

SpaceWire 总线的流量控制机制研究

赵云富,吴一帆,孙 强,许 娜,吴 军

(北京控制工程研究所,北京 100190)

摘 要: 为了解决 SpaceWire 总线在进行大数据块收发时,现有的流量控制措施会导致总线性能下降的问题,提出了一种两级存储结构.介绍了 SpaceWire 总线目前的应用状况,以及存在的问题;通过分析 SpaceWire 的流量控制机制以及影响总线效率、系统性能的因素,提出了一种能够保证第一级存储器有足够的空间用于收发数据的两级存储结构;根据 SpaceWire 总线协议和两级存储结构,设计了一款通过硬件预取技术实现大数据块传输的 SpaceWire 节点 IP 核;最后在 Leon3-FT 处理器上集成了该 IP 核,并进行了系统级的仿真和测试.结果表明,与现有的 SpaceWire 接口相比,本设计在大数据块传输中能够提高总线效率 62.7 倍,系统处理效率提高至少 14.5 倍.

关键词: SpaceWire 总线;流量控制机制;大数据块;两级存储器;硬件预取

中图分类号: TP393.03

文献标识码: A

文章编号: 1000-7180(2016)01-0001-05

Research on the Flow Control Mechanism of SpaceWire

ZHAO Yun-fu, WU Yi-fan, SUN Qiang, XU Na, WU Jun

(Beijing Institute of Control Engineering, Beijing 100190, China)

Abstract: To solve the problem of SpaceWire performance degradation, which is caused by the current SpaceWire flow control mechanism when SpaceWire interface transfers the bulk data, a Two-Level memory structure is proposed. The current applications and the existing problems about SpaceWire are introduced. By analyzing the flow control mechanism of the SpaceWire and the factors impacting the bus efficiency and the whole system performance, a Two-Level memory structure, which ensures the first level memory have sufficient space to receive or transfer the bulk data, is proposed. Based on SpaceWire protocol and the Two-Level memory structure, a SpaceWire node IP Core is designed, which can effectively control bulk data transfer through hardware prefetching. Finally, while the IP Core is integrated with the LEON3-FT processor, system level simulation and testing are carried out. The results show that, compared with the conventional SpaceWire node, this design can improve 62.7 times in bus efficiency, and at least 14.5 times in system processing data efficiency when it transfers the bulk data.

Key words: SpaceWire; flow control mechanism; bulk data; two-level memory; hardware prefetching

1 引言

SpaceWire 总线是一种面向航天应用的高速、全双工的串行总线标准.它兼顾了 1394 技术、ATM 技术和以太网技术的优点,同时考虑了空间应用的特点,除具有很好的 EMC(电磁兼容)特性之外,在错误检测、异常处理、故障保护和故障恢复及时间确定性方面做了相应加强.近年来,SpaceWire 总线系统在国内外的多个航天任务中得到了应用^[1-2].

SpaceWire 系统由节点、链路、路由组成.节点是数据包的信源和信宿,它可以是任意的应用单元,包括传感器、大容量存储器、数据处理单元、遥测子系统、电地设备等,是 SpaceWire 总线系统的基本单元.在节点中由接口控制器负责完成 SpaceWire 协议处理.链路是用于连接节点的路径,它的信号层采用 LVDS 协议.路由是组建 SpaceWire 网络的重要单元,负责完成数据包的接收、寻址、转发等工作.SpaceWire 传输的数据以包为单位,数据包由包头、

载荷、包尾组成。其中包头由一个或者多个目的地址组成,目的地址用于标识目的节点,以实现数据包的正确寻址。目前针对 SpaceWire 总线研究多集中在节点控制器、路由器的设计上,针对总线的流量控制研究比较少。如果 SpaceWire 总线的流量控制处理不当,将严重影响该总线在高速、大数据块传输领域的应用前景。SpaceWire 总线协议规定的流量控制机制将用于收发数据的 FIFO 限制在 56 个字节以内^[3],如果节点不能及时释放 FIFO 空间,将造成总线上传输大量的 NULLs,影响总线的效率。为此,本文提出了两级存储结构,能够实现大数据块的自动处理,有效提高总线系统的处理效率和总线传输效率。

2 SpaceWire 总线流量控制

2.1 SpaceWire 总线介绍

ECSS-E-ST-50-12C 中针对 SpaceWire 节点定义了物理层、信号层、字符层以及交换层的内容^[3]。物理层采用航天专用的微型 9 针 D 型接插件,差分输入、输出,电缆由四对双绞线组成,每对单独屏蔽,电缆长度最大 10 m。信号层采用 LVDS 进行传输,传输速率在 2~400 Mb/s,目前实际应用的最大传输速率为 200 Mb/s。字符层采用 DS 编码,数据信号 D 直接输出,而 S 值的输出规则是每当 D 值相邻量保持不变时,S 值翻转。采用这种编码后,在接收端将 D 与 S 异或后可产生接收时钟,有利于消除数据传输的影响。交换层是 SpaceWire 协议控制器的核心层,协议状态机负责链路之间连接建立、数据流量控制、错误检测及恢复等。

2.2 SpaceWire 总线流量控制

SpaceWire 总线为了防止接收缓冲区溢出,通过控制字符 FCT 实现 SpaceWire 节点的流量控制^[3]。每一个 FCT 字符代表接收端至少有 8 个 N-Char 的可用缓存空间,并且已经准备好接收数据。内部使用信用计数器(Credit-Counter)用来维护它所希望接收的 N-Char 数量,发送一个 FCT,计数器加 8,接收到一个 N-Char,计数器减 1,协议规定该计数器的上限值为 56^[3],对应 7 个 FCT 字符。当计数器大于 56 时,就会发生信用错误;当信用计数器值为 0 时,表明接收 FIFO 中的 N-Char 已经几乎满额,剩余的空间不足以接收 8 个 N-Char。

此时为了维持链路的正常连接,接口控制器开始发送 NULLs。流量控制是对总线带宽、吞吐率、传输时延以及缓存容量等一系列矛盾的简单管理,解决了数据丢失问题,却影响了总线的效率。

2.3 总线效率与系统处理效率

定义 1 总线效率是指传输一组固定数据时,有效数据的传输时间 T_m 占总传输时间 T_0 的百分比,如式(1)所示。

$$\eta = \frac{T_m}{T_0} \times 100\% \quad (1)$$

由协议可知,SpaceWire 系统中传输的有效数据为 N-Char 和 Time-Code,其他数据为控制字符 FCTs 和空字符 NULLs,因此 T_m 和 T_0 如式(2)、式(3)所示。

$$T_m = T_{N-Chars} + T_{Time-Code} \quad (2)$$

$$T_0 = T_{N-Chars} + T_{Time-Code} + T_{NULLs} + T_{FCTs} \quad (3)$$

由 SpaceWire 协议可知,FCTs 在总线上传输的所消耗的时间远远小于有效字符和空字符,为了简化计算,忽略 T_{FCTs} ,因此将式(2)、(3)代入式(1)得

$$\eta = \frac{T_m}{T_m + T_{NULLs}} \times 100\% \quad (4)$$

定义 2 系统处理效率是指 SpaceWire 总线系统接收或发送一组数据所消耗的理想时间 T_{ideal} 占实际所消耗时间 T_{real} 的百分比,如式(5)所示。

$$\eta = \frac{T_{ideal}}{T_{real}} \times 100\% \quad (5)$$

系统处理效率与处理方式相关,本文以中断处理方式为例来说明。假设系统在接收一组数据时进入中断的次数 n ,其中进、出中断所消耗的时间 T_{int} ,在中断处理程序中读取数据的时间 T_{read} ,假设每次读取相同数量的数据,则系统处理效率如式(6)所示。

$$\eta = \frac{\sum_{i=1}^n T_{read}}{\sum_{i=1}^n (T_{int} + T_{read})} \times 100\% \quad (6)$$

由总线流量控制机制可知,如果 SpaceWire 节点不能够及时释放缓冲区的空间,将影响有效数据的传输,造成总线效率的下降,如果节点频繁进入中断读取小量的数据,将影响总线系统的处理效率,由此可见,如何自主释放缓冲区的空间以及减少节点的干预程度是解决该问题的有效方法。通过分析式(4)和式(6)以及 SpaceWire 总线工作机制,本文设

计了一种两级存储结构,第一级存储单元是小容量的 FIFO,实现发送、接收数据缓存;第二级存储单元为大容量的存储器.通过两级存储结构的快速调度机制,保证 SpaceWire 第一级存储器中有足够的空间用于接收或发送有效数据(N-Char、Time-Code),这样能够减少总线上 NULLs 传输的数量,有效提升 SpaceWire 总线的效率.在第二级存储器中实现以数据包为单位的大数据块处理,减少了 CPU 频繁访问 SpaceWire 接口控制器,有效提升系统的处理效率^[4].

3 ZTLM-SPW 节点 IP 核设计

3.1 总体设计

本文提出的具有两级存储结构的 SpaceWire 接口控制器 IP 核 ZTLM-SPW 结构如图 1 所示,分为发送器、接收器,完成 SpaceWire 字符层协议,负责处理 DS 的编解码;协议状态机及两级缓存控制逻辑,完成 SpaceWire 交换层协议,负责处理 SpaceWire 节点链路初始化、数据流控制、错误检测以及恢复等;AHB 总线接口,完成 SpaceWire 应用层协议.

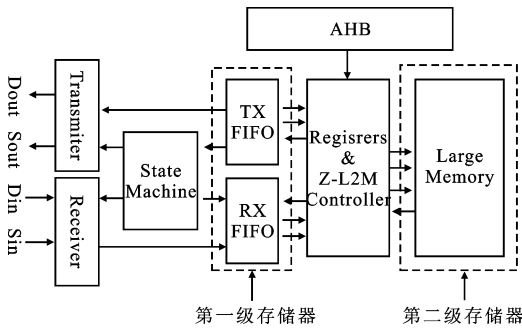


图 1 ZTLM-SPW IP 核结构框图

通过分析第一级存储器的读出端与写入端的传输速率^[5],以及存储器的利用率,本设计的第一级存储器采用 64×9 bit 的 FIFO,第二级存储器的宽度采用 32 bit,深度与实际应用相关,设计中采用了 16×1024×32 bit 的 RAM.

两级存储单元之间的数据调度状态机如图 2 所示,状态机根据第一级存储单元的状态以及处理器的指令完成数据的调度.为了防止第一级存储单元发生上溢或下溢,状态机的状态转移采用短路径方式,完成一种操作之后立刻回到 S0(idle)状态进行实时调度. S1(Read L1 Mem)、S2(Write L2 Mem)

负责将第一级存储器中接收到的数据转移到第二级存储器中. S3(Read L2 Mem)、S4(Write L1 Mem)负责将第二级存储器中待发送的数据转移到第一级存储器中. S5(Write L2 Mem)、S6(Read L2 Mem)负责相应 CPU 的数据操作.

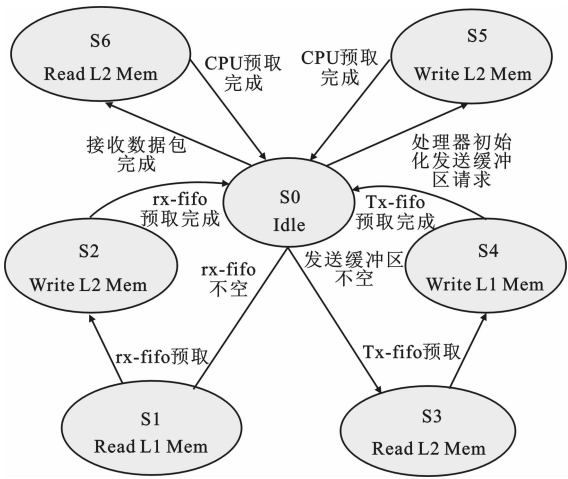


图 2 数据调度状态机

3.2 数据预取及调度算法

数据预取是指控制器(SpaceWire 接口控制器和处理器)在申请数据之前,硬件自主发出访存请求,并将数据准备好.这样就能够将控制器的其他时间与存储器的访问时间重叠在一起,减少了控制器因等待数据而产生的停顿时间^[6-7].由于 SpaceWire 总线中的数据以包为单位,顺序操作,访问地址可预知,所以不存在不命中的问题.通过数据预取,能够有效减少处理器的干预程度.

本设计有三种数据预取:(1)TX-FIFO 数据预取(PF_{tx-fifo}),是指从第二级存储单元中的发送缓冲区中将待发送的数据预取到第一级存储单元 TX-FIFO 中,这样 SpaceWire 总线发送器可以快速取数并发送;(2)RX-FIFO 数据预取(PF_{rx-fifo}),即将第一级存储单元 RX-FIFO 中接收到的数据预取到第二级存储单元的接收缓冲区,这样能够保证 RX-FIFO 中有足够的空间用于接收剩余的 N-CHAR,减少了 NULLs 的发送,提高了总线的效率;(3)CPU 数据预取(PF_{cpu}),即将第二级存储单元中接收到的数据预取到 CPU 读访问队列中,或者 CPU 将待发送的数据预取到第二级存储单元中的发送缓冲区中,这样能够保证 CPU 快速读取数据,提高处理性能.

RX-FIFO 的状态满(full, RF)、空(empty, RE)和 TX-FIFO 的状态满(full, TF)、空(empty, TE)作为数据预取优先执行的判据. SpaceWire 总线系统的数据传输是异步的、全双工的,接收数据具有不确定性,数据的发送由本地的 CPU 发起,因此两级缓存之间的数据调度的优先级如式(7)所示.

$$PF_{rx-fifo} > PF_{tx-fifo} > PF_{cpu} \tag{7}$$

函数 PF_x 的含义:启动 X(rx-fifo、tx-fifo、cpu)预取,两级存储器之间数据调度算法如下:

输入:第一级 FIFO 的存储状态(空状态、满状态)

输出:完成两级存储器间的数据转移

步骤 1:如果(! RE)并且(! TF),则执行

$PF_{rx-fifo}$;

$PF_{tx-fifo}$;

否则步骤 2:如果(! RE),则执行

$PF_{rx-fifo}$;

否则步骤 3:如果 TE,则执行

$PF_{tx-fifo}$;

否则步骤 4:如果包接收完成或有包发送请求,则执行

PF_{cpu} ;

3.3 存储结构设计

在第二级存储器中采用堆栈区和数据区的存储结构实现大数据块灵活传输^[8],堆栈区用于存储收发包的基础信息,如包的起始地址、包的长度,包的时标以及包的状态.数据区用于存储已接收的数据包和待发送的数据.处理器通过堆栈区的信息来读取实际数据,并通过设置多包触发中断的方式来减少 CPU 进入中断的次数.

4 实验验证

为了验证本文提出的两级存储结构在提高 SpaceWire 总线效率以及总线系统的处理效率方面的有效性,设计了一个 SpaceWire 总线系统仿真平台,将 ZTLM-SPW IP 核通过 AHB 总线集成在航天专用 CPU LEON3-FT 上,模拟两个节点实现 SpaceWire 点对点通信,操作系统采用 SpaceOS. 本文将小于 56 Byte 的数据块定义为小数据块,第一级存储器能够一次接收全部数据,不影响总线效率,所以实验中主要测试大数据块传输对总线效率和系统处理效率的影响. 分别选取 1 kB、2 kB、4 kB、8 kB 的数据进行测试,处理器收发数据分别采用查询方式和中断方式. 为了进行比较,本文对采

用一级存储器的 SPW IP 核(简称 OLM-SPW)进行了相同测试.

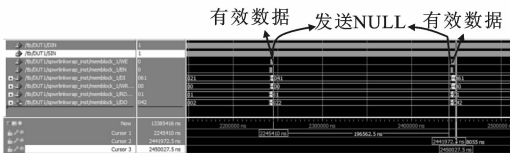
在仿真验证的基础上,搭建的测试平台如图 3 所示. A 为 SpaceWire 专用电缆;B 为 SpaceWire 的第二级大容量存储器;C 为 SPW-RTC 专用测试设备;D 为集成了 ZTLM-SPW 和 LEON3-FT 等逻辑的 FPGA 芯片.

在同等测试环境下,两种设计分别采用查询方式、中断方式进行测试,将测试结果代入式(4)得出的总线效率如表 1 所示.

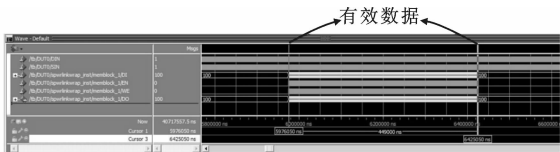
表 1 总线效率

数据量	1kB	2 kB	4 kB	8 kB
OLM-SPW, 查询方式		100 %		
ZTLM-SPW, 查询方式		100 %		
OLM-SPW, 中断方式		1.57 %		
ZTLM-SPW, 中断方式		100 %		

由表 1 可知,采用查询方式能够实时释放接收缓冲区,不影响总线的效率;当采用中断方式后,由于数据量大于第一级存储器的容量,需要多次进入中断完成数据的接收,每次进出中断的时间约为 200 μ s,这期间造成总线上传输大量的 NULLs,影响了总线效率. 在中断方式下,两种设计总线上传输有效数据和 NULLs 如图 3 所示. 本设计能够有效解决 SpaceWire 总线传输大数据块的总线效率问题. 相比 OLM-SPW,本设计能够提高总线效率 62.7倍.



(a)OLM-SPW 的总线效率



(b)ZTLM-SPW 的总线效率

图 3 中断方式下两种设计的总线效率

系统处理效率如表 2 所示.

表 2 系统处理效率

数据量	1 kB	2 kB	4 kB	8 kB
OLM-SPW,查询方式		44.4%		
ZTLM-SPW,查询方式		44.4%		
OLM-SPW,中断方式		1.1%		
ZTLM-SPW,查询方式	17%	19%	45%	62%

采用查询方式一直占用处理器,检测到数据后及时读取,两种设计所消耗的时间相同.在中断方式下,两种设计所消耗的时间有明显的区别 OLM-SPW 需要频繁进入中断,进出中断所消耗的时间远远大于读取数据的时间,系统处理效率仅为 1.1%;而本设计仅需要进入一次中断即可完成数据的接收,并且数据量越大,本设计的优势越明显,系统处理效率由 17%增加到 62%.相比 OLM-SPW,本设计能够提高系统处理效率至少 14.5 倍.两种设计中处理读取数据所消耗的时间如图 4 所示.

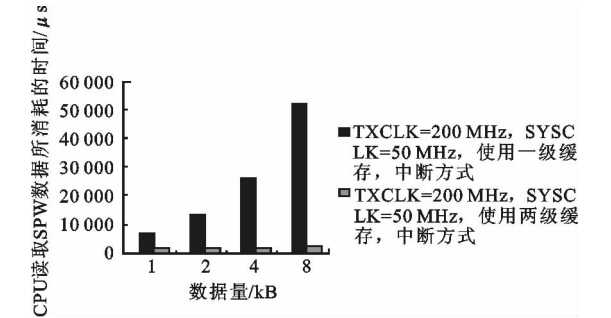


图 4 两种设计的系统处理效率

5 结束语

本文提出一种基于两级存储结构的 SpaceWire 节点控制器,通过硬件预取方式实现数据的自主收发,有效解决了流量控制所导致总线效率低以及系统处理效率差的问题.通过实验验证,本设计能够有

效克服现有流量控制机制在大数据块传输中的缺陷.本设计在多任务系统、大数据块传输的应用领域能够保证 SpaceWire 总线高速、高效传输的优势.

参考文献:

[1] Parkes S, Sheynin Y, Nomachi M. SpeceWire mis-sions and architectures[C]// The 60th International Astronautical Congmss, Daejeon, Korea, 2009;12-16.

[2] 赵俊艺,何兵哲.基于 Spacewird 总线的星载数据系统[J].飞行器测控学报,2009,28(2):13-17.

[3] ECSS-E-ST-50-12C. Space wire-links, nodes, routers and networks[S]. European Cooperation for Space Standardization(ECSS), 2008;24-128.

[4] Parkes S, Armbruster P. Space wire: spacecraft on-board data-handling network[J]. ActaAstronautica, 2010,66(1):88-95.

[5] 王剑,王宏,杨志家.动态参数模型确定 SoC 中异步 FIFO 深度的方法[J].电子科技大学学报,2009,38(3):447-450.

[6] 邓让钰,谢伦国.一种硬件预取机构及其对系统影响的研究[J].计算机工程与科学,2001,23(6):70-72.

[7] Vanderwiel S P, Lilja D J. Data prefetch mechanisms[J]. ACM Computing Surveys(CSUR), 2000,32(2):174-199.

[8] DDC. ACE/Mini-ACE series BC/RT/MTAdvanced communication engine integrated 1553 Terminal User's Guide[M]. New York: Data Device Corporation, 1999;113-130.

作者简介:

赵云富 男,(1981-),硕士.研究方向为 SOC 技术、星载总线技术、大规模集成电路的抗辐射加固技术.
E-mail:zhaoyun6057@163.com