

芯片间时间触发消息堆叠调度方法

臧光界^{1,2}, 李靖^{2,*}, 王彤², 熊华钢²

(1. 中国航空研究院, 北京 100029; 2. 北京航空航天大学 电子信息工程学院, 北京 100191)

摘 要: 时间触发 (TT) 通信方式应用于芯片间互连网络, 以保证航空电子通信任务之间消息传递的严格时间确定性。当航空电子任务具有多种操作模式, 属于不同模式的时间触发调度表会重叠占用时隙, 提出芯片间时间触发消息堆叠调度方法, 以提高利用网络资源的灵活性和效率, 同时减小应用层消息由于等待时间触发时间窗的排队延迟。仿真实验表明: 与超调度方法相比, 所提方法能够减小芯片间互连网络中时间触发消息的总端到端延迟和链路平均时隙占用率, 对于端到端延迟时间较长且链路平均承载消息传输较多的场景, 采用所提方法减少端到端延迟的效果更显著。

关 键 词: 芯片间互连; 时间触发消息; 操作模式; 堆叠调度; 端到端延迟

中图分类号: TP393

文献标志码: A

文章编号: 1001-5965(2023)07-1838-09

在综合模块化航空电子系统(integrated modular avionics, IMA)是最常见的航空电子系统体系架构^[1]的基础上, 分布式综合模块化航空电子系统(distributed IMA, DIMA)采用具有时间触发(time-triggered, TT)能力的交换式网络实现安全关键性消息的严格周期确定性, 如 SAE AS6802 标准定义的时间触发以太网(time-triggered ethernet, TTE)^[2]。

随着可综合模块逐渐朝着微小化方向发展, 综合化互连将不仅局限于局域网, 片上系统(system on chip, SoC)之间也可以进行综合化互连^[3]。例如, 欧洲的 DREAMS(Distributed Real-time Architecture for Mixed-criticality Systems)项目提出了多核芯片构成的混合关键性流量网络模型, 旨在实现网络的安全性和实时性, 提高应对突发情况的能力^[4]。Paukovits^[5] 和 Kopetz^[6] 等提出了时间触发片上网络(time-triggered system-on-chip, TTSoC)的概念, Urbina 通过 TTE 实现了多个多核芯片互连的跨层次体系结构^[7], 文献 [8-9] 对片内和片间互连通信方式进

行了研究, 提出了实现片内片间通信的综合互连网络模型。

传统上 TT 消息在网络中传输依照离线生成的调度表。文献 [10-11] 提出了基于满足性模理论(satisfiability modulo theories, SMT) 的 TTE 调度方法, 对消息的端到端延迟、任务的优先级等进行约束, 得到 TT 消息的调度表。文献 [12] 提出一种基于贪婪随机自适应搜索算法的 TTE 调度方法。

随着混合关键性系统的规划与设计方式的进展, 需要改进调度方法提高其灵活性, 使其可以在一定程度上不需要重新离线配置, 就可以承载意料之外的关键流量。文献 [13] 提出了一种芯片间互连网络 TT 消息调度方法, 文献 [14] 使用免疫算法对芯片间 TT 消息调度方法进行了优化, 但都未提出提高其灵活性以对新加入 TT 消息进行调度的方法。文献 [15] 提出使用拓展的软件定义网络(software-defined networking, SDN) 来实现动态的数据包处理和调度; 文献 [16] 通过调整现有流量在各

收稿日期: 2021-09-15; 录用日期: 2022-01-16; 网络出版时间: 2022-02-15 11:34

网络出版地址: kns.cnki.net/kcms/detail/11.2625.V.20220214.1419.003.html

基金项目: 国家自然科学基金(62071023)

* 通信作者. E-mail: avionics@buaa.edu.cn

引用格式: 臧光界, 李靖, 王彤, 等. 芯片间时间触发消息堆叠调度方法[J]. 北京航空航天大学学报, 2023, 49(7): 1838-1846.

ZANG G J, LI Q, WANG T, et al. Stacking scheduling method for time-triggered messages in off-chip network[J]. Journal of Beijing University of Aeronautics and Astronautics, 2023, 49(7): 1838-1846 (in Chinese).

端口的触发时间来为新加入消息预留时隙,但是该方法改变了原有消息的实时性需求,且在网络规模较大、流量较多时难以实现。以上方法都属于超调度方法(super-schedule approach),即不考虑通信任务所属的模式,只是将其所有模式中的消息依次调度。网络互连通信中的所谓“模式”,指实现各种给定功能的不同工作阶段,在不同模式中,网络传输的消息不同。文献 [17] 提出了一种堆叠调度方法(stacked scheduling approach, SSA),将不同模式下的 TT 消息堆叠在同一时隙,该方法的目的在于为偶发消息预留带宽;文献 [18] 给出了堆叠调度方法的具体实施方案,但只适用于交换机控制模式切换的情况,并只考虑了芯片间互连中交换结构均集中于各个端口的情况。

基于以上研究现状,本文提出一种芯片间综合化互连网络 TT 消息堆叠调度方法,使 TT 消息的调度综合考虑了 TT 消息的顺序、模式及芯片间互连网络的拓扑等因素,并根据给定的芯片数量及消息的属性、所属模式、顺序关系等条件,给出生成芯片间互连拓扑结构和 TT 消息调度表的方法。该方法可使链路对网络带宽的利用率达到最大,并尽可能减小流量的端到端延迟,提高芯片间互连网络的灵活性和适应性。

1 系统模型

1.1 芯片间互连结构

芯片间互连的拓扑结构可以是规则的(如网格形),也可以根据应用要求设计呈现不规则的拓扑,而前者可以被看作后者的特例。为了一般性,本文提出并讨论的调度方法可以适用于不规则的拓扑结构。图 1 为不规则的互连拓扑结构。

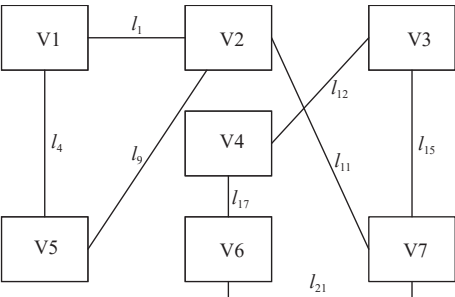


图 1 芯片间互连结构示意图
Fig. 1 Sketch map of off-chip interconnection

图 1 中, V1~V7 表示芯片,在网络拓扑中以节点形式表示,芯片之间通过开放式的介质无关接口建立物理链路,进行全双工通信,目前常用的芯片间互连方式有 SPI 总线协议、I²C 总线协议、Rapid IO 架构等。图中芯片之间的连线表示物理链路,芯

片的物理链路数量和布局根据数据传输的需求确定,在一般情况下是不对称的。为使链路序号反映链路两端节点,不失一般性,对链路序号 i 的标定依据式 (1) 进行:

$$i = \sum_{k=n-s}^{n-1} k - (n-d) \tag{1}$$

式中: n 为网络芯片数; s 和 d 分别为该链路两端芯片编号的较小值和较大值。

可以使用 n 阶对称矩阵 G 在算法中表示芯片间互连网络的拓扑结构, n 为网络中芯片的数量,其主对角线上各元素为 0,其他元素中若 $G(i, j) = 1$,则认为芯片 V_i 与 V_j 由物理链路直接相连,若 $G(i, j) = 0$ 则认为 2 个芯片不由物理链路直接互连。

1.2 芯片间互连 TT 消息模型

设在芯片间互连网络中传输的 n 条消息集合为 $M = \{m_1, m_2, \dots, m_n\}$, 其中,第 i 条 TT 消息 $m_i (i = 1, 2, \dots, n)$ 的属性可表示为

$$m_i = \langle p_i, s_i, \text{path}_i, o_i, \text{mode}_i \rangle \tag{2}$$

式中: p_i 为消息周期; s_i 为消息长度; path_i 以 TT 消息源节点、中转节点、目的节点编号的有序数列表示 TT 消息的传输路径; o_i 为消息在经过各条物理链路上的发送起始时间偏移量; mode_i 为该 TT 消息在该网络中所处的模式,TT 消息不会在自身不处于的模式中产生。在堆叠调度方法中,默认所有 TT 消息的周期、长度和模式已知,计算每条 TT 消息的发送起始时间偏移量和路径,并由此规划芯片间互连网络的拓扑。

1.3 堆叠调度原理

所谓堆叠,即在同一时隙内,同时安排 2 条或更多的 TT 消息传输,被堆叠的 TT 消息不属于同一模式,因此不会同时产生和传输引发冲突。传统的 TT 网络中,事先已为每条周期性消息分配一段固定的传输时隙,因此,在一条物理链路的某段时隙,即使该时隙上没有对应 TT 消息的实例传输,系统也不会将其分配给其他消息。在对多模式网络进行调度时,若采用超调度方法,则在多模式情况下会导致很低的带宽利用率和很差的适应性。堆叠调度方法则避免了对带宽和时隙的浪费,提高了网络的适应性。

例如,某 TT 网络中的一条物理链路在一段时间之内的 TT 消息传输情况如图 2 和图 3 所示。设该网络具有 3 种操作模式,分别为 mode_1 、 mode_2 、 mode_3 ,在该链路上有 6 条 TT 消息传输,分别为 m_1 、 m_2 、 m_3 、 m_4 、 m_5 、 m_6 ,周期均为 T ,其中, m_1 、 m_2 属于 mode_1 , m_3 、 m_4 属于 mode_2 , m_5 、 m_6 属于 mode_3 。使用

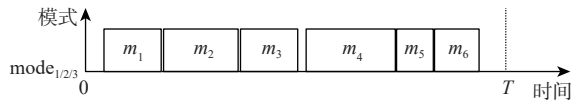


图2 物理链路在多模式情况下使用超调度方法的调度结果

Fig. 2 Scheduling results of physical link using super-schedule approach in multi-mode

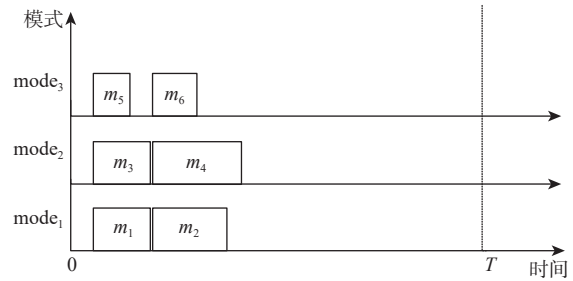


图3 物理链路在多模式情况下使用堆叠调度方法的调度结果

Fig. 3 Scheduling results of physical link using stacking-schedule approach in multi-mode

时间窗来表示 TT 消息在链路上的调度结果^[19],当使用超调度方法进行调度时,如图 2 所示,虽然由于不需要在各芯片中加入模式转换功能使得调度更加容易实现,但是在该调度方法下,时间资源要按多种模式下 TT 消息负载之和预留,因此留给低优先级流量的时隙更少,当该链路所在网络处于运行状态时,如果有新的消息在线加入,网络将很难为其在该链路上分配时隙,导致该网络具有较差的灵活性和适应性。

图 3 为该链路采用堆叠调度方法的调度结果。转换请求消息包含有要求目的节点转换为指定模式的信息,模式转换请求消息可通过修改的协议控制帧(modified protocol control frame, MPCF)传输, MPCF 是具有最小长度的标准以太网帧(46Byte),与普通协议控制帧(protocol control frame, PCF)不同,其类型字段被设置为 0x06^[19],且将 PCF 的保留和透明时钟字段都改为数据加载字段。MPCF 与 PCF 具有相同的优先级,高于 TT 和 RC 流量,为避免干扰时钟同步过程, MPCF 在同步过程之外传输。当网络的操作模式变化时,由一个或几个节点发出模式转换请求消息,通过链路传输至网络的其他节点,收到转换请求消息的节点在该网络中所有 TT 消息的超周期末尾时转换模式,使用新模式下的调度表。与超调度方法相比,堆叠调度方法增加了链路时隙的利用率,能够在运行过程中为更多必要的事件触发消息分配时隙,增加了 TT 网络的灵活性和适应性,同时减小了被堆叠消息的端到端延迟。

1.4 芯片间网络模式切换方法

在由交换机和端系统组成的交换式网络中,由网络中的主节点发送模式更改消息,从节点接收模式更改消息后在该超周期的末尾改变操作模式^[20]。但是在芯片间互连时间触发网络中,网络的拓扑及每个节点需要处理的消息数在调度完成之前都不确定,因此并不能设定一个或多个可以产生和传输模式更改消息的主节点。

在模式更改消息设计方面,若模式更改消息的传输路径与该模式下 TT 消息的传输路径不同,则很可能出现模式更改消息早于某条 TT 消息帧到达其目的或中转节点的情况,在此情况下,由于该节点的模式已被改变,将不会为该 TT 消息帧提供接收或转发服务,造成 TT 消息的丢失;另外,若某条 TT 消息在网络中的跳数较多,当全局时钟到达超周期的末尾时,该 TT 消息可能还未完成传输过程,此时更改模式同样会造成消息的丢失。

在交换式 TT 网络的模式切换方法基础上,提出一种以单条 TT 消息为单位,依节点顺序依次切换模式方法,即当前模式需要切换时,该模式下每条 TT 消息的起始节点在发送当前 TT 消息的最后一帧后紧接着发送一条与此 TT 消息传输路径相同的模式转换请求消息,路径上的每一个转发节点芯片在接收到此模式转换请求后即停止对应 TT 消息的转发和接收服务,当一个节点接收到所有该模式下此节点接收和转发的 TT 消息对应的模式转换请求时,改变该节点的工作模式。图 4 为该模式转换方法的一个实例。

图 4 中,操作模式 1 包含 2 条 TT 消息, m_1 的传输路径 $path_1 = \{V1, V3, V4\}$, TT 消息 m_2 的传输路径 $path_2 = \{V2, V3, V4\}$, 当网络从模式 1 切换至模式 2 时, m_1 的起始节点 V1 和 m_2 的起始节点 V2 分别沿传输路径发送模式转换请求消息 cm_1 和 cm_2 。如图 4(a)所示,假设 cm_1 先于 cm_2 发送, V1 发送 cm_1 后进入模式 2, V2 此时还处于模式 1, 在图 4(b)中, V2 发送 cm_2 后进入模式 2, 此时 cm_1 已先到达 V3, 由 V3 接收后向 V4 转发, 此时 V3 停止对 m_1 的接收和转发服务,但仍属于模式 1, 仍为模式 1 的其他消息提供服务, 在图 4(c)中, 当 cm_2 到达 V3 后, V3 已接收到模式 1 的所有模式转换请求消息, V3 切换为模式 2, 同理, V4 在接收到 cm_1 和 cm_2 后也切换为模式 2, 整个网络模式切换完成, 如图 4(d)所示。

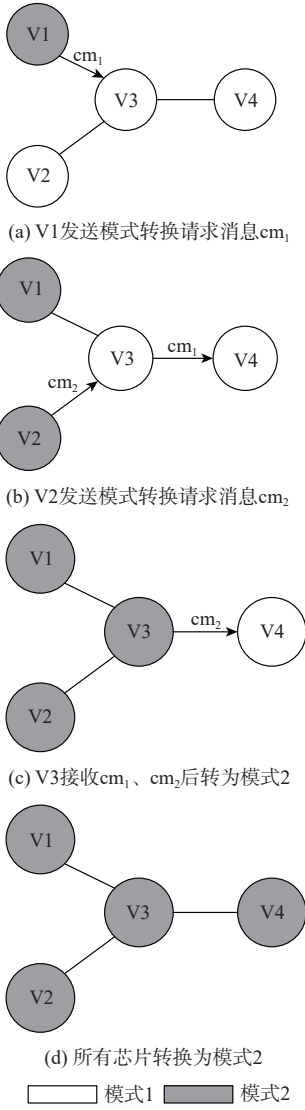


图 4 芯片间互连网络模式切换过程实例

Fig. 4 An example of mode-change in off-chip interconnection network

2 本文方法

对于一条 TT 消息 m_i , 若其发送起始时间偏移量 $o_i = t$, 则其第 j 帧的发送时间 o_i^j 满足:

$$o_i^j = t + jp_i \quad (3)$$

因此对于一组 TT 消息, 只要确定了其第一个超周期内的偏移量, 即可确定每条 TT 消息在所有时段上的调度结果。设 c_i 为 m_i 的执行时间, 满足:

$$c_i = \frac{s_i}{r} \quad (4)$$

式中: r 为网络数据传输速率。

在一条物理链路上, 可供调度的时间资源被记为 S , A 为一定数目的整数集合, 其中会出现几组连续的整数, 将非空集合 A 中出现的连续整数组中的数字最多的一组数字个数记为 $LLC(A)$, 集合 H 表示一个超周期内所有时间, 集合 U 表示超周期

内已被调度的所有时刻, 集合 R 表示一个超周期内可调度的空余时间, 则有

$$U \cup R = H \quad (5)$$

$$U \cap R = \emptyset \quad (6)$$

m_i 在该链路上可调度的条件为^[20]

$$c_i \leq LLC(R) \quad (7)$$

若 m_i 需经多个中转节点转发, 其在前一链路上传输的结束时间为 t_e , 后一链路上传输的开始时间为 t_s , 这 2 个参数应满足:

$$t_e \geq t_s \quad (8)$$

图 1 所示拓扑用矩阵表示为

$$G = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \quad (9)$$

由于拓扑是不确定的, 可以先假设所有节点之间都是互相连接的, 再分别在每个模式下对消息依序在各链路上寻找时隙, 在此期间对已使用端口数等于其最大端口数的芯片周围链路进行“剪枝”, 即切断与其相连但未被使用的链路。根据以上规定和原则提出芯片间网络 TT 消息堆叠调度方法的具体算法, 在该算法中定义了几种操作函数, 包括模式提取函数 `checkmode`、链路检查函数 `linkavailable`、时隙匹配函数 `LLCavailable` 及剪枝函数 `cutedges`, 具体步骤如下。

输入: TT 消息集合 M , 芯片数量 n , 消息数量 msg , 模式数 mo , 各芯片端口数 a 。

输出: 拓扑矩阵 G , 调度表 S 。

```

1: function OFFCHIPSTK
2:  $S \leftarrow \emptyset$ ,  $S_i \leftarrow \emptyset$ ,  $G \leftarrow \text{fillbyone}(n) - E$ 
3: for  $i \leftarrow 1$  to  $mo$ 
4:    $M_i \leftarrow \text{checkmode}(M, i)$ ,  $msg_i \leftarrow \text{size}(M_i)$ 
5:   for  $j \leftarrow 1$  to  $msg_i$ 
6:      $route_j \leftarrow \text{kshortestpath}(\text{source}_j, \text{dest}_j, G)$ 
7:      $rtn \leftarrow \text{size}(route_j)$ 
8:     for  $k \leftarrow 1$  to  $rtn$ 
9:       if linkavailable( $G, route_j(k)$ ) == true
10:        if LLCavailable( $S_i, route_j(k)$ ) == true
11:           $path_j \leftarrow route_j(k)$ 
12:           $S_i \leftarrow S_i \cup \text{singlesche}(route_j(k), c_j, S_i)$ 
13:        break
14:      end if
15:    end if

```

```
16:       $G \leftarrow \text{cutedges}(\text{path}_j, G, S_i)$ 
17:    end for
18:  end for
19:   $S \leftarrow S \cup S_i, S_i \leftarrow \emptyset$ 
20: end for
21: end function
```

首先,对 TT 消息按照重要性排序,若无已给出的重要性规定,则优先调度周期小的消息。为了给通过 MPCF 传输的模式转换消息预留时隙,在计算每条 TT 消息的帧长时,都在其原长度的基础上加上 MPCF 的长度,这样在没有模式转换请求的情况下,该预留时隙也可供其他事件触发消息传输使用,在一定程度上减少了事件触发消息的拥塞。

在本文方法中,首先假设所有芯片之间均互相连通,函数 $\text{fillbyone}(n)$ 表示创建一个所有元素均为 1 的 n 阶方阵。然后按照模式—消息—链路的顺序依次调度,对于原始消息列表 M ,使用函数 checkmode 将其中属于 mode_i 的所有消息提取出来组成新的消息列表 M_i 。路径确定方面,可以使用 k 最短路径方法($k_shortest\ paths$)^[20],依据拓扑和消息的源节点和目的节点确定 k 条实现传输的最短路径,依次判定该路径是否可行。函数 linkavailable 和 LLCavailable 分别用来判断该路径在拓扑中是否存在及该路径上所有链路是否满足式(7)中的 TT 消息可调度条件,若满足,则通过函数 singlesche ,依据延迟最短原则为其安排传输时隙并将其加入当前模式调度表 S_i ,若不满足则判断下一条传输路径是否满足。当一条 TT 消息在其所经过的各条链路上的发送时间偏移量都确定之后,即认为其调度完成,此时使用函数 cutedges 完成对拓扑的“剪枝”,其具体功能为:判定网络中的每个节点已经使用的链路数是否已经达到其端口数 a ,若已达到则删除拓扑中所有与该节点相连的未使用的链路。当一个模式下所有的消息都调度完成后,将当前模式下调度表与其他模式调度表合并,并新建一张空的调度表进行下个模式的调度。

本文方法中,不同模式下的各组 TT 消息调度过程是互相独立的,即调度过程中每一模式都继承之前模式的拓扑但不继承其调度表,相当于在每个模式调度结束后都将当前调度表重置一次,这样就可以使一段时隙在多个模式中都可使用,采用延迟最短原则要求在每条链路上满足式(7)的条件下使 t_s 尽可能小,这 2 项原则共同使得消息的时间窗在链路的调度表上实现堆叠。

3 仿真实验与分析

3.1 实验参数设置

设网络中 TT 消息的传输速率为 100 Mbit/s,已

知以太网帧的帧长范围为 64~1 514 Byte,时间触发流量的周期取为 1~128 ms 之间的幂次值,以此来随机生成 TT 消息,案例中将 TT 消息的周期设为 $p = 2^m \times 3^n$ ms,其中, $0 \leq m \leq 7$ 、 $0 \leq n \leq 4$ 且 m 、 n 均为整数,最终需满足 $1 \leq p \leq 128$ 。消息的源节点、目的节点和所属模式都根据事先设定好的限制随机生成,每组实验的限制都各不相同。为进行对比,以现有的一种超调度方法——芯片间 TT 消息拓扑路径时刻(topology, route, schedule, TRS)调度方法的调度结果作为对照^[15],以消息列表中所有 TT 消息的端到端延迟之和及链路的平均时隙占用率分别表示调度后网络的实时性和灵活性。其中,链路平均时隙占用率为链路中已被占用的时隙占总时间的比例,表示为

$$U = \frac{1}{f} \sum_{i=1}^f \frac{u_i}{h}$$

(10)

式中: f 为网络中最终被激活的链路数; u_i 为链路 l_i 上已被占用的时隙。 U 越小,表明链路的平均占用率越低,可以用来调度零星事件触发消息的时隙也就越多,网络的灵活性也就越高。

设置网络的芯片数、TT 消息数和操作模式数来对比堆叠调度和 TRS 调度方法,通过比较评价堆叠调度方法的效果。具体参数设置如表 1 所示,其中对实验组内具体每条消息的属性是否相同不做限制。

表 1 实验参数设置

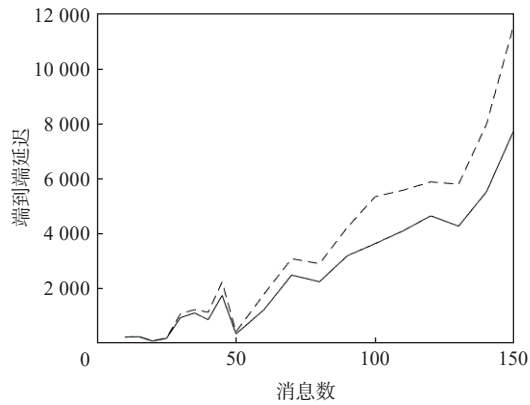
Table 1 Experimental parameter settings

实验组号	消息组数	每组消息数	网络芯片数	操作模式数
1	19	10~150	10	3
2	8	50	5~15	5
3	8	50	10	3~10

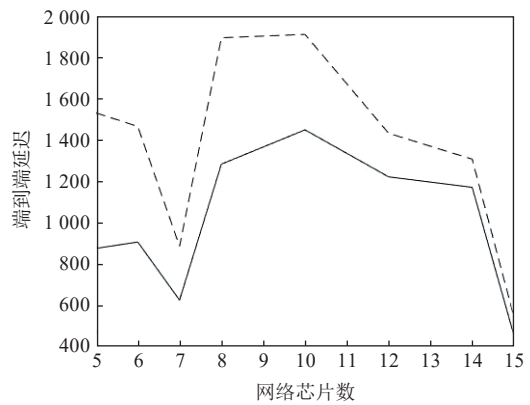
3.2 仿真结果分析

实验中,消息的端到端延迟为消息的发送起始时间偏移量与消息在其路径上各个芯片之间传输时的执行时间之和。其中,发送起始时间偏移量即为消息产生到发送所等待的时间,同时当网络的传输速度不变时,数据帧在任意 2 个芯片之间传输的执行时间相同,因此传输的执行时间之和可以用消息在网络中传输的跳数与在任意链路上传输时间之积表示。分别计算每组消息使用 TRS 调度方法和堆叠调度方法后,该组 TT 消息总的端到端延迟和链路平均时隙占用率,结果如图 5 和图 6 所示。

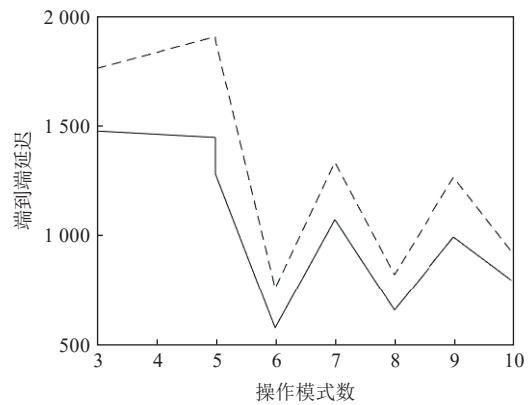
从图 5 可以看出,在 3 组实验中,由于每组 TT 消息生成的随机性,导致不同组消息的平均长度具有较大差异,从而使得各组 TT 消息的总端到端延



(a) 端到端延迟随消息数变化



(b) 端到端延迟随网络芯片数变化

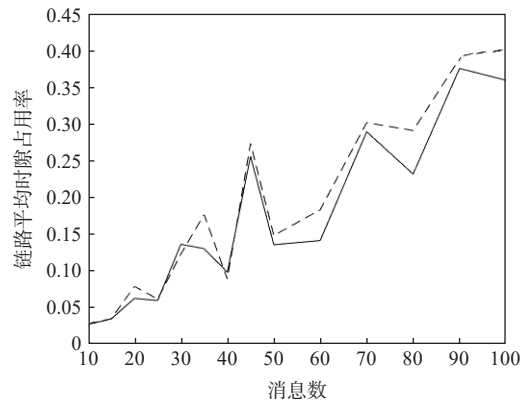


(c) 端到端延迟随操作模式数变化

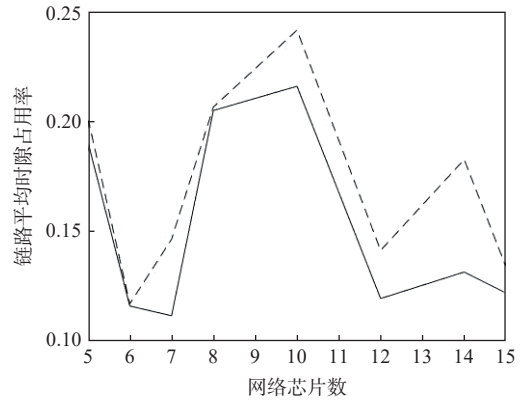
—— 堆叠调度方法 --- TRS调度方法

图 5 不同条件下端到端延迟的变化

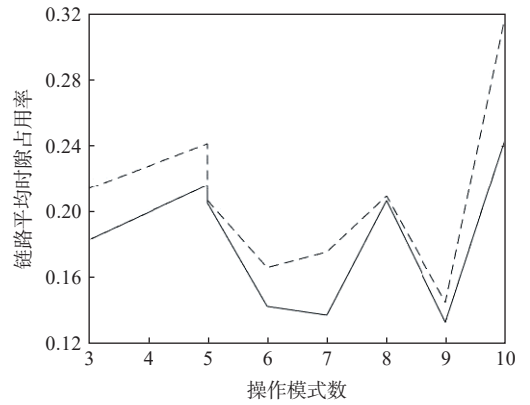
Fig. 5 Change of end-to-end delay under different conditions



(a) 链路平均时隙占用率随消息数变化



(b) 链路平均时隙占用率随网络芯片数变化



(c) 链路平均时隙占用率随操作模式数变化

—— 堆叠调度方法 --- TRS调度方法

图 6 不同条件下链路平均时隙占用率的变化

Fig. 6 Changes of average slot utilization rate of link under different conditions

迟存在较大差异,但是在所有组别的仿真实验中,与 TRS 调度方法相比,堆叠调度方法都减少了 TT 消息的端到端延迟,在该实验所设置的仿真参数下,减少的幅度在 0%~45% 之间。从图 6 可以看出,与 TRS 调度方法相比,堆叠调度方法可以减少链路平均时隙占用率,从而增加网络运行时链路上的空余时隙,当对事件触发消息进行在线调度时,可以增加可供调度的事件触发消息的长度或数量,由此可看出堆叠调度方法在提高网络的实时性和

灵活性方面有着显著的效果。

实际上,因为该实验只是从宏观上研究了消息数、网络芯片数等的影响,实际上端到端延迟还会受到每一条消息属性(长度、源节点、目的节点等)的影响。在堆叠调度方法中,如果能够堆叠较多的消息,效果会更明显,如果因为消息属性问题导致能够堆叠的消息较少(如同一链路上的 TT 消息刚好都是同一模式,或处于不同模式下的 TT 消息不在同一链路上传输),效果变差,因此图 5 和图 6 中

会出现明显的转折。为了更准确地看出堆叠调度方法对端到端延迟的影响,将所有实验结果按照使用 TRS 调度方法的总端到端延迟进行排序。图 7 中给出了随 TRS 调度方法的端到端延迟增加,堆叠调度方法的端到端延迟对应变化情况的拟合曲线。可以看出,当使用 TRS 方法调度的 TT 消息组端到端延迟增加时,使用堆叠调度方法的 TT 消息组端到端延迟也呈上升趋势,但堆叠调度端到端延迟增长速率要缓于 TRS 方法,因此可推断出总端到端延迟时间越长,堆叠调度减少延迟的效果越明显。

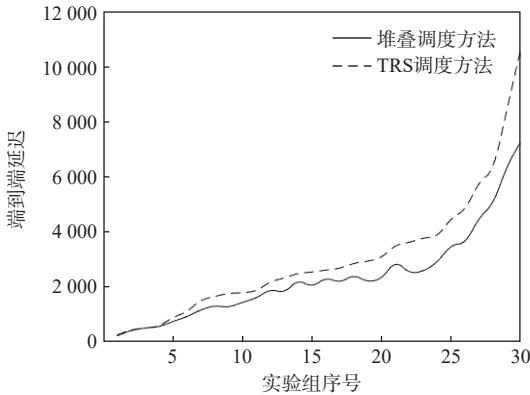


图 7 TRS 调度方法端到端延迟增加时堆叠调度方法端到端延迟增加趋势

Fig. 7 Increase trend of end-to-end delay of stack scheduling method, as end-to-end delay in TRS increases

更进一步,通过不同参数下堆叠调度方法减少端到端延迟的效果,定性研究影响使用堆叠调度方法调度 TT 消息必要性及影响堆叠调度方法效果的主要因素。图 5(a)中,在网络芯片数、操作模式数相同的条件下,随着 TT 消息数的增加,堆叠调度方法相较于 TRS 调度方法减少的端到端延迟呈现上升趋势;图 5(b)中,在 TT 消息数、操作模式数相同的条件下,随网络中芯片数的增加,堆叠调度方法相较于 TRS 调度方法减少的端到端延迟呈现下降趋势。由于增加 TT 消息数量和减少网络中芯片数量本质上都是增加了平均每条链路需要传输的 TT 消息数,据此对使用堆叠调度方法后端到端延迟减少量随不使用堆叠调度方法时平均每条链路传输 TT 消息数变化情况进行统计并拟合曲线。如图 8 所示,随平均每条链路所传递的消息数增加,堆叠调度方法减少的端到端延迟呈明显上升趋势。因此,可得出如下结论:芯片间互连网络中,当使用常规调度方法时,平均每条链路负责传输的 TT 消息越多,转为使用堆叠调度方法时相较于平均链路传输消息少的网络减少总端到端延迟的效果越明显,即使用堆叠调度方法调度消息的必要性就越高。

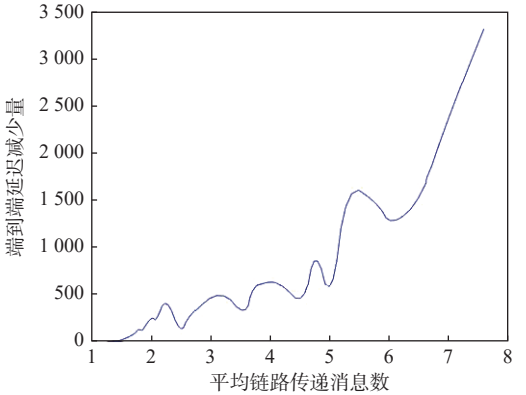


图 8 堆叠调度方法端到端延迟减少量随平均链路传递消息数变化

Fig. 8 Change of amount of end-to-end delay reduction in stack scheduling method varies with average number of messages delivered on link

此外,考虑到在使用堆叠调度方法的前提下,操作模式也会对调度效果产生影响。通过改变 3.1 节中一组 TT 消息不同的模式分布,使用堆叠调度方法对其进行调度,比较不同模式分布下总端到端延迟的变化情况。该组消息共有 150 条,当使用不考虑操作模式的 TRS 调度方法时,其总端到端延迟约为 4.7×10^4 ms,在改变模式分布的过程中,所有消息的周期、数据帧长、源节点、目的节点都不改变,也不会出现不同模式下消息重组为同一模式,以此来减少其他因素干扰。实验研究了 5~20 个操作模式下的总端到端延迟的变化情况,结果如图 9 所示。从结果可看出,其他条件不变时,随着操作模式数的增加,堆叠调度方法下的端到端延迟呈现先陡峭后平缓的减少趋势,因此可得出如下结论:使用堆叠调度方法调度 TT 消息时,对于一组 TT 消息,其操作模式数越多,使用堆叠调度方法后的总端到端延迟越短。此外,通过与 TRS 调度方法端到端延迟比较,可以看出若设定除操作模式外所有属

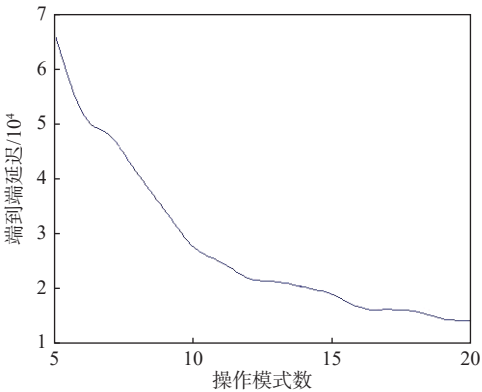


图 9 堆叠调度方法端到端延迟随操作模式数变化

Fig. 9 Change of end-to-end delay in stack scheduling method varies with number of messages' modes

性都相同,在操作模式数较少时,堆叠调度方法延迟减少效果不一定优于 TRS 调度方法。这是因为调度顺序改变而导致了一些“剪枝”的提前或延后进行,从而可能使得部分消息传输的跳数与对应延迟增加,其增加量在操作模式数较少时,在某些情况下会大于堆叠调度方法减少的排队延迟;同时在硬件方面,使用堆叠调度方法需要在芯片内增加对应的模式切换消息处理模块,这对芯片的设计提出了更高的要求,这些都是使用堆叠调度方法应考虑到因素和成本,在实际工程中应加以考虑。

4 结 论

1) 建立了多操作模式情况下的芯片间互连网络与时间触发消息模型,提出了适用于芯片间互连网络的模式转换方法和步骤。

2) 提出了芯片间时间触发消息堆叠调度方法。在不同操作模式下分别根据路径选择策略和 TT 消息严格可调度原则为 TT 消息确定传输路径和发送时间偏移量,并通过重置不同模式下 TT 消息调度表和继承上一模式拓扑的方式保证 TT 消息的堆叠和每个芯片使用的端口数符合限制。

3) 仿真实验结果表明,芯片间时间触发消息堆叠调度方法能够随着网络规模的变化,不同程度减小网络中 TT 消息的端到端延迟和链路平均时隙占用率,从而提高网络的实时性和灵活性。其中,与 TRS 调度方法相比,在模式较多及硬件支持的情况下,堆叠调度方法可以减少 TT 消息的端到端延迟,其减少效果与 TRS 调度方法调度结果的端到端延迟和平均每条链路传递消息数呈正相关。此外,堆叠调度方法的端到端延迟随着网络中消息的操作模式数增加而减少。当链路上可堆叠的消息较多时,可以显著减少调度结果的端到端延迟。

参考文献 (References)

- [1] WOLFIG R, JAKOVljevic M. Distributed IMA and DO-297: Architectural, communication and certification attributes[C]// IEEE/AIAA 27th Digital Avionics Systems Conference. Piscataway: IEEE Press, 2008: 1. E. 4-1.
- [2] 何锋. 机载网络技术基础[M]. 北京: 国防工业出版社, 2018: 9-15. HE F. Fundamentals of airborne network[M]. Beijing: National Defense Industry Press, 2018: 9-15 (in Chinese).
- [3] 蒲小勃. 现代航空电子系统与综合[M]. 北京: 航空工业出版社, 2013: 70-86. PU X B. Modern avionics system and integration[M]. Beijing: Aviation Industry Press, 2013: 70-86(in Chinese).
- [4] AHMADIAN H, OBERMAISSER R, PEREZ J. Distributed real-time architecture for mixed criticality systems[M]. Boca Raton: Taylor & Francis Group, 2018: 51-57.
- [5] PAUKOVITS C, KOPETZ H. Concepts of switching in the time-triggered network-on-chip[C]//2008 14th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications. Piscataway: IEEE Press, 2008: 120-129.
- [6] KOPETZ H, STEINER W. Real-time systems: Design principles for distributed embedded applications[M]. 2nd ed. Berlin: Springer International Publishing, 2011: 326-337.
- [7] DURRIEU G, FOHLER G, GALA G, et al. Dreams about reconfiguration and adaptation in avionics[C]//Proceedings Embedded Real Time Software and Systems. Toulouse: HAL Science Ouverte, 2016: 48-57.
- [8] URBINA M, OBERMAISSER R. A gateway core between on-chip and off-chip networks for an AUTOSAR message-based multi-core platform[C]//Automotive Meets Electronics. Dortmund: VDE Verlag GmbH, 2016: 1-6.
- [9] OLIVER R S, CRACIUNAS S S. Hierarchical scheduling over off- and on-chip deterministic networks[J]. ACM SIGBED Review, 2016, 13(4): 14-19.
- [10] STEINER W. An evaluation of SMT-based schedule synthesis for time-triggered multi-hop networks[C]//Real-Time Systems Symposium. Piscataway: IEEE Press, 2011: 375-384.
- [11] CRACIUNAS S S, OLIVIER R S. SMT-based task- and network-level static schedule generation for time-triggered networked systems[C]//22nd International Conference on Real-Time Networks and Systems (RTNS 2014). New York: ACM, 2014: 45-54.
- [12] 郑重, 何锋, 李浩若, 等. 基于贪婪随机自适应搜索法的TTE通信调度算法[J]. 北京航空航天大学学报, 2021, 47(11): 2268-2276. ZHENG Z, HE F, LI H R, et al. Scheduling algorithm of TTE network based on greedy randomized adaptive search procedure[J]. Journal of Beijing University of Aeronautics and Astronautics, 2021, 47(11): 2268-2276(in Chinese).
- [13] 孔韵雯, 李峭, 熊华钢, 等. 片间综合化互连时间触发通信调度方法[J]. 航空学报, 2018, 39(2): 258-267. KONG Y W, LI Q, XIONG H G, et al. Time-triggered communication scheduling method for off-chip integrated interconnection[J]. Acta Aeronautica et Astronautica Sinica, 2018, 39(2): 258-267(in Chinese).
- [14] 汪晶晶, 李峭, 熊华钢, 等. 芯片间时间触发通信综合规划方法及其优化[J]. 北京航空航天大学学报, 2020, 46(1): 170-180. WANG J J, LI Q, XIONG H G, et al. Integrated planning method and optimization for off-chip time-triggered communication[J]. Journal of Beijing University of Aeronautics and Astronautics, 2020, 46(1): 170-180(in Chinese).
- [15] MOUTINHO L, PEDREIRAS P, ALMEIDA L. A real-time software defined networking framework for next-generation industrial networks[J]. IEEE Access, 2019, 7: 164468-164479.
- [16] CRACIUNAS S, SERNA OLIVER R S. Online incremental scheduling method for deterministic networks: US20170070439[P]. 2017-03-09.
- [17] HEILMANN F, SYED A, FOHLER G. Mode-changes in COTS time-triggered network hardware without online reconfiguration[J]. ACM SIGBED Review, 2016, 13(4): 55-60.
- [18] SYED A. Model-based design and adaptive scheduling of distributed real-time systems[D]. Kaiserslautern: Technical University of Kaiserslautern, 2018: 60-63.
- [19] CHEN J C, DU C L, XIE F, et al. Schedulability analysis of non-

preemptive strictly periodic tasks in multi-core real-time systems[J]. Real-Time Systems, 2016, 52(3): 239-271.

[20] TANG X Q, LI Q, ZUO Y J, et al. Online schedule of sporadic life-critical traffic in TTEthernet[C]//2019 IEEE 5th International Conference on Computer and Communications (ICCC). Piscataway: IEEE Press, 2020: 1312-1316.

Stacking scheduling method for time-triggered messages in off-chip network

ZANG Guangjie^{1, 2}, LI Qiao^{2, *}, WANG Tong², XIONG Huagang²

(1. Chinese Aeronautical Establishment, Beijing 100029, China;
2. School of Electronic Information Engineering, Beihang University, Beijing 100191, China)

Abstract: To ensure the strict time determinism of message transmission between avionics communication tasks, time-triggered (TT) communication methods are also applied to the off-chip interconnection network. When avionics tasks have multiple operation modes, the time-triggered schedules between chips belonging to different modes will overlap and occupy time slots. The stacking scheduling method for time-triggered messages in off-chip network, which can reduce the queuing delay of application layer messages due to the waiting TT time window, is proposed to improve the flexibility and efficiency of using network resources,. Simulation experiments show that compared with the super scheduling method, the stack scheduling method can reduce the total end-to-end delay of TT messages in the off-chip interconnection network and the average time slot occupancy rate of the link. For scenarios where the end-to-end delay time is long and the average link bearer message transmission is large, the effect of using the stack scheduling method to reduce the end-to-end delay is more significant.

Keywords: off-chip interconnection; time-triggered message; operation mode; stacking scheduling ; end-to-end delay