doi:10.3969/j.issn.1671-637X.2017.06.016

时间触发总线时钟同步技术研究

陈长胜[1], 刘智武[1], 李晓庆[1], 季 雷[1], 杨 洁[2]
(1.中航工业西安航空计算技术研究所,西安 710068; 2.中航工业西安飞行自动控制研究所,西安 710065)

摘 要:时间触发协议(TTP)总线,或称时间触发总线,是一种用于安全关键实时控制系统的总线,采用基于全局时间基的时间触发通信机制。针对时钟同步高精度、容错的要求,研究了时间触发总线时钟同步算法,在OMNeT++仿真环境下建立了节点机模型和时钟同步算法模型,构建了时间触发总线系统模型,对时钟同步算法进行仿真,验证了同步算法。在同步周期足够的情况下,可以达到1 μs以内的同步精度。

关键词:时间触发协议; 总线; 时钟同步算法; 仿真
中图分类号:TP393.0 **文献标志码**:A **文章编号**:1671-637X(2017)06-0074-05

Synchronization Algorithm of Time Triggered Protocol Bus

CHEN Chang-sheng[1], LIU Zhi-wu[1], LI Xiao-qing[1], JI Lei[1], YANG Jie[2]
(1. Xi'an Aeronautics Computing Technique Research Institute, AVIC, Xi'an 710068, China;
2. Xi'an Flight Automatic Control Research Institute, AVIC, Xi'an 710065, China)

Abstract: Time-Triggered Protocol (TTP), or time-triggered bus, is a bus used in safety-critical, real-time control system adopting the time-triggered communication mechanism based on global time. Considering the requirements of clock synchronization on high precision and fault tolerance, we studied the algorithm of clock synchronization of TTP, established a TTP node model and the algorithm of clock synchronization by using OMNeT++. Simulation was made to the algorithm of clock synchronization and verified the correctness of it. With enough synchronization period, the precision can be less than 1 μs.

Key words: time-triggered protocol; bus; synchronization algorithm; simulation

0 引言

实时控制系统是指在系统规定的时间间隔内,调节或强制被控制对象完成预定动作或做出及时响应,以实现对被控对象控制的系统。民用飞机中包含了众多的实时控制系统,例如飞行控制系统、发动机控制系统、机电管理系统等。随着计算机技术、通信技术和控制技术的快速发展,民用飞机实时控制系统经历了从传统机械式传动到电传的过程,目前正逐步朝着网络化的分布式控制系统方向发展,传感器、控制器、作动器等各个单元之间采用通信总线进行连接和数据的传输。

实时控制系统执行计算、通信等任务是由触发条件控制的,如果这个触发条件是一个事件的发生则称为事件触发,而如果触发条件是一个指定时间点的到达则称为时间触发。在分布式实时控制系统中,采用时间触发模式能够将可用的通信资源合理地静态分配给系统内每个节点,通过时间触发的协议排除了事件触发通信协议要考虑的资源共享冲突和恢复操作等问题,提高系统的确定性和安全性。近年来,控制系统中出现了基于ARINC659总线[1]以及基于SAE AS5643的MIL-1394B总线[2]等多种时间触发的架构。而时间触发协议(TTP)总线[3]由于传输速率较高、拓扑灵活,具有较好的安全性支持,且成本低,适用于网络化实时控制系统的数据传输,成为一种广受关注的技术。欧洲已经成功开发了TTP协议芯片、工具链等产品,并在航空、航天、汽车、工业等领域成功应用。其中在航空领域,TTP已经应用于GE的F110发动机FADEC,A380的舱内压力控制系统[4]以及B787的动力控制系统和环境控制系统[5]。

TTP采用总线型架构[6],基于时分复用(TDMA)的

收稿日期:2016-11-22 **修回日期**:2017-03-01
基金项目:国家"十二五"项目(MJZ-2014-S-47)
作者简介:陈长胜(1982—),男,安徽芜湖人,硕士,高工,研究方向为机载网络技术。

通信调度策略，总线上各个节点按照预先定义的消息描述表(MEDL)，基于全局时间基进行数据传输的调度，如图 1 所示。由于控制系统高安全性的需求，TTP 总线提出了一种分布式的高精度时钟同步算法[7]。通过多个授时节点同时发出授时帧对网络进行同步，避免了传统的集中式时钟同步技术中的授时端单节点故障问题；通过达到亚微秒的高精度同步过程，提高传输过程中帧到达的“准点率”，使得由于同步偏差导致的时间预留尽可能减小，从而提高了系统通信的效率。

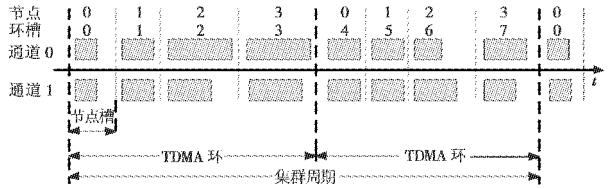

图 1　介质访问策略

Fig. 1　Media access scheme

本文对 TTP 总线的同步协议进行研究，提出 TTP 总线时钟同步实现方案，并通过仿真的方式对该方案进行验证，确保其能够满足基于时间触发架构的实时控制系统对于高精度、容错的时钟同步的要求。

1　时钟同步协议

1.1　参数定义

由图 1 可以看到，TTP 调度大周期是集群周期，MEDL 中定义一个集群周期内的总线行为；每个集群周期内可以划分为多个调度的小周期，即 TDMA 环，TDMA 环同时也是时钟同步算法执行的周期；每个 TDMA 环内包含了每个节点发送的槽，各个节点按照 MEDL 定义的槽进行帧的发送和接收。而通信调度的时间刻度采用 macrotick，它由一定数目的 microtick 组成。从时钟同步的角度来看，TTP 网络全局时间的基本参数包括 microtick，macrotick 和 precision。

microtick 是定义 TTP 控制器的最小测试时间刻度的周期性信号，时间长度 Δ_{mt} 关联到物理的 TTP 控制器时钟，实际值为输入时钟的周期。例如典型的设计中采用晶振的频率为 40 MHz 时，microtick 为 25 ns。

macrotick 是界定全局时间的周期性信号，被 TTP 控制器用来触发协议事件的执行，并确定 TTP 控制器发送或者接收一个帧的具体时间。每一个 macrotick 长度为 Δ_{MT}，由一定数目的 microtick 组成，在 MEDL 表的 Clock Setup 中配置 microticks/macrotick 参数，设定二者的比例关系。不同节点的 microtick 可以不相同，例如图 2 中的节点 A 和节点 B。但集群中所有 TTP 控制器应基于相同的 Δ_{MT} 值进行操作。

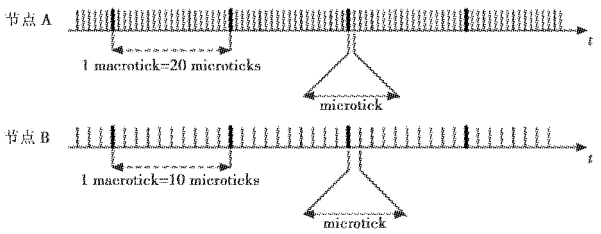

图 2　microtick 和 macrotick 的关系

Fig. 2　Macrotick-microtick relationship

由于各个节点的时钟存在漂移(例如，因为环境因素影响)，节点之间必然存在一定范围内的时钟偏差。采用参数 precision 定义全局时间基的精度，标记为 π。集群内所有节点的同步协议执行、集群操作都在一个可配置的时间间隔(精度)之内。π 是一个时间间隔，所有节点的 macrotick 都发生在这个间隔内，如图 3 所示。

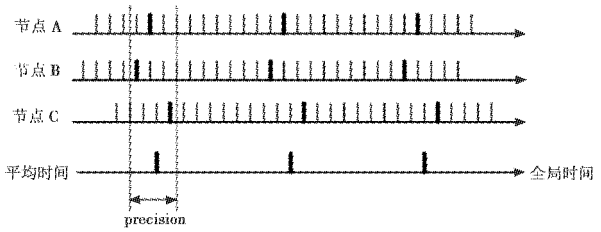

图 3　时钟同步精度

Fig. 3　Precision of clock synchronization

1.2　同步算法

传统网络上的时钟同步都是通过专门的授时帧来传输时钟信息，以达到同步的目的，例如基于以太网的 SNTP 和 IEEE1588[8] 等。TTP 总线并没有定义授时帧，而是通过测量总线上帧的实际传输时间、并与预期时间对比的方式，间接得到不同节点之间的时间差，再进行对时操作，实现整个总线上所有节点的时钟同步。因此，这种方法不需要额外的流量，具有更高的效率。

假定总线上有 n 个主时钟节点，能够容忍 k 个主时钟节点的故障，并且考虑到控制系统能够容忍拜占庭故障的需求，则应该满足 $n>3k$。时钟同步基于每个环进行，定义环中包含 s 个槽。在总线传输过程中的每个槽，针对采样帧实际到达时间与预期到达时间，计算时间偏差并求得相对时间值 C_i，直到环的最后一个槽。在最后一个槽收集齐计算同步时间，将相对时间值 $(C_1,\cdots,C_s)$ 从小到大排序，再计算 C_{k+1} 和 C_{s-k} 两个相对时间的平均值，用于本地时钟的修正，即

$$cnf(C_1,\cdots,C_s)=\frac{C_{k+1}+C_{s-k}}{2}。\qquad(1)$$

以下通过示例进行说明。例如总线上有 4 个节点并且都作为主时钟节点，标记为 N1 ~ N4，要求能够容忍 1 个主时钟节点故障。假定帧的传输延时都是 1 个

Δ_{MT}。MEDL 定义如表 1 所示。

表 1 总线 MEDL 定义

Table 1 MEDL of the bus Δ_{MT}个数

节点	槽长度	发送时间	预期接收时间
N1	[0,20]	15	16
N2	[20,45]	40	41
N3	[45,75]	70	71
N4	[75,100]	95	96

在发送时间点的绝对时间，各个节点的本地时间值如表 2 所示。

表 2 总线上各节点的本地时间

Table 2 Local clock of each node Δ_{MT}个数

绝对时间	N1 时间	N2 时间	N3 时间	N4 时间
15	16	15	15	14
40	39	43	42	40
70	70	71	69	70
95	94	95	94	96

根据上述信息，可以计算出各个节点发送和接收帧的时间点，如图 4 所示。

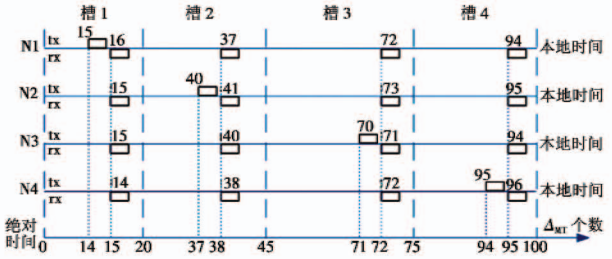

图 4 总线上帧收发的时间点

Fig. 4 Time point of transmission on the bus

由此可见，对于节点 N1，4 个槽上帧的实际接收时间分别是(16,37,72,94)，与表 1 中的预期接收时间做差值，并且按照升序进行排序，就可以得到(C_1,…,C_4)=(-4,-2,0,1)。由于 $k=1$，则 $cnf(C_1,\cdots,C_4)=\frac{C_2+C_3}{2}=-1$。

因此，节点 N1 应将其本地时间增加 1 个 Δ_{MT}，完成这个环的时钟同步，其他节点也分别按照同样的计算和同步方法调整本地时钟。

2 时钟同步过程

2.1 执行时间和时间信息交换

AT 表示为节点在一个槽内发送或者接收 TTP 帧的时间点，计划的和实际的 AT 时间差，建立了节点之间时钟同步的基础。图 5 描述了一个 TTP 帧的发送节点和接收节点之间时间关联的不同时序参数。图中发送节点的时钟比接收节点的时钟稍快。因此，相对于计算的时间 $t_{AT'r}$，接收节点实际的接收时间要稍微早一些。理想条件下，$t_{reception\ r}$与 $t_{AT'r}$严格发生在同一时间，而实际上这两个值之间存在偏差，标记为 Δ_{dif}。

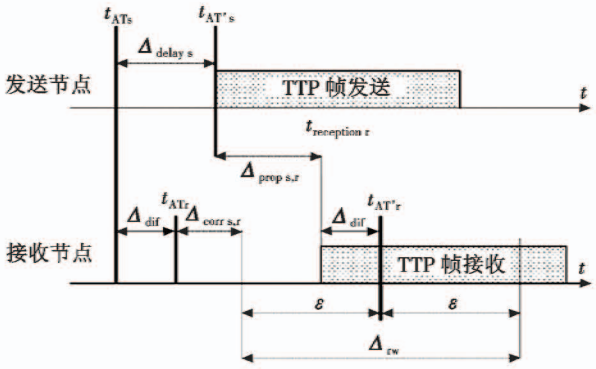

图 5 计算时间差值

Fig. 5 Time difference capturing

执行时间 t_{AT}是 TTP 控制器开始 TTP 帧传输(在其发送槽)或者预期另一个 TTP 控制器发送 TTP 帧的时间点，所有 t_{AT}的值根据配置数据操作，通过同步操作控制在一个精度范围内。图 5 所示的 Δ_{rw}定义了一个特定帧的预期接收时间相关的对称的有效接收范围，即接收窗口。由于两个节点之间最大偏移是在精度范围内，接收窗口最小定义为 2π。

有效帧的起始位置应在接收窗口内被检测到，没有在接收窗口内开始的接收被认为无效，基于无效帧获得的时间值不被用作时钟同步。

发送节点在本地时间 $t_{AT's}$开始数据传输，称为延时执行时间。这个时间是发送节点计划的本地时间加上一个发送的延时 $\Delta_{delay\ s}$。从发送节点的角度考虑，需要通过 $\Delta_{delay\ s}$来补偿接收节点因为接收窗口的延迟引起的延时。

接收节点预期的接收时间 $t_{AT'r}$的算式为

$$t_{AT'r}=t_{ATr}+\frac{\Delta_{rw}}{2}+\Delta_{corr\ s,r}\ 。\tag{2}$$

以下算式定义了基于执行时间 $t_{AT'r}$分别从发送节点和接收节点角度考虑的时序关系，在每个环槽的任一发送-接收节点之间实现。其中，帧传输时间 $\Delta_{prop\ s,r}$和 $\Delta_{corr\ s,r}$基于环槽定义，而 $\Delta_{delay\ s}$针对每个通道定义。

$$\Delta_{prop\ s,r}+\Delta_{delay\ s}=\Delta_{corr\ s,r}+\frac{\Delta_{rw}}{2}\tag{3}$$

$$\Delta_{prop\ s,r}\geqslant 0\ 。\tag{4}$$

接收节点以 microtick 的粒度测量帧的实际到达时间 $t_{reception\ r}$，并计算与预计到达时间的差值，结果使用计数器表示。

$$\Delta_{dif}=t_{reception\ r}-t_{AT'r}\ 。\tag{5}$$

如果节点的两个通道都接收到正确帧，计算两个 Δ_{dif}的平均值作为测量值。如果节点只有一个通道接收到正确帧，使用这个 Δ_{dif}作为测量值。如果该槽配置用于时钟同步，则保存上述测量值，用于计算修正值。

2.2 修正值的计算

在每一个 TDMA 环(即同步周期)中,每个 TTP 控制器都以 microtick 为单位,测量帧的预期到达时间和实际到达时间之间的差值 Δ_{dif},用于计算修正值。

全局时钟的重同步在每个重同步间隔内,在所有节点上同时执行。在一个 TDMA 环中,至少执行一次时钟修正,执行时钟修正的时间点在配置数据中定义。

考虑到容错时钟同步的集群中,需至少 4 个主节点的槽,最小重同步间隔也应至少是 4 个槽,在总线配置定义。

时钟状态修正值(CSCT)基于保存的测量值进行计算,按照同步算法中的描述,取其中两个值进行平均,求出平均值后,与 MEDL 表定义的时钟修正门限进行比较。如果超出门限值则表示时钟偏差太大,无法进行同步,向主机报告同步错误,进入 Freeze 状态,并停止操作。

2.3 修正本地时钟

在 TTP 系统中,时钟同步必须保证每个节点的时钟偏移在精度范围内。每个重同步间隔中,经过时钟修正值的计算后,TTP 控制器可以根据配置使用以下的一个策略调整本地的 macrotick。

1) 单次调整:在一个 macrotick 内调整 CSCT 个 microtick。

2) 逐步调整:每个 macrotick 增加/减少一个 microtick,直到 CSCT 耗尽。

3) 间隔调整:每隔几个 macrotick,增加/减少一个 microtick,直到 CSCT 耗尽。

例如,要调整 -5 个 Δ_{mt},可以采用图 6 所示的 3 种方案。

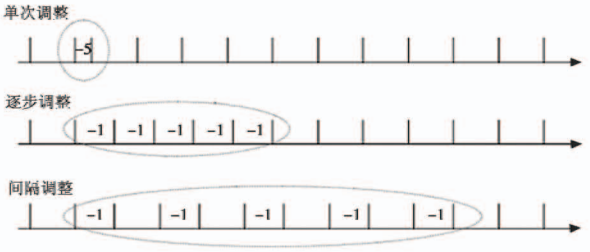

图 6　时钟同步调整方法示例

Fig. 6　Example of synchronization correction

3 时钟同步仿真

基于上述算法,进行 TTP 总线的时钟同步仿真,仿真目的是:1) 验证同步算法有效性;2) 观察不同参数配置情况下的同步精度;3) 验证部分节点故障时,总线是否能够继续保持同步状态。

仿真过程中使用的参数包括同步周期(即 TDMA 环)、macrotick 和节点所用晶振的频率稳定度。

3.1 仿真场景设计

设计 8 个节点构成总线型拓扑,每个节点都是主时钟节点,容忍 3 个故障节点。晶振标称频率 40 MHz,考虑到温度、电源电压、负载阻抗和大气压力变化及机械振动等[9]因素导致的频率漂移,假定频率稳定度 50 ppm,也即频率范围是 40 MHz ± 2000 Hz。仿真设定的频率值如表 3 所示。

表 3　总线上各节点本地时钟频率

Table 3　Frequency of local clocks　　MHz

节点	频率	节点	频率
TTPN1	40.002	TTPN5	40.000
TTPN2	40.001	TTPN6	40.000
TTPN3	40.000	TTPN7	39.999
TTPN4	40.000	TTPN8	39.998

仿真过程中设计 4 种场景。

1) 场景 1:不执行同步算法。

2) 场景 2:定义 Δ_{MT} 为 200 个 Δ_{mt},即 5 μs。每个槽长度为 100 个 Δ_{MT},即 0.5 ms。每个 TDMA 环包含 8 个槽,即同步周期为 4 ms。

3) 场景 3:定义 Δ_{MT} 为 1000 个 Δ_{mt},即 25 μs。每个槽长度为 100 个 Δ_{MT},即 2.5 ms。每个 TDMA 环包含 8 个槽,即同步周期为 20 ms。

4) 场景 4:TTPN1 ~ TTPN3 工作一段时间后故障,之后再恢复。

3.2 仿真结果分析

OMNeT ++ 是一款开源的、基于组件的、模块化的开放网络仿真平台[10]。基于 OMNeT ++ 仿真环境,实现了时间触发总线节点模型和时钟同步算法,采用单次同步的方式,构建了 8 个 TTP 节点的时间触发总线系统,如图 7 所示。

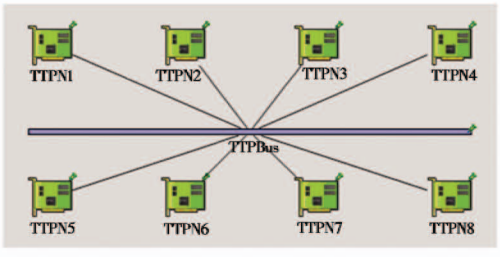

图 7　仿真拓扑结构图

Fig. 7　Topology architecture of simulation

场景中的所有节点同时启动,即本地时间都是同时从 0 开始,各个节点按照自己的时钟频率进行通信调度,并基于通信调度表和帧的实际接收时间执行时钟同步算法,观察频率漂移最大的两个节点(TTPN1 和 TTPN8)之间的时钟与真实时钟的差值。仿真目的是验证时间触发总线同步的效果和同步精度。

针对上述3种场景分别进行仿真,结果见图8。

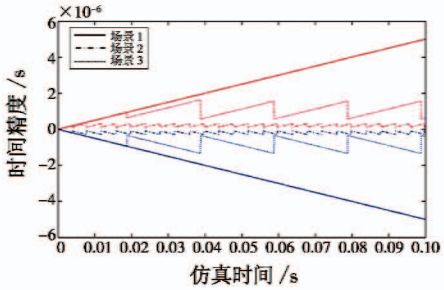

图8 时钟同步仿真结果

Fig. 8 Result of simulation

仿真结果如下。

1)场景1:不执行同步算法时,节点之间的时钟偏差会越来越大,系统无法保持同步状态。

2)场景2:同步周期为4 ms时,节点之间能够保持时钟同步,同步精度为0.29 μs。

3)场景3:同步周期为20 ms时,节点之间能够保持时钟同步,同步精度为1.34 μs。

4)场景4:当TTPN1 ~ TTPN3工作一段时间后故障,之后再恢复时,总线仍然能够保持同步状态。

由此可见,时间触发总线时钟同步算法能够有效实现总线上不同节点之间的时钟同步,并且具有分布式容错的特性,部分节点故障不会导致总线时钟同步的故障。由于该同步方法不需要额外的授时帧,因此具有效率高的特点。实际应用中,同步周期越小同步精度越高,而同步周期本身也是总线通信调度的TDMA环,因此,设计中应当对总线进行合理的配置,使其既可以满足通信调度的要求,又能够保持其同步精度在合适的范围之内。

4 结束语

本文深入研究了时间触发总线中的时钟同步算法,并在OMNeT++环境下实现了时间触发总线节点模型和时钟同步算法,针对不同的场景分别进行了时钟同步算法和同步精度的仿真验证,结果表明该算法能够在不需要授时帧的情况下有效进行时钟同步。

参考文献

[1] 张喜民. ARINC 659背板数据总线协议初探[J]. 电光与控制,2013,20(3):93-97.

[2] 任齐凤,楼俊荣,贺轶斐. MIL-1394b总线的确定性[J]. 航空电子技术,2015,46(3):34-39.

[3] HERMANN K, GUNTER G. TTP-A protocol for fault-tolerant real-time systems[J]. IEEE Computer, 1994, 27(1): 14-23.

[4] TTTech. Airbus A380-TTP based cabin pressure control system[EB/OL]. [2016-11-03]. https://www.tttech.com/markets/aerospace/projects-references/airbus-a380.

[5] TTTech. Boeing 787-TTP based communication platform [EB/OL]. [2016-10-28]. https://www.tttech.com/markets/aerospace/projects-references/boeing-787.

[6] 赵罡,何锋,王红春,等. 航空电子环境TTP/C总线应用技术研究[J]. 航空计算技术,2014,44(6):110-115.

[7] 刘冬冬,张天宏,陈建,等. TTP/C协议的关键特性研究[J]. 计算机测量与控制,2012,20(10):2769-2772.

[8] LEE K, EIDSON J C, WEIBEL H, et al. IEEE 1588 standard for a precision clock synchronization protocol for networked measurement and control systems[C]//Sensors for Industry Conference, IEEE, 2002:98-105.

[9] 沈连丰. 信息与通信工程原理与实验[M]. 北京:科学出版社,2007.

[10] 杨光旭,刘方爱,赵学臣. OMNeT++平台上无线传感器网络仿真系统的研究[J]. 计算机应用研究,2011,28(9):3443-3446.

(上接第73页)

控件开发[J]. 微处理机,2013(6):42-45.

[5] 石磊,张瑞平. 用VAPS XT与Open GL进行三维视景开发[J]. 电光与控制,2015,22(3):97-100.

[6] 邓勇. 虚拟座舱显示与视景仿真技术[D]. 西安:西安电子科技大学,2015.

[7] 周涛,李璐. 民用飞机电子飞行仪表仿真系统设计与实现[J]. 电子设计工程,2014,22(22):160-163.

[8] 丁继伟. 某型通用航空飞机地面飞行仿真技术研究[D]. 哈尔滨:哈尔滨工业大学,2012.

[9] 马存宝,朱超,王彦松. 飞行管理系统模块的仿真[J]. 工业仪表与自动化装置,2014(2):42-45.

[10] 邱伟龙,陈国兴. 飞机虚拟仪表建模仿真关键技术研究[J]. 飞机设计,2014,34(3):49-54.