

基于 FPGA 的 ROHC 硬件解压器设计

孙宇豪, 张盛兵

(西北工业大学 计算机学院, 陕西 西安 710072)

摘 要: 鲁棒性头压缩算法 (ROHC) 是一种工作在无线链接环境下, 并旨在同时提供具有高压压缩效率和高可靠性的压缩方案. 介绍了 RHOC 硬件解压器的实现框架, 并详细给出了 ROHC 单向工作模式下硬件解压器的 FPGA 实现. 运行结果显示 ROHC 报头压缩算法可以将 40 字节 RTP/IPV4/UDP 数据包压缩至到 3 ~ 4 个字节并在解压缩端实现有效解压缩.

关键词: ROHC 协议; 分组头压缩; 硬件加速器; 解压器

中图分类号: TN919.5

文献标识码: A

文章编号: 1000-7180(2013)08-0112-04

FPGA-based ROHC Hardware Decompressor Design

SUN Yu-hao, ZHANG Sheng-bing

(College of Computer, Northwestern Polytechnical University, Xi'an 710072, China)

Abstract: Robust Header Compression (ROHC) is a compression scheme aims at providing a compression scheme that has high compression efficiency and high robustness when used wireless links. In this paper, we introduce the basic concept and technology of RHOC and give the architecture of RHOC hardware decompressor which works under U mode. The decompressor is implemented on FPGA. The design of the decompressor is present in this essay also. It can work effectively, and decompress the packet such as IR, IR-DYN and UO-0/UO-1/UO-2 correctly. The performance of ROHC hardware accelerator which includes compressor and decompressor is evaluated through simulation. The result shows that the ROHC hardware compressor can compress the 40 bytes IPV4/UDP/RTP packet to 3~4 bytes, and decompress effectively by the decompressor.

Key words: ROHC; header compression; hardware accelerator; decompressor

1 引言

由于 TCP/IP 框架基于每层协议的封装, 相当一部分无线带宽用来传送控制信息 (即协议分组头), 而这些控制信息对于终端用户来说是无用的, 导致了无线信道带宽的利用率较低. 分组头会引入冗余, 该冗余比整个数据分组的 50% 还多^[1], 这样不仅浪费带宽, 而且增加了分组因为出错而被丢弃的概率. 为了提高带宽的利用率, 解决无线链路上传输实时业务带来的服务质量和分组头开销方面的问题, 就必须对分组头进行压缩.

ROHC (Robust Header Compression) 分组头压缩协议针对无线链路环境的特点, 实现了稳健高

效的分组头压缩. 国内外对 ROHC 分组头压缩协议的研究主要是以下几方面:

- (1) 将 ROHC 应用于第三代或第四代移动通信系统中, 提高数据传输的可靠性^[2];
- (2) 通过仿真给出 ROHC 的一些关键参数的优化值, 平衡 ROHC 协议的压缩率和鲁棒性, 从而使 ROHC 性能得到优化^[3];
- (3) 在 Linux 操作系统中嵌入 ROHC 报头压缩协议^[4], 方便提供分组头压缩功能;
- (4) 用硬件实现 ROHC 协议中的 CRC 计算, 帮助解决 ROHC 协议的硬件解决方案^[5].

本文给出了 ROHC 解压器在 U 模式下的系统框架, 在此系统框架下使用硬件描述语言 Verilog

收稿日期: 2012-10-30; 修回日期: 2012-12-25

实现硬件解压缩器的功能,使用解压缩算法对 IR, IR-DYN, UO-0/UO-1/UO-2 等压缩报头进行解压缩.最后,在 Altera 平台上完成 ROHC 硬件解压器的功能验证,并通过实验对压缩端的压缩效率进行了分析.

2 ROHC 压缩协议

与大部分压缩标准仅具有两种压缩状态不同,ROHC 在发送未压缩包和发送完全压缩包状态之间,又多了一个过渡状态.过渡状态增加了压缩算法的灵活性以及压缩的效率.与 ROHC 压缩算法相对应,ROHC 解压缩算法也具有三个不同的状态.压缩端和解压缩端都从较低的状态开始工作,并逐渐向高状态转移;但是压缩端和解压缩端的实时状态并不需要一一对应.一般情况下,除非发生连续性的解压缩错误,解压缩端会保持在高级状态.

解压缩端的三个状态分为为:无上下文状态(No Context, NC),静态上下文状态(Static Context, SC)和全上下文状态(Full Context, FC).NC 状态是解压端的初始化状态,该状态下解压端还没有任何上下文信息,此时只能解压含有全部上下文的报文(IR 包).在 SC 状态下,解压器已经获得了上下文信息中的静态部分,此时除了 IR 包,解压器还可以解压含有动态上下文的报文(IR-DYN 包).而在 FC 状态下,解压器已经获得了整个上下文的信息,因此可以解压任何类型的报文.

解压缩端的状态转移策略如图 1 所示.解压缩端初始化时,工作在 NC 状态.当工作在 NC 状态的时候,解压缩端可能还无法成功的解压压缩分组,如 IR-DYN 或者 UO 类包.但解压缩端一旦正确解压了一个分组(比如,当收到带有静、动态信息的初始化分组时),解压方将转移到 FC 状态,在 FC 状态时只有当发生重复的失败时,才转回到较低状态.而且,这种情况发生时,它首先要转回到 SC 状态,此时,收到的来自 FC 态的任何一个分组的成功解压都足以使解压缩端的状态再次转移到 FC 状态.仅当收到收到压缩端 FC 状态下发送的许多分组在 SC 下无法成功解压的时候,解压缩端才会转回到 NC 状态.

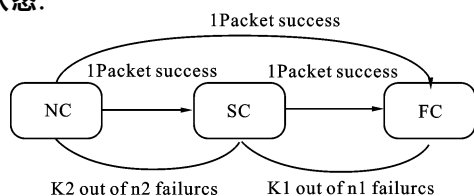


图 1 解压缩端的状态转移逻辑^[6]

3 硬件加速器解压缩端的结构

如图 2 所示,ROHC 解压缩端工作在数据链路层和网络层之间.解压缩端由四部分组成:数据包输入模块(Packet Input Module, PIM),数据包解压缩模块(Packet Decompression Module, PDM),数据包提交模块(Packet Submittal Module, PSM)和上下文存储模块(Context Storage Module CSM).

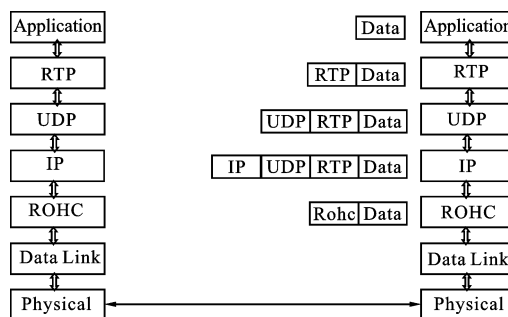


图 2 ROHC 协议在 IP 协议栈中的位置

为了使模块可以并行执行,在各个模块之间均设计了 FIFO: 压缩报头缓冲队列(Compression Header FIFO, CHF), 载荷缓冲队列(Payload FIFO, PLF), 解压缩报头缓冲队列(Decompression Header FIFO, DHF). 各个模块详细描述如下,设计示意图如图 3 所示.

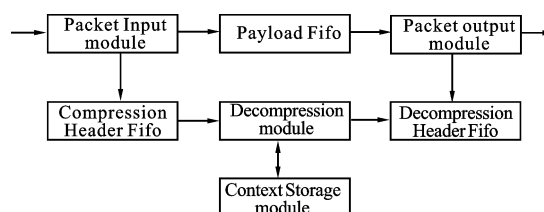


图 3 ROHC 解压缩端架构设计示意图

数据包输入模块: PIM 主要负责解压缩端与数据链路层的交互. 当一个数据包被数据链路层完整接收后, 链路层会将数据包提交给 PIM. PIM 会对数据包进行解析, 并分离出数据报头和数据载荷, 然后将数据报头和数据载荷分别放入 CHF 和 PLF.

数据包解压缩模块: PDM 是压缩端的主要部分, 负责压缩报文报头(如 IR, IR-DYN, UO-*)的解压缩. PDM 主要有一个有限状态机及相应的解压缩逻辑组成. 在 PDM 中, 不同的报文压缩类型, 对应着不同的解压缩算法及解压缩逻辑, 下节将阐述详细的 PDM 设计方案.

数据包提交模块: PSM 主要负责 ROHC 压缩端与网络层的交互. 当一个报头被成功解压缩之后, PDM 会将它放入 DHF 中. 当 PSM 检测到 DHF 非

空时,就会从 DHF 中取出被成功解压缩后的数据报头,同时从 PLF 中取出对应的报文载荷,并将两者重新封装成报文. 如果一个报头解压缩失败,PSM 会将解压缩失败的报头以及对应的 PLF 中的载荷丢弃.

上下文存储模块:CSM 有 RAM 模块及对应的控制逻辑组成. CSM 存储了与报文流对应的上下文内容. ROHC 的解压缩端会同时接收到几种不同的报文流,CSM 会将这些报文流分别存放,并使用 CID(Context ID Number)对不同报文流进行索引. 报文流之间使用一个五元组进行区分:源 IP 地址,目的 IP 地址,源端口,目的端口,SSRC.

4 数据包解压缩模块的实现

数据包解压缩模块有一个状态机和对应的解压缩逻辑实现. 如图 4 所示.

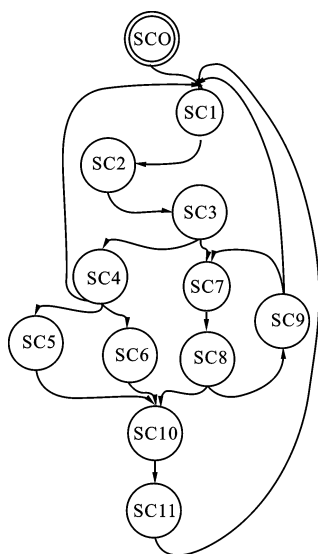


图 4 ROHC 数据包解压缩模块有限状态机

SC0: 在 SC0 状态,解压缩端完成解压缩的初始化,包括寄存器,计数器,状态位等的复位操作.

SC1: 在 SC1 状态,如果解压缩模块检测到 CHF 非空,并从中取得压缩后的数据包报头,则 FSM 转移到 SC2,否则,CHF 为空,FSM 将保持在 SC1 状态.

SC2: 在 SC2 状态,解压缩模块读取数据包报头并提取中报头的 CID. 解压缩模块使用 CID 索引 CSM,并取出对应的上下文.

SC3: 在 SC3 状态,解压缩端首先检查压缩包的类型在当前解压缩状态(NC/SC/FC)下是不是合法的. 例如,按照 ROHC 协议的规定,在 NC 状态下 IR-DYN 包是不合法的,因为在 NC 状态下解压缩

端并没有足够的信息来解压缩 IR-DYN 包. 如果当前数据包格式非法,解压缩模块将会丢弃该报头,并将错误标志位置位. 该标志位将被用来指示是否将 PLF 中对应的载荷丢弃. 如果当前数据包格式合法则根据数据包的类型转移到对应的状态: IR/IR-DYN(SC4), UO-*(SC7).

SC4/SC8: IR/IR-DYN 类压缩包在压缩后计算得出,因此这两类压缩报头的 CRC 要在解压缩之前进行校验,SC4 状态完成解压缩前的 CRC 校验. 对应的 UO-*(SC7)类压缩包在压缩前就进行 CRC 校验,因此 UO-*(SC7)类压缩包的 CRC 校验在解压缩后进行,由 SC8 负责.

SC5/SC6/SC7: SC5、SC6、SC7 分别对应 IR、IR-DYN 和 UO-*(SC7)类压缩包的解压缩.

SC9: 如果 UO-*(SC7)类压缩包的 CRC 校验失败,解压缩模块将尝试使用 W-LSB 算法对解压缩参数进行修复. 如果修复失败的话,FSM 将会忽略数据包包头,置状态位为解压缩错误,然后转移到 SC1.

SC10: 根据解压缩的结果,更新各个寄存器,计数器等.

SC11: 生成解压缩后的上下文,并写会到 CSM.

5 验证及性能分析

验证平台由五部分构成:虚拟网络层,压缩端,数据统计模块,解压缩端,虚拟链路层. 虚拟网络层主要负责模拟网络层的行为,如:产生报文流,接收解压缩后的报文等. 虚拟链路层主要负责模拟链路层以及物理层的行为. 它从压缩端接收数据,并将数据提交到解压缩端. 数据统计模块主要完成已发送数据包以及成功接收数据包的统计见图 5.

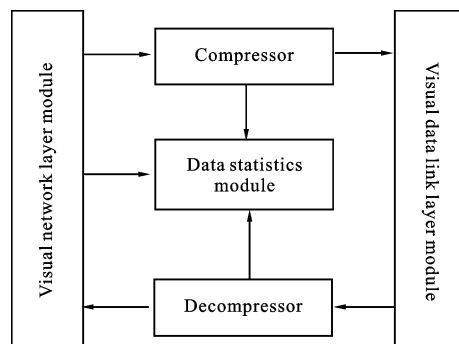


图 5 验证平台系统架构

所有以上模块均使用 Verilog 编写,并在 Altera Stratix IV 平台实现. 并用链路模拟模块模拟无线链路的误码率等问题. 布局布线结果显示,

Stratix IV 可以工作在 250 MHz,并且仅使用了 3% 的资源。

在验证中,仅 IPV4/UDP/RTP 报文在验证过程中被使用。验证使用 Quartus II 的 SignalTap 工具抓取数据统计模块中的各个寄存器值,来获取仿真结果。仿真结果如图 6 所示($n1/n2$ 和 Timeout 均为压缩端的参数)。

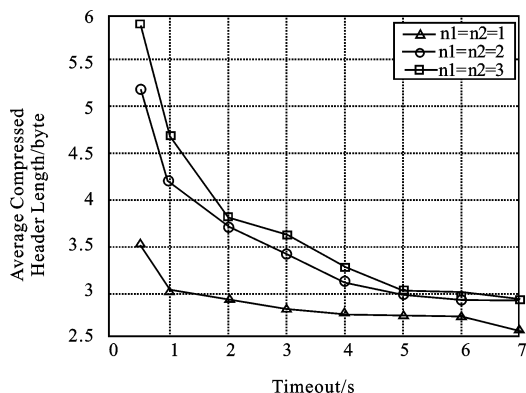


图 6 数据链路无误码情况下仿真结果

我们同时测试了在存在误码的情况下 ROHC 的性能。在最理想的情况下,使用 UOR-2 类压缩包,ROHC 可以将 40 个字节的 IPV4/UDP/IP 报头压缩到 2~4 个字节。在实现中,ROHC 可以将一个报文流的平均字节长度压缩到 3~5 个字节左右。有效的降低了对带宽的需求。无线链路的误码率一般为 $1E-3$,由图 7 可以看出在这种情况下,ROHC 仍能将 IPV4/UDP/IP 报头压缩到 5.5 个字节左右,大大提高了无线带宽利用率。

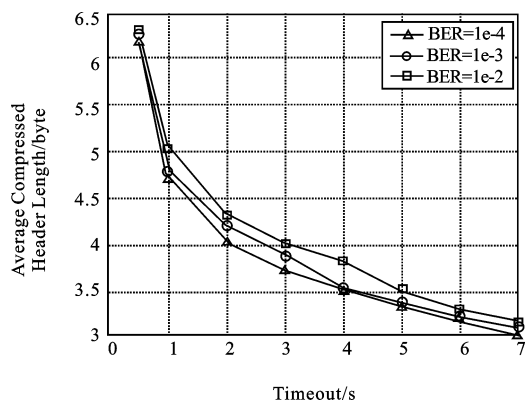


图 7 数据链路有误差情况下仿真结果

6 结束语

本文介绍了 ROHC 压缩算法的解压缩的基本原理,并且给出来了解压缩端的硬件设计框架,在此框架下针对 IPV4/UDP/RTP 分组头实现了硬件加速器的解压缩功能。通过仿真,以及 FPGA 板上测试,对 ROHC 的压缩效率进行了验证和分析。

通过实验可以验证:基于 ROHC 的硬件加速器可以将 40 字节的 IPV4/UDP/RTP 分组头压缩到平均 2.5~3.5 个字节左右,极大的减小了分组头的开销,提高了链路有效利用率。

参考文献:

- [1] 张载龙,张顺颐,陈春玲,等. 无线因特网中信头压缩技术研究 [J]. 重庆邮电大学学报, 2008, 28(6): 23-24.
- [2] 李小文,胡现岭. 3G 系统中 PDCP 子层 ROHC 的研究和实现 [J]. 重庆邮电大学学报, 2010, 22(2): 174-175.
- [3] 张凤翔,洪佩琳,李津生. 应用于 ROHC 的窗基 LSB 编码修正算法实现机制 [J]. 电路与系统学报, 2004, 9(1): 153-154.
- [4] 周逻辑. 一种在 Linux 系统中实现 ROHC 报头压缩的方案 [EB/OL]. [2012-10-30]. <http://www.paper.edu.cn>. 2008.
- [5] 翁维,刘世培. ROHC 协议中 CRC 计算的硬件实现 [J]. 电子技术, 2011, 38(6): 19-20.
- [6] Bormann C, Burmeister C, Degermark M, et al. Robust header compression (ROHC): framework and four profiles: RTP, UDP, ESP and Uncompressed. RFC 3095[S]. 2001.

作者简介:

孙宇豪 男, (1989-), 硕士. 研究方向为微处理器设计。

张盛兵 男, (1968-), 教授, 博士生导师. 研究方向为计算机系统结构。