基于 PCIe 交换机的多处理器节点动态管理系统设计

王 轩[1] 邱凯强[1] 王 璞[1]
WANG Xuan QIU Kaiqiang WANG Pu

摘 要 为了满足数据处理模块高性能、高准确性、高可靠性的需求，设计了一种基于 PCIe 交换机的多节点动态管理方法及系统。系统包括 PCIe 交换机、多处理器节点以及 MCU 辅助电路，利用 PCIe 交换机的高速交换特性提升数据交换能力，利用多处理器节点并行工作提升数据计算与存储能力，利用 MCU 辅助电路快速定位系统故障原因并提供备份机制，以提升系统容错能力。结果表明，所设计的多处理器节点动态管理系统清晰、结构明确，具有可实施性，提高了数据处理模块的性能与可靠性，可以作为当前航空机载计算机设计的参考方案。

关键词 PCIe 交换机；多处理器；可靠性；数据交换；容错能力

doi：10.3969/j.issn.1672-9528.2024.07.007

0 引言

随着技术的发展和航空设备飞行功能需求的日益增加，对于航空机载计算机处理任务的准确性、及时性以及可靠性提出了更高的要求。数据处理模块作为机载计算机的核心部件，主要负责提供计算、通信和数据转换等功能。为满足现如今对于机载计算机功能、性能的诸多需要，数据处理模块应具备高性能及高准确性的数据通信与计算能力，同时还需要具备一定的容错备份能力。

PCI-Express（peripheral component interconnect express，PCIe）作为一种高速串行计算机扩展总线标准，主要用于处理器与其他设备间的数据通信。PCIe 为处理器的每个下游设备分配其专用的通道带宽，以保证设备与处理器间进行点对点的全双工高速率传输[1-9]。PCIe 交换机是一种基于 PCIe 通信协议的数据交换与通信设备，其将 PCIe 所支持的处理器与下游设备点对点全双工高速通信能力扩充为多处理器节点间的高速数据交换能力，因此 PCIe 交换机不仅可以提高计算机产品数据交换的速率，还可以为产品多余度架构提供支持。

本文基于 PCIe 交换机的功能特性，设计出一种可应用于数据处理模块的多处理器节点动态管理系统。该系统利用多处理器节点提升计算能力，利用 PCIe 通信特性提高数据交换能力，利用 PCIe 交换机和单片微型计算机（microcontroller unit，MCU）提供多处理器节点管理及容错备份能力。本文对该系统的软硬件架构以及多处理器节点管算法进行了阐述。本文提出的方法很好地应用于机载计算机数据处理模块中，为满足其高性能、高准确性、高可靠性提供支持。

1 系统硬件架构

多处理器节点动态管理系统由四个处理器节点、一个 PCIe 交换机，以及一个 MCU 辅助电路构成。其中，四个处理器节点均采用 FT-2000/4 处理器。PCIe 交换机采用 SM8619 PCIe 交换机。其硬件结构图如图 1 所示。

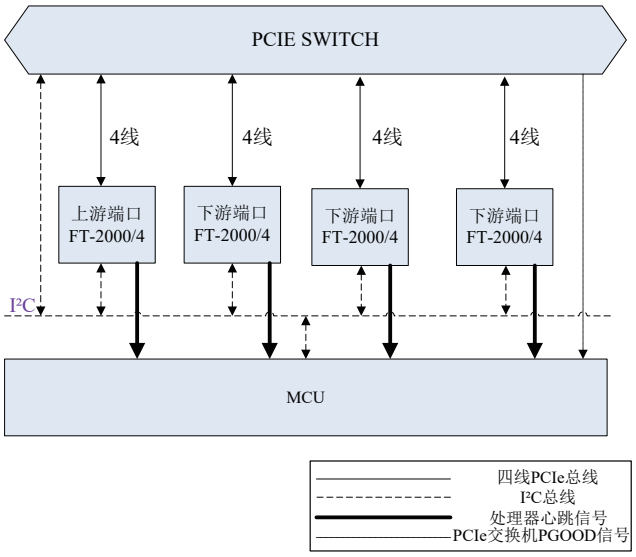

图 1 多处理器节点动态管理系统硬件架构

PCIe 交换机用于实现四个处理器节点间的高速互联。SM8619 PCIe 交换机是一个具有 16 个通道、16 个端口的 PCIe 交换器件。在该系统中，将 SM8619 PCIe 交换机配置为 4 个端口，每个端口对应 4 个通道，且均配置一个处理器节点。此种方式下，每个处理器节点的交换速率均为 4 倍的单通道

1. 航空工业西安航空计算技术研究所 陕西西安 710065

速率，最大程度地利用了 SM8619 PCIe 交换机可提供的数据交换能力。

多处理器节点包括一个上游端口节点、三个下游端口节点，上游端口节点用于 SM8619 PCIe 交换机链路配置、数据处理、数据分发、数据管理。下游端口节点用于接收和处理上游端口节点分配的数据，并将处理后的数据输出。下游端口节点可以将处理后数据直接输送给 SM8619 PCIe 交换机，也可以经上游端口节点输送给 SM8619 PCIe 交换机。四个节点通过 PCIe 总线实现并行工作，因此多处理器节点的配置相较单处理器可处理的数据量以及计算速率成倍增加，进而大大提升了该系统的数据处理和计算能力。

MCU 辅助电路通过 I^2C 总线将四个处理器节点和 SM8619 PCIe 交换机串接，基于此对多处理器节点的工作状态进行监控，并依据监控结果对多处理器节点内的上游端口节点与下游端口节点进行切换。同时，MCU 辅助电路也通过通用输入 / 输出口（general purpose input/output，GPIO）管脚与 SM8619 PCIe 交换机连接，用于 SM8619 PCIe 交换机的工作状态监控。

2 多处理器节点动态管理算法

2.1 算法概述

基于上一章节对于系统硬件架构的阐述，可以得出，本文所提出的多处理器节点动态管理系统基于 PCIe 交换链路，有效地集合了四个处理器的数据计算资源与存储资源，可满足数据处理模块高性能、高准确性的数据计算与交换需求。

本章节主要对节点动态管理算法进行描述。该算法由 MCU 辅助电路、处理器节点以及 SM8619 PCIe 交换机共同实现，可以实时监控 PCIe 交换网络状态、快速定位故障原因并提供备份机制，进而满足了可靠性需求。其具体算法如下。

SM8619 PCIe 交换机内设置有端口配置寄存器，该寄存器可对与指定端口相连的处理器节点是上游处理器节点还是下游处理器节点进行配置。MCU 辅助电路通过配置 SM8619 PCIe 交换机中的寄存器设置节点对应端口的上 / 下游属性。系统初次上电时，硬件通过采样引脚默认值自动加载指定处理器节点为上游端口节点，其他处理器节点为下游端口节点。

正常工作时，上 / 下游节点基于 SM8619 PCIe 交换机提供的交换链路实现数据的高速交换和处理，MCU 负责实时监控 PCIe 交换网络状态。当出现数据传输丢包甚至链路彻底断开的情况时，MCU 辅助电路可以快速定位故障原因并提供备份机制，具体步骤如下。

步骤 1：将 SM8619 PCIe 交换机中的 Lane 状态输出引脚（PEX_LANE_PGOOD[15:0]）连接到 MCU 中的 GPIO 管脚。MCU 通过该管脚发出的电平信号来监测 PCIe 链路状态检测，低电平为该链路未连接，高电平表示已成功建链（5.0 GT/s），高低波形信号表示已成功建链（2.5 GT/s）。

步骤 2：将每个处理器中的一个 GPIO 管脚连接到 MCU 中的 GPIO 管脚。处理器通过其 GPIO 管脚发送心跳信号给 MCU，通过心跳信号实时反映处理器节点自身工作状态，若心跳信号以约定的频率更新代表处理器节点工作正常，心跳信号消失代表处理器节点工作不正常。

步骤 3：MCU 辅助电路通过其 GPIO 管脚接收到的 SM8619 PCIe 交换机的链路状态中电平信号和各处理器节点的心跳信号，判断是链路状态故障还是处理器节点故障，并进行记录故障及上报结果。

步骤 4：若为电平信号不正常，则判断为链路故障，通过外部总线上报故障信息给外部整体控制设备，从而及时作出补救措施。

步骤 5：若上游节点的心跳信号不正常，则判断上游节点出现故障。

在传统的 PCIe 架构下，PCIe 的设备枚举和交换空间配置均由上游节点完成。若上游节点故障，则会由于无法配置管理 PCIe 交换空间及下游节点而导致整个 PCIe 交换网络丧失工作能力。而本系统可在上游节点故障时，将其自动切换为下游节点，并将一个下游节点配置为新的上游节点。由新上游节点进行链路配置、数据处理、数据分发、数据管理，以保证 PCIe 交换网络正常工作，进而提供备份机，提高产品的容错能力。

其中，上下游节点切换示意图见图 2，具体步骤如下。

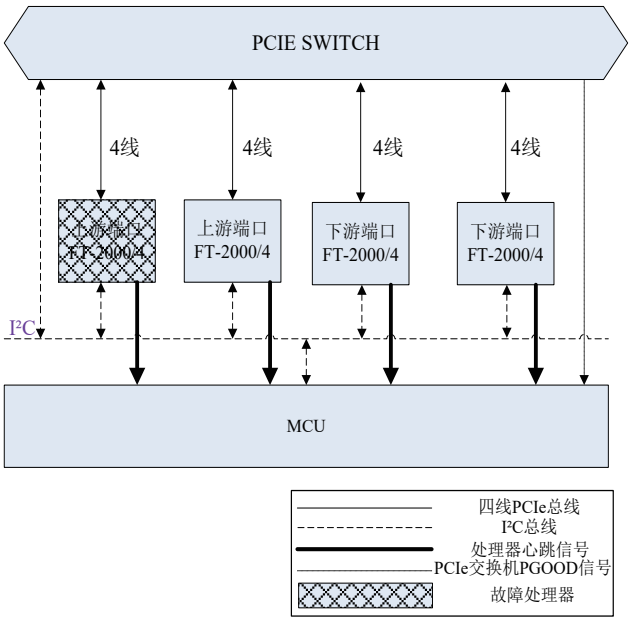

图 2 处理器节点上 / 下游属性切换示意图

步骤 5.1：MCU 通过 I²C 总线将 SM8619 PCIe 交换机的奇偶端口禁止寄存器的 Disable Port X 位置 1。

步骤 5.2：MCU 通过 I²C 总线向 SM8619 PCIe 交换机的 1DCh[27:24] 中写入当前需要设置为上游端口的处理器节点对应端口号。

步骤 5.3：MCU 通过 I²C 总线将 SM8619 PCIe 交换机的奇偶端口禁止寄存器的 Disable Port X 位置 0。

2.2 冷 / 热备份机制

基于多处理器节点动态管理算法可为系统配置冷、热两种备份机制。在上游节点发生故障时，若系统处于运行状态，则可启用热备份机制，以保证系统正常工作；若系统处于下电状态，则可启用冷备份机制。

2.2.1 热备份机制

配置热备份机制除确定上游端口外，还需要确定一个下游端口为非透明端口。该端口可通过非透明桥 [10] 与交换机上所有下游端口进行数据交换，进而可在上游端口故障后接替其完成数据分发任务。其具体操作步骤如下。

步骤 1：选定一个下游端口用作非透明端口，上电后首先进行非透明桥配置，之后挂机等 MCU 发送消息。

步骤 2：在 MCU 检测到当前上游端口故障后，通过 I²C 总线向非透明端口发送消息。

步骤 3：非透明端口在接收消息后，接替故障端口执行上游节点的周期任务并发送心跳标。其他下游端口在此期间一直正常运行周期任务。

2.2.2 冷备份机制

针对冷备份机制，其具体操作步骤如下。

步骤 1：在 MCU 检测到当前上游端口故障后，在剩余处理器节点中，选择一个正常工作的处理器作为上游端口（默认按照端口号由小到大进行选择），记录在其 Flash 指定区域。

步骤 2：重新上电后，MCU 从 Flash 读取待配置的上游端口号。

步骤 3：MCU 配置 SM8619 PCIe 寄存器，设置上游端口。

步骤 4：MCU 通过 I²C 向上游端口对应处理器节点发送消息，告知其被配置为上游端口。

步骤 5：上游端口收到消息后，执行 PCIe 总线初始化操作，初始化完成后，开始执行周期任务，并通过 GPIO 管脚定时发送心跳标。

步骤 6：下游端口读取 PCIe 配置寄存器获取各自的数据交换空间地址，基于最新的地址执行周期任务，并通过 GPIO 管脚定时发送心跳标。

步骤 7：MCU 接收到下游端口发送的第一拍心跳后（以接收到第一个下游端口发送的心跳标为基准），开始执行链路健康管理任务。

3 软件设计

3.1 上游端口

上游端口的主要任务包括 PCIe 的设备枚举、交换空间配置以及周期性数据处理任务。PCIe 的设备枚举与交换空间配置采用深度优先算法递归实现，在周期任务开始时执行。算法以上游端口为根节点进行搜索，并将 SM8619 PCIe 交换机上所有下游端口视为子树根节点。在搜索过程中，每访问到一个节点便依据硬件，需要为其划分相应的交换空间，直至遍历完下游端口连接的设备。若当前系统中包含非透明端口，则将该端口视作叶子节点，搜索到该节点时停止遍历，不再对其下连接的设备进行递归枚举操作 [11]。

在 PCIe 初始化完成后，上游端口开始执行周期任务，包括接收外部数据；将按照数据包格式定义进行组包，并通过 PCIe 总线分发给不同的下游端口；通过 PCIe 从指定地址区域读取下游端口处理好的数据包，并在每个周期任务执行过后向 MCU 发送心跳标。

3.2 下游端口

下游端口的主要任务包括数据处理以及配合 MCU 完成上下游端口的切换工作。在上电后，下游端口进入挂机等待状态，直至接收到 MCU 发送的消息或挂机超时。若接收到消息，则执行上游端口相关任务；若超时，则开始执行周期任务，包括通过 PCIe 从指定地址区域读取上游端口分发数据包，解析后处理数据，将处理好数据按照数据包格式定义进行组包放入交换区域，并在每个周期任务执行过后向 MCU 发送心跳标。

3.3 非透明桥

如图 3 所示，非透明桥包括两个接口，分别为虚拟接口（link interface）和链路接口（virtual interface）。链路接口与图 3 中的系统主机域相连，由非透明端口所连接节点进行管理。除此之外，所有的下游节点与虚拟接口均属于本地主机域，在上游端口设备枚举阶段进行初始化。系统主机域与本地主机域进行跨域传输时，通过地址翻译机制（address translation）实现。

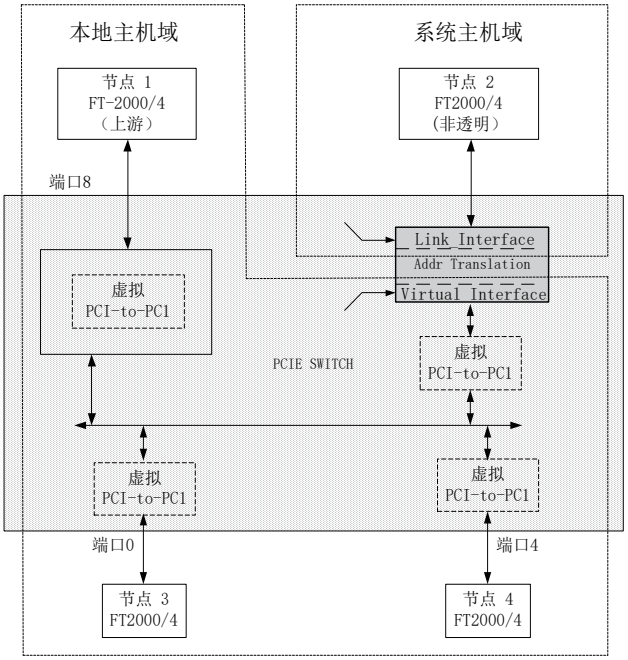

图 3 配置非透明端口的系统拓扑图

通过配置链路接口相关寄存器，可实现系统主机域访问本地主机域；通过配置虚拟接口相关寄存器，可实现本地主机域访问系统主机域。针对本文所述系统，需配置链路接口以支持热备份机制，具体步骤包括以下 6 步。

步骤 1：上游端口在设备枚举后，配置链路接口的 setup 寄存器，寄存器的值为下游端口各个交换空间的大小。

步骤 2：上游端口配置链路接口的 translation 寄存器，寄存器的值为下游端口各个交换空间的起址。

步骤 3：上游端口配置链路接口的 LUT 寄存器，寄存器的值为回读后 &0x0001，以允许非透明端口穿过非透明桥，访问下游端口设备。

步骤 4：仅 1 次，测试上游端口枚举 + 上述配置所需的时间，记为 T_1。

步骤 5：非透明端口上电初始化过程前，等待 T_1 时间，以便上游端口完成对非透明桥链路接口的配置。

步骤 6：非透明端口后续可通过链路接口访问所有下游节点设备。

3.4 MCU 辅助电路

MCU 辅助电路上电后，首先读取 Flash 地址获取默认上游端口号，若端口号为初始值，则执行周期任务，包括读取四个节点心跳标以及 Lane 状态输出信号。若发生更改，则根据冷备份机制执行链路重配置操作，并在配置结束后按照新的上下游端口号执行周期监控任务。若在周期监控时发现故障，则依据故障类型或上报故障状态执行热备份机制。

4 结语

本文提出一种可应用于数据处理模块的多处理器节点动态管理系统，并对该系统的软硬件架构以及多处理器节点管算法进行了阐述。经前文论述可知，本文提出的方法很好地应用于机载计算机数据处理模块中，为满足其高性能、高准确性、高可靠性提供支持。

参考文献：

[1] 王之光 , 高清运 . 基于 FPGA 的 PCIe 总线接口的 DMA 控制器的设计 [J]. 电子技术应用 ,2018,44(1):9-12.

[2] 李亚南 , 吴建斌 , 谢桂辉 , 等 . 一种 PCIe3.0 高速数据采集卡驱动及上位机软件的实现 [J]. 电子测量技术 , 2018, 41(7): 129-133.

[3] 韩强 . 基于 PCI Express 总线架构的多处理器模块设计 [J]. 信息通信 ,2018(5):137-139.

[4] 王齐 .PCI Express 体系结构导读 [M]. 北京 : 机械工业出版社 ,2010.

[5] 张锐 , 曹彦荣 . 基于 PICE 交换的数据处理模块设计 [J]. 电子技术设计与应用 ,2014(6) :83-85.

[6] 毕城 , 元永红 . 基于 PCIe 总线的多处理器数据交换技术 [J]. 电子科技 ,2017,30(7):118-120.

[7]SMITH J A, JOHNSON M P.PCIe bus technology: an overview and analysis[J].Journal of computer and communications,2018,6(2):1-10.

[8] 李明 , 王刚 . PCIe 总线技术研究及其在高速数据传输系统中的应用 [J]. 通信技术 ,2020,53(4):94-100.

[9]DAVIS T R, WILLIAMS R S.Design and implementation of a high-speed data transmission system based on PCIe bus[J]. IEEE transactions on industrial electronics,2022,69(3):2713-2722.

[10] 徐健 , 张建泉 , 张健 . 基于 PCIE 非透明桥的嵌入式异构平台设计 [J]. 微电子学与计算机 ,2018,35(1):26-30.

[11] 黄睿 , 苏阳 , 赵英潇 , 等 .PCIe 高速数据记录方案设计与软件实现 [J]. 电子科技 ,2018,31(7):75-78.

【作者简介】

王 轩（1995—），女，陕西乾县人，硕士研究生，工程师，研究方向：嵌入式软件。

（投稿日期：2024-05-13）