通信方法的传输耗时随内存注册失败概率FRP(failed registration probability)变化的情况如图4所示。从实验结果可以看出，在相同的实验环境，动态RDMA通信方法的传输耗时均小于传统RDMA通信方法，其主要是因为动态RDMA通信方法中的间接传输方式在内存注册失败的情况下实现了数据传输与连接建立的并行进行，从而节省了部分时间。

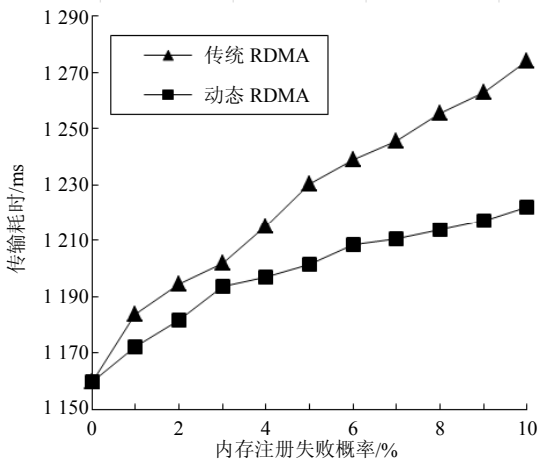

图4　内存注册失败情况下传输耗时对比

在无内存注册失败的环境下进行实验，测试链路丢帧对两种通信方法传输效率的影响。实验中链路丢帧概率LFP(lost frame probability)从0%逐步递增至10%，得到两种通信方法的传输耗时如图5所示。由图可知，在链路丢帧概率一定的情况下，动态RDMA通信方法与传统RDMA通信方法相比具有更高的传输效率。通过纵向的比较分析两组实验环境下的传输耗时，可以得到：在链路丢帧环境下，由于超时重传的原因，因此无论是动态RDMA通信方法还是传统RDMA通信方法，其传输耗时均普遍大于内存注册失败下的传输耗时。上述分析表明：在千兆以太网等不可靠网络环境下进行RDMA传输时，链路丢帧是造成传输耗时增大的主要因素。

通过对比两组实验环境下的实验结果还可以得到，内存注册失败情况下动态与传统RDMA传输耗时的差距比链路丢帧情况下的传输耗时的差距数值更大。这表明在内存注册失败的情况下，动态RDMA通信方法的效果更明显。这是因为内存注册失败只会发生在连接建立的过程中，而链路丢帧不仅会发生在连接建立的过程中，还会发生在RDMA数据传输的过程中。然而动态RDMA通信方法只能解决连接建立过程中的内存注册失败和链路丢帧问题，无法对数据传输过程中的丢帧现象进行处理，因此动态RDMA通信方法针对内存注册失败问题表现出了更佳的效果。

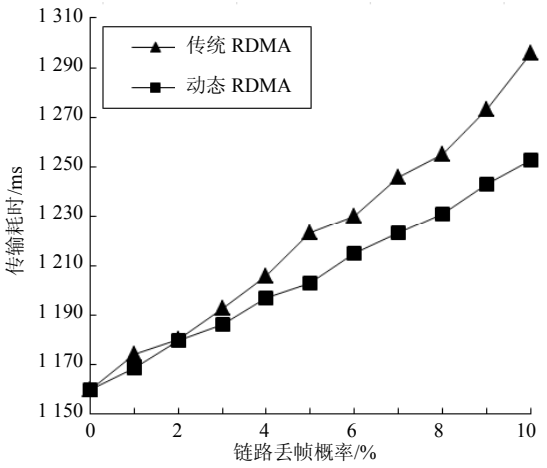

图5　链路丢帧情况下传输耗时对比

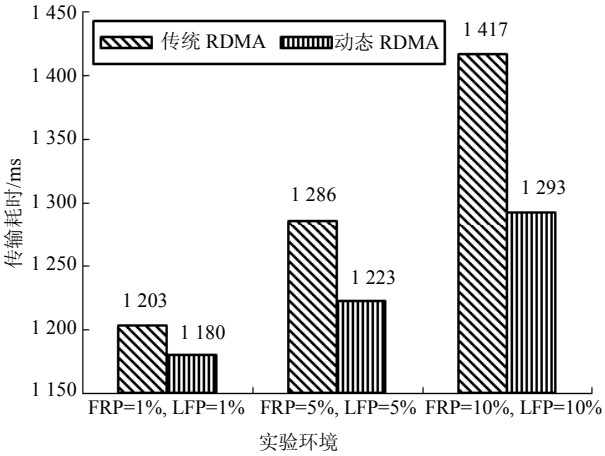

图6　内存注册失败及链路丢帧共同作用下传输耗时对比

为了进一步对比与分析两种传输方法对传输效率的影响，在内存注册失败和链路丢帧共同作用的环境下进行实验。实验中选取FRP和LFP分别同时为

1%，5%，10%的3种情况进行，其他实验环境不变。两种通信方法的传输耗时如图6所示。显然，动态RDMA通信方法在更短的时间内完成了数据的传输。在3种实验环境下，动态RDMA通信方法的传输时间较传统RDMA通信方法分别降低了1.9%，4.9%和8.8%。通过对实验数据的进一步挖掘可以得到，在网络环境较差或端系统内存资源紧张的情况下，动态RDMA通信方法的性能优势愈发明显。

2.2 实物实验

基于Stratix V FPGA(Device:5SGSMD4E2H29I3)测试板分别实现了动态RNIC和传统RNIC，然后在对应的背板上实现了端系统模型。背板中搭载Freescale公司的MPC5200B嵌入式处理器，运行Wind River公司的VxWorks操作系统，以实现端系统模型中的主机功能，并通过PCI-E插槽与FPGA测试板相连。实验中使用两组完全相同的端系统模型，采用点对点的网络拓扑方式直连，且链路工作在千兆全双工模式。

对动态RNIC的硬件开销进行分析。采用Quartus II分别对端系统模型中的动态RNIC和传统RNIC进行综合，得到硬件资源开销如表2所示。由于增加传输控制逻辑以及提前预留间接缓存，因此动态RNIC的硬件开销大于传统的RNIC。然而，由于直接传输与间接传输在数据传输中复用了相同的传输控制模块，因此在硬件逻辑使用率方面，动态RNIC仅比传统RNIC增加了3%；在存储资源使用率方面，动态RNIC较传统RNIC增加了2%。其他硬件开销方面，如PLL、高速及通用I/O接口等，二者保持一致。综上，动态RNIC的实现并未占用大量硬件资源，同时在接口和时钟方面与传统RNIC保持了良好的兼容性。

在实物环境下对两种传输方法的传输效率进行对比。实验中仍以RDMA写操作为例进行数据传输，每次RDMA传输的数据规模一律为128 kB，且进行800次传输。为了模拟在实际应用中其他任务对系统内存和链路的使用情况，在实验过程中端系统主机并行执行3个ping命令，向对端主机持续发送大小分别为256、512和1 024字节的数据帧。实验环境搭建完成后，在RDMA传输数据帧大小分别为256、512和1 024字节3种情况下进行实验。每组实验进行5次，并取均值作为最终实验结果，从而得到两种传输方法的平均传输耗时如图7所示。

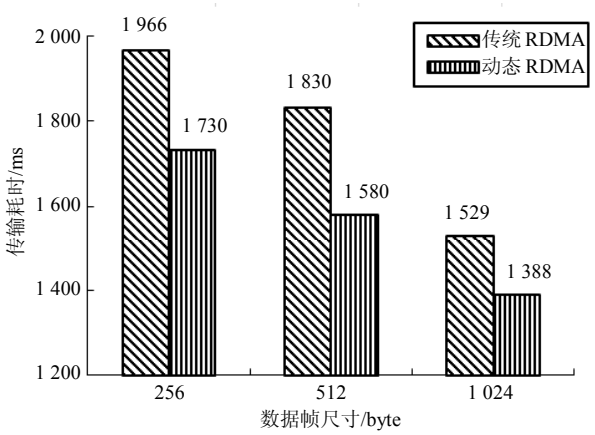

图7 实物环境下传输耗时对比

从实验结果来看，在实物环境下，无论RDMA数据帧为多大，动态RDMA通信方法的传输耗时仍然较传统RDMA通信方法要小，且传输耗时分别降低了12.0%，13.6%和9.2%，平均降低了11.6%。在之前的仿真实验中只考虑了内存注册失败和链路丢帧这两个因素对传输耗时的影响，而在实物实验中，进一步考虑了数据帧尺寸这一因素，同时端系统模型中从主机到链路这一数据通路更加贴近实际应用，逻辑和物理延迟均得到体现，因此该组实验结果是各方面因素共同作用下的传输耗时，其数值普遍大于仿真实验中的传输耗时是合理的。

表2 两种方法中RNIC硬件开销对比

Stratix V	传统RDMA			动态RDMA		
5SGSMD4E2H29I3	使用个数	总数	使用率/%	使用个数	总数	使用率/%
ALMs	106 754	135 840	79	111 846	135 840	82
RAM Blocks	288	957	30	302	957	32
Block memory bits	3 323 924	19 599 360	17	3 717 140	19 599 360	19
Pins	247	416	59	247	416	59
PLLs	4	40	10	4	40	10

另外，之所以在数据帧尺寸较小时，动态RDMA通信方法的传输耗时降低更为显著，这是因为动态RDMA通信方法的间接传输方式会将接收到的数据帧统一进行缓存，并发起一次PCI-E请求完成数据到主机内存的传输；而在数据帧尺寸较小时，当采用直接传输方式时，RNIC会发起较多次数的PCI-E请求，造成传输耗时的增大，从而使动态RDMA通信方法的优势更加明显。横向比较3组实验数据也可以

看出，两种传输方法的传输耗时均随着数据帧尺寸的增大而减小，其也是由于数据帧尺寸较小时，相对较多的PCI-E请求造成的。

3　结 束 语

本文提出了一种面向千兆以太网的动态RDMA通信方法，通过在RNIC中预留资源，使RDMA数据传输与连接建立同步进行，克服了在连接建立过程中等待应答时间过长从而造成传输效率降低的问题。在千兆以太网环境下构建模型并进行实验，结果表明动态RDMA通信方法在不可靠网络环境下可以有效提高数据传输效率。

下一步的研究工作将针对具体的上层协议，通过对其进行修改和优化，将动态RDMA通信方法应用到实际网络环境中，并对其性能进行实验和分析。

参 考 文 献

[1] BROWN D. Are new approaches needed for developing long-term strategies for STEM information?[J]. Learned Publishing, 2017, 30(3): 21-28.

[2] KAGAN M. Performance evaluation of the RDMA over ethernet (RoCE) standard in enterprise data centers infrastructure[C]//The 23rd International Teletraffic Congress. San Francisco, USA: ACM, 2011: 9-15.

[3] GRANT R E, RASHTI M J, AFSAHI A, et al. RDMA capable iWARP over datagrams[C]//Parallel & Distributed Processing Symposium. Boston, USA: IEEE, 2013: 628-639.

[4] GUO C, WU H, SONI G, et al. RDMA over commodity Ethernet at scale[C]//Conference on ACM SIGCOMM. Florianópolis, Brazil: ACM, 2016: 202-215.

[5] WANG Shao-gang, XU Wei-xia, WU Dan, et al. Fast NIC based RDMA implementation for adaptive unreliable networks[C]//The 11th International Conference on Computer Systems and Applications (AICCSA). Ifrane, Morocco: IEEE, 2014: 302-309.

[6] MOUZAKITIS A, PINTO C, NIKOLAEV N, et al. Lightweight and generic RDMA engine para-virtualization for the KVM hypervisor[C]//International Conference on High PERFORMANCE Computing & Simulation. Genoa, Italy: IEEE, 2017: 737-744.

[7] 董勇, 周恩强, 卢宇彤, 等. 基于天河2高速互连网络实现混合层次文件系统H~2FS高速通信[J]. 计算机学报, 2017, 40(9): 1961-1979.
DONG Yong, ZHOU En-qiang, LU Yu-tong, et al. The implementation of communicating operation in hybrid hierarchy file system H2FS with TH-Express 2[J]. Chinese Journal of Computers, 2017, 40(9): 1961-1979.

[8] MA S, KIM J, MOON S. Exploring low-latency interconnect for scaling out software routers[C]//IEEE International Workshop on High-Performance Interconnection Networks in the Exascale and Big-Data Era. Barcelona, Spain: IEEE, 2016: 9-15.

[9] 夏军, 庞征斌, 刘路, 等. 一种基于NIC的RDMA可靠传输协议的设计与实现[J]. 计算机工程与科学, 2014, 36(2): 216-221.
XIA Jun, PANG zhen-bin, LIU Lu, et al. Design and implementation of a NIC based RDMA reliable communication protocol[J]. Computer Engineering and Science, 2014, 36(2): 216-221.

[10] FREY P W, ALONSO G. Minimizing the hidden cost of RDMA[C]//The 29th IEEE International Conference on Distributed Computing Systems. Montreal, Canada: IEEE, 2009: 553-560.

[11] MAC A P, RUSSELL R D. A performance study to guide RDMA programming decisions[C]//The 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems (HPCC-ICESS). Liverpool, United Kingdom: IEEE, 2012: 778-785.

[12] SUBRAMONI H, LAI P, LUO M, et al. RDMA over Ethernet—a preliminary study[C]//IEEE International Conference on Cluster Computing and Workshops. New Orleans, USA: IEEE, 2009: 1-9.

[13]ZHANG W, HAO M, XU Z. Communication optimization for RDMA-based science data transmission tools[J]. Journal of Super Computing, 2016, 72(9): 3312-3327.

[14] MAC A P, RUSSELL R D. An efficient method for stream semantics over RDMA[C]//The 28th International Parallel and Distributed Processing Symposium. Phoenix, USA: IEEE, 2014: 841-851.

[15] 苏文, 章隆兵, 高翔, 等. 基于Cache锁和直接缓存访问的网络处理优化方法[J]. 计算机研究与发展, 2014, 51(3): 681-690.
SU Wen, ZHANG Long-bing, GAO Xiang, et al. A cache locking and direct cache access based network processing optimization method[J]. Journal of Computer Research and Development, 2014, 51(3): 681-690.

[16] ANDREW S T, DAVID J W. Computer networks[M]. New Jersey, USA: Pearson, 2011.

[17] JIN H W, NARRAVULA S, BROWN G, et al. Performance evaluation of rdma over ip: a case study with the ammasso gigabit ethernetnic[C]//The 14th IEEE International Symposium on High Performance Distributed Computing (HPDC-14). Virginia, USA: IEEE, 2005: 598-605.

[18] LI Long-fei, HE Zhan-zhuang, WANG Jian-feng, et al. Implementation of gigabit ethernet controller with fault tolerance and prevention mechanism[C]//2017 Prognostics and System Health Management Conference (PHM-Harbin). Harbin, China: IEEE, 2017: 1-8.

编　辑　刘飞阳