

引用格式:严思哲,龚龙庆,刘露,等. 基于 P2P 透明时钟的环形网络时钟同步方法[J]. 微电子学与计算机,2020,37(10):69-74. [YAN S Z,GONG L Q,LIU L,et al. Clock synchronization method in ring network based on P2P transparent clock [J]. Microelectronics & Computer,2020,37(10):69-74.]

基于 P2P 透明时钟的环形网络时钟同步方法

严思哲,龚龙庆,刘 露,徐丹妮
(西安微电子技术研究所,陕西 西安 710054)

摘 要: 由于非对称网络的特殊性,在运用 1588v2 协议进行时钟同步时存在困难. 本文剖析了协议难以兼容的非对称网络的原因,重点分析了 1588v2 在非对称环形网络下的使用缺陷,并对协议原算法进行了仿真模拟. 本文在避免使用 delayAsymmetry 修正值的基础上,设计了一套基于 P2P 透明时钟的非对称环网同步算法,并进行了实验验证. 实验结果表明,该算法提高了同步精度,使 1588v2 协议更好地适用于环形网络.

关键词: 时钟同步;IEEE1588v2;delayAsymmetry;环网

中图分类号: TN915.04 **文献标识码:** A **文章编号:** 1000—7180(2020)10—0069—06

Clock synchronization method in ring network based on P2P transparent clock

YAN Si-zhe,GONG Long-qing,LIU lu,XV Dan-ni
(Xi'an Microelectronics Technology Institute, Xi'an 710054,China)

Abstract: Due to the particularity of the asymmetric network structure, there are difficulties in using the protocol for clock synchronization. This article analyzes the reason why the protocol is difficult to be compatible with asymmetric network. It focuses on the defects of using 1588v2 in the asymmetric ring network and gives simulations of original algorithm based on the protocol. Thus, a set of P2P transparent clock-based clock synchronization algorithms for asymmetric ring network is designed on the basis of avoiding the use of delayAsymmetry correction values. Experimental results show that the algorithm improves the synchronization accuracy and makes the 1588v2 protocol better applicable to ring networks.

Key words: clock synchronization;IEEE1588v2; delayAsymmetry; ring network

1 引言

《网络测量、控制系统精准时钟同步协议标准》(Precision clock synchronization protocol for networked measurement and control systems^[1], PTPv2) IEEE 1588v2 发布于 2008 年^[2],其基本原理是采用主(主时钟, Master)从(从时钟, Slave)时钟机理,主从时钟之间交互同步信息包,记录信息包的发出和接收时间,并对每一条信息包“加盖”时戳. 有了时戳,系统利用网络的对称性计算出主从时间

偏差,并利用该偏差调整从时钟时间,完成向主时钟的同步.

同步报文从出发到到达目的地所消耗的时间称为线路传输延迟,该延迟分为两部分:在链路上传输时产生的链路延时、经过中间设备时产生的驻留延时. 这两种延时会降低同步精度. 1588v2 克服了上述延时问题,此外还应用了硬件时戳技术,使时戳的精度大大提升,因此它比传统应答式同步的精度更高. 协议所用的报文称为 PTP 报文,分为两种:事件报文(报文中加盖有精确时戳,包括 Sync、Delay_

Req, Pdelay_Req 和 Pdelay_Resp) 和普通报文(不用加盖精确时戳, 包括 Announce、Follow_Up、Delay_Resp、Management、Signaling 等)^[3-4].

本文的研究背景是实时网络通信项目, 系统要求实现通信功能和同步功能. 通信报文和同步报文会在网络中产生冲突, 因此, 即使报文来回经过的链路相同, 期间遭遇的排队、等待情况也是不同的, 导致报文来回消耗的时间不同, 故本项目的网络是非对称网络. 对非对称网络, 1588v2 的解决方法是引入修正值 delay Asymmetry^[4]. 但正是由于报文在经过中间设备时会产生排队、冲突等不可预测的延迟, 导致报文每次传输的 delay Asymmetry 值都不同. 因此该值不固定, 不能做到预设, 在工程实际中难以使用.

为解决上述问题, 本文设计了一种不需使用 delay Asymmetry 值的环形非对称网络时钟同步算法, 并进行了实验验证, 对 1588v2 协议做出了一定扩展.

2 1588v2 下的非对称环网同步算法及其缺陷

2.1 P2P 透明时钟与 P2P 同步机制

1588v2 提出了透明时钟(Transparent Clock, TC)的概念. 透明时钟用于测量报文的驻留时间^[5], 并写入报文或其跟随报文(Follow_Up)的修正域中. 协议中给出了两种透明时钟模型: 端到端透明时钟(E2E TC)和对等透明时钟(P2P TC)^[6].

P2P TC 的每个端口都有一个额外模块, 作用是测量本端口与链路上游某一类型端口之间的链路延时^[4]. 这样, 各节点之间就没有了主从之分, 在链路中是对等的^[7]. 如图 1 所示, 这里以 [Master-TC-Slave] 结构为例介绍 1588v2 的 P2P 同步流程:

(1) Master 向 Slave 发送 Sync 报文(注: 下文所有的时刻都是指报文所在设备的本地时间), 报文于 $T_{\text{sync_in}}$ 时刻到达 TC, 于 $T_{\text{sync_out}}$ 时刻出 TC, Slave 在 T_{s0} 时刻收到 Sync 报文.

(2) TC 在收到 Sync 报文后, 于 T_{t1} 时刻向 Master 发送 Pdelay_Req 报文, Master 于 T_{m1} 时刻接收到此报文. 随后 Master 向 TC 回复 Pdelay_Resp 报文, 该报文的出发、到达时间分别是 T_{m2} 、 T_{t2} .

(3) Slave 在收到 Sync 报文后, 于 T_{s1} 时刻向 TC 发出 Pdelay_Req 报文, 该报文于 T_{t3} 时刻到达 TC. 随后, TC 于 T_{t4} 时刻向 Slave 回复 Pdelay_Resp 报文, 该报文于 T_{s2} 时刻到达 Slave.

设主从时钟偏差为 T_{offset} , 驻留时间为 $T_{\Delta S}$, TC

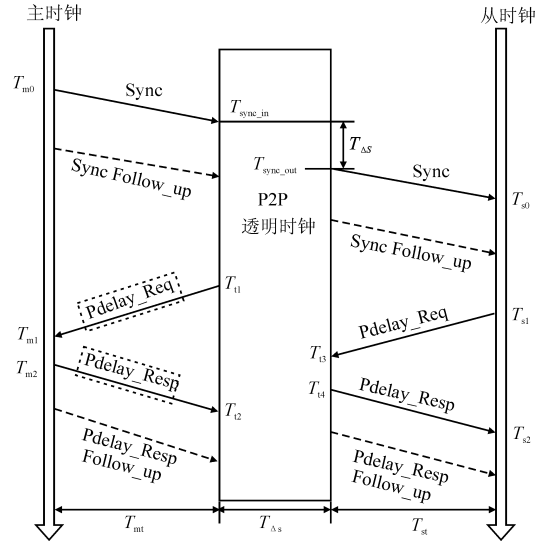


图 1 P2P 同步机制

与 Master、Slave 之间的链路延时分别是 T_{mt} 和 T_{st} , 有:

$$T_{\Delta S} = T_{\text{sync_out}} - T_{\text{sync_in}}$$

假设报文从 Master 到 TC 经过的链路与从 TC 回到 Master 的链路一致, 有:

$$T_{\text{mt}} = (T_{t2} - T_{m2} + T_{m1} - T_{t1})/2 \quad (1)$$

同样假设 Slave 到 TC 的链路与 TC 到 Slave 的链路一致, 则:

$$T_{\text{st}} = (T_{s2} - T_{t4} + T_{t3} - T_{s1})/2 \quad (2)$$

$$T_{\text{offset}} = T_{s0} - T_{m0} - T_{\text{mt}} - T_{\text{st}} \quad (3)$$

这样就得到了主从时钟偏差 T_{offset} , Slave 根据此值修正本地时间, 就完成了向 Master 的同步.

2.2 不对称路径与 delayAsymmetry 修正值

1588v2 同步是利用了网络的对称性, 协议规定: 在任何网络路径上的系统中, 主时钟到从时钟的报文和从时钟到主时钟的报文应通过同一网络路径^[4]. 针对不对称网络, 协议引入了修正值 delayAsymmetry, 如图 2 所示, t_{ms} 指报文从 Master 到 Slave 消耗的时间, t_{sm} 指报文从 Slave 返回 Master 消耗的时间. meanPathDelay 表示两者的均值, 即 $\text{meanPathDelay} = (t_{\text{ms}} + t_{\text{sm}})/2$.

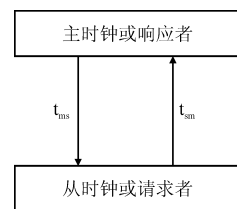


图 2 不对称路径

报文从 Master 到 Slave 的路径与从 Slave 返回 Master 的路径虽然一致,但消耗的时间不一定相同,delayAsymmetry 就代表来回时间之差,定义如下:

$$\begin{aligned} t_{ms} &= \text{meanPathDelay} + \text{delayAsymmetry} \\ t_{sm} &= \text{meanPathDelay} - \text{delayAsymmetry} \\ \text{则 } \text{delayAsymmetry} &= (t_{ms} - t_{sm})/2 \quad (4) \end{aligned}$$

2.3 环形网络

环形网络即是各设备首尾连成环形^[8]. 下图为 4 设备环形网络,各设备时间均不相同,需确定某设备为 Master,其余设备为 Slave,Master 时间不变,所有 Slave 修正自身时间轴,向 Master 同步.

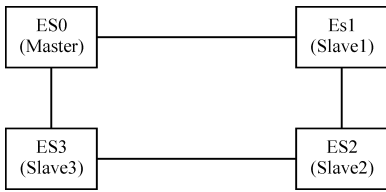


图 3 环形网络

2.4 1588v2 下的非对称环网同步缺陷

为说明 1588v2 下的环网同步缺陷,这里以图 3 所示的环形网络为例,取 Slave3 分析.

设同步报文从 Master 出发时,Master 本地时间为 t_{m1} ,到达 Slave3 的时刻为 t_{s1} ;随后 Slave3 向 Master 回复同步报文,报文的出发、到达时刻分别为 t_{s2} 、 t_{m2} . 设 Slave3 与 Master 之间的时钟偏差为 T_{offset} ,有:

$t_{s1} - t_{m1} = t_{ms} + T_{\text{offset}}$, $t_{m2} - t_{s2} = t_{sm} - T_{\text{offset}}$ 代入式(4)得:

$$T_{\text{offset}} = \text{delayAsymmetry} + (t_{s1} - t_{m1} + t_{s2} - t_{m2})/2 \quad (5)$$

因此,要计算出 T_{offset} ,就需要预设 delayAsymmetry 值. 但 1588v2 协议规定,对该值的测量不在协议范围内^[4]. 为说明该值的缺陷,下面以沿顺时针方向同步为例进行分析.

报文从 Master 到 Slave3 所消耗的时间 t_{ms} 可分为三部分:

(1) 在线路上传输的时间 $T_{\text{dm-s}}$.

(2) 被中间设备转发所消耗的时间,称为转发时间 T_t . 转发过程一般包括解析报文内容、修改报文内容、加盖时戳等行为,这些行为的消耗时间在设计中是固定的,是一个恒定值. 由于 Master 和 Slave3 之间有两个中间设备 ES1 和 ES2, 故为 $T_{\text{tES1m-s}}$ 和 $T_{\text{tES2m-s}}$;

(3) 如果中间设备自身有发送任务,则报文在转发之前还需要等待,该延时称为等待时间 T_w . 本网络中同时存在同步报文和通信报文,两种报文会产生冲突,排队等待情况无法预测,因此本设计中的 T_w 是非固定值,此处为 $T_{\text{wES1m-s}}$ 和 $T_{\text{wES2m-s}}$.

$$t_{ms} = T_{\text{dm-s}} + T_{\text{tES1m-s}} + T_{\text{tES2m-s}} + T_{\text{wES1m-s}} + T_{\text{wES2m-s}}$$

同理,从 Slave3 到 Master 的时间 t_{sm} 为:

$$t_{sm} = T_{\text{ds-m}} + T_{\text{tES1s-m}} + T_{\text{tES2s-m}} + T_{\text{wES1s-m}} + T_{\text{wES2s-m}}$$

由于来回路径一致,故 $T_{\text{dm-s}} = T_{\text{ds-m}}$,代入式(4)有:

$$\text{delayAsymmetry} = (T_{\text{tES1m-s}} + T_{\text{tES2m-s}} - T_{\text{tES1s-m}} - T_{\text{tES2s-m}})/2 + (T_{\text{wES1m-s}} + T_{\text{wES2m-s}} - T_{\text{wES1s-m}} - T_{\text{wES2s-m}})/2,$$

根据上式,delayAsymmetry 值大体上可以分为两部分:来回的转发时间之差与等待时间之差. 上文已经提到,转发时间可以预估,而等待时间受中间设备当前发送任务、线路占用状态等因素的影响,无法预估,这就使式(5)中的 T_{offset} 不可计算,从而影响同步精度,甚至影响同步的完成.

3 改进算法

为解决此问题,本文设计了一种不需使用 delayAsymmetry 值的非对称环网同步算法,流程如下图所示.

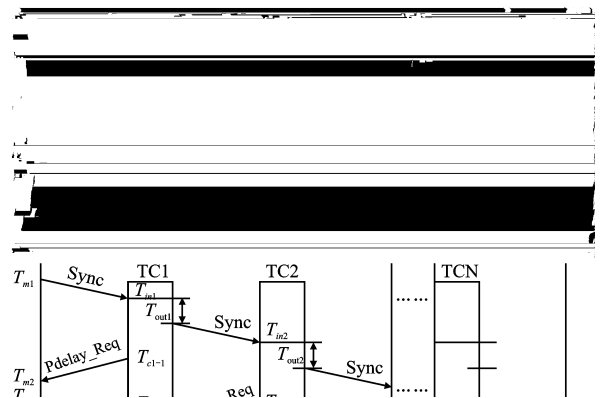


图 4 同步示意图

算法分为 5 个阶段:

(1) 驻留时间测量

Master 从任一端口发出 Sync 报文. 各 TC 从某一端口收到报文后,测量驻留时间,并将驻留时间写入报文修正域,再将报文从另一端口转发出去.

(2) 链路延时分段测量

本节的主要思想是将整段链路划分成一段一段首尾相连的短链路,分段测量链路延时.具体步骤为:将 Sync 报文转发出去后,各 TC 设备从接收到 Sync 报文的端口发出 Pdelay_Req 报文,与上游设备交互 Pdelay_Req 和 Pdelay_Resp 报文,分段测量链路延时. Master 和各 TC 在收到 Pdelay_Req 报文后,只解析报文内容而不再转发,TC 和 Slave 在收到 Pdelay_Resp 报文后,也只解析报文内容,不再转发.

环形网络中一般都存在环网转发的情况,即报文会在环网中转发一周,直至到达源设备才会停止.这种环网转发的目的是避免产生信息遗漏,但会增加报文交互,使链路上的排队、等待情况变多,降低同步精度.为使 Pdelay_Req 报文和 Pdelay_Resp 报文不进行多余转发,本文设计了一种可以避免环网转发的算法,具体方式为:①每个设备在生成 PTP 报文时,在报文头的 flag 域设置一个 step 位,初值为 0,报文每经过一个设备,该值就加 1.②设备在收到报文后,先检测报文类型,如果是 Pdelay_Req 或 Pdelay_Resp 报文,则检测其 step 值,如果为 0,说明报文没有经过转发,则解析报文内容,执行链路延时的测量步骤;如果 step 不为 0,说明报文经过了转发,已经完成了某一段链路延时的计算,报文已经完成了自身使命,则丢弃该报文.这样就避免了 Pdelay_Req 和 Pdelay_Resp 报文的环网转发造成精度下降的问题.

(3) 链路延时累加

在上一节中对链路延时进行了分段测量,分段的延时计算结果储存在该段链路的终点设备中,第一段(Master 至 TC1)储存在 TC1 中,第二段储存在 TC2 中,直至最后一段储存在 Slave 中.本节的目的是将这些分段的延时累加,得到完整的延时,并发送给 Slave.

为累加链路延时,本文引入一种新的事件报文,称为 Delay_Sync.为了与现有报文区分开,在该报文的报文头 messageType 域中填入值 0x04(协议中,0x00 到 0x03 代表现有的事件报文,0x04 到 0x07 预留).报文内容与 Sync 报文基本一致,区别在于:Sync 报文修正域中存储的是各设备的驻留时间,而 Delay_Sync 报文的修正域中还需储存各设备计算好的链路延时.

本节详细步骤为:

Master 定期发出 Delay_Sync 报文,每个设备在接收到后,首先检测报文的 messageType,如果值

为 0x04,则将报文中的链路延时与本地储存的延时值作比较;如果不同,则将报文中的延时值累加到本地延时值中,然后将这个新的延时值写入报文,替换报文中的旧延时值;如果两者相同,则不做修改.最后从另一端口将报文转发出去.如果收到报文时,链路延时的计算尚未完成,则链路延时一栏省缺不填.通过这样一级一级累加的方式,所有 Slave 都能得到自身与 Master 间的总链路延时.

当一个设备连续多次收到的 Delay_Sync 中的延时值一致,说明前方设备都已完成链路延时的累加,则本设备立即执行步骤(4).

当 Master 连续两个周期收到的 Delay_Sync 报文中的延时值一致,说明链路中所有设备都已完成链路延时的计算,Master 停止发送 PTP 报文.

(4) 时钟偏差计算

如图 1 所示,式中 $t_{c(i)}$ 为第 i 个 TC, $t_{c(i)-k}$ 为该 TC 时钟轴上的第 k 个时刻点; $T_{\text{delay}(i)}$ 为第 i 段链路延时; $T_{\Delta m}$ 为第 n 个透明时钟的驻留时间; T_{offset} 为时钟偏差. Slave 据此修正本地时间,完成向 Master 的同步.

(5) 同步校验

由于之前经过了完整的同步过程,各节点已将自身与 Master 之间的链路延时储存在本地,因此不需要重新计算链路延时,只需 Master 向外发送一次 Sync 报文,测量驻留时间. Slave 收到后,得到路径上的驻留时间,再配合本地保存的链路延时,就可以重新进行偏差计算,完成同步校验.

4 实验验证

4.1 delayAsymmetry 值的不确定性验证

为验证 delayAsymmetry 值的不确定性,及影响其变化的因素,本文基于时间触发网络对使用 delayAsymmetry 值的同步算法(如 2.3、2.4 节所述)进行仿真验证.该时间触发网络基于全网节点同步,以周期为单位定时进行数据通信和时钟同步.本文用 Verilog 语言设计同步引擎,搭建了实验平台,以 1 ms 为周期让三个 Slave 与 Master 进行同步,同时实现 4 个节点相互通信的功能,并搭建验证平台进行仿真验证.

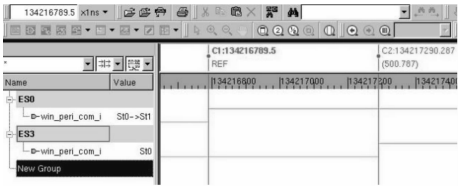
(1) 仿真条件

硬件配置为 CPU Intel i3, 4GRAM. 仿真环境为 Linux 平台下的新思公司仿真软件 VCS. 本文设计转发时间 T_f 为 220 ns,等待时间 T_w 无法设置,仿真环境下报文在链路上传输的时间 T_d 为 0. 编写测试

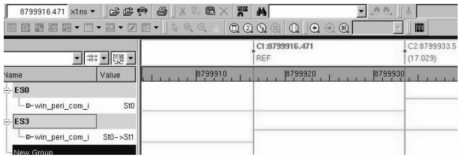
用例,选定 ES0 为 Master,其余 3 设备为 Slave.取 ES3 为研究对象.共进行三次仿真测试.

表 4.1 仿真条件			
T_d	T_t	T_w	中间设备是否有转发任务
0	220ns	不确定	不确定

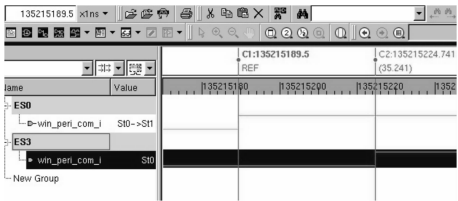
(2)仿真结果



(a)第 1 次仿真



(b)第 2 次仿真



(c)第 3 次仿真

图 5 仿真结果图

三次仿真结果如图 5.在设计中,本地时钟累加到预设的周期时间后会归零.为方便观测时钟信号,在本地时钟的 0 时刻点设计了一定宽度的测试信号,当本地时钟归零时,测试信号置 1.测试信号上升沿即本地时钟 0 时刻点,因此测试信号上升沿的偏差即是 Master 和 Slave3 的同步偏差.

ES0 代表 Master,ES3 代表 Slave3,两者波形上升沿(图中红色竖线处)之差就代表本地时钟偏差.其中第二次测试时,中间设备 ES1 和 ES2 恰好没有发送任务,同步报文没有经过转发,因此此次测试的 T_w 为 0.三次时钟偏差分别为 500.787 ns、17.029 ns和 35.241 ns.

4.2 改进算法验证

为验证第 3 章所述改进算法的可行性,对其进行实物验证.验证载体为沿形验证系统,验证环境为实时以太网.网络具备同步与通信功能,且同步与通信都按预设的周期时间进行.本算法应用于网络的同步模块中.

(1)实验条件

设计验证平台如图 6 所示,拓扑中有 4 个包含 2 个端口的独立时钟节点,各节点均具备透明时钟功能.在该网络拓扑中执行算法,取 ES0 为 Master,剩余设备为 Slave,取 Slave3 为研究对象,同步与通信的周期取 1 ms,测试时长为 2 min.

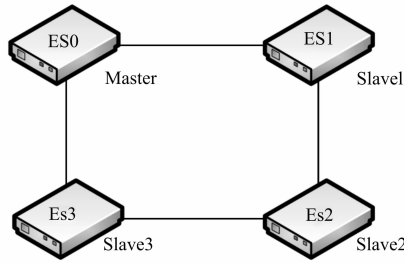


图 6 验证平台

(2)实验结果

使用示波器对 Master 和 Slave3 的时钟信号进行多次测试,结果如图 7:

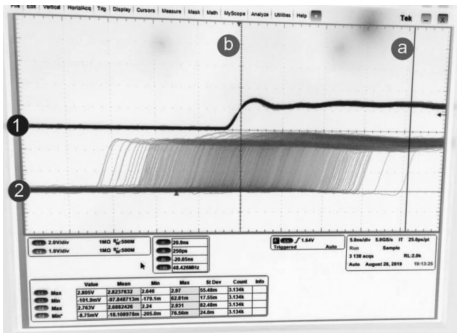


图 7 测试波形图

图中波形同样是为方便观测本地时钟而设计的标志信号,上升沿代表本地时钟 0 时刻点.

图中线 1 为 Master 时钟测试信号,线 2 为 Slave3 时钟测试信号,其中线 b 表示 Master 的 0 时刻点.可以看到,经过多次同步后,Master 的 0 时刻不变,而 Slave3 的 0 时刻点与 Master 的偏差始终保持在一定范围内,最大偏差出现在线 a 处,差值为 20.9 ns.

实验结果表明,在非对称环网中运用本文提出的同步算法,同步精度为 20.9 ns,达到亚微秒级别,算法可行.

4.3 实验结论

表 2 实验数据对比表

使用修正值的算法同步偏差			改进算法 最大同步偏差
第 1 次仿真	第 2 此仿真	第 3 次仿真	
500.787 ns	17.029 ns	35.241 ns	20.9 ns

表 2 为使用 delayAsymmetry 值的同步算法仿真验证结果与改进算法的实物验证结果. 其中第 2 次仿真时,中间设备自身没有发送任务,直接转发同步报文, T_w 为 0,其余 2 次仿真中 T_w 不为 0.

结论 1:第 1 次仿真与第 3 次仿真的同步精度均不如第 2 次仿真,这说明报文在中间设备的等待过程会影响 delayAsymmetry 值,进而降低同步精度.

结论 2:第 1 次仿真与第 3 次仿真的同步精度彼此之间存在较大差异,这说明报文在中间设备中的等待时间是不确定的. 因此 delayAsymmetry 值不固定.

结论 3:改进算法的同步精度为 20.9 ns,比第 1 次仿真和第 3 次仿真的同步精度都更高,这说明本文提出的改进算法可行,并且可以提高同步精度.

5 结束语

本研究设计了一套无需 delayAsymmetry 值的非对称环网同步算法,提高了同步精度;提出了一种及时丢弃 Pdelay_Req 报文和 Pdelay_Resp 报文的方式,以减少环网转发带来的等待延迟,达到提高同步精度的目的;引入一种新型报文 Delay_Sync 累加链路延时,以提高算法完成度. 未来可以将本算法扩展到多种非对称网络结构中,提高算法适应性.

本文分析了 1588v2 协议应用于非对称网络时的缺陷,提出了改进算法,使原本需要使用 delayAsymmetry 值的非对称环网同步算法不再受制于该值,并进行了实验验证. 实验结果表明,改进算法的同步精度高于原算法,可以提高同步精度,使协议在环形网络下的适应性得到了提升.

参考文献:

[1] LI S X. Research and implementation of 2-layer ethernet swith based on IEEE 1588v2[D]. WEI H; Shan-

dong University,2014.

[2] LIU S P. Research on the application of IEEE 1588 in high-availability seamless redundancy [D]. Tianjin: Tianjin University,2013.

[3] IEEE 1588-2002. IEEE Standard for a precision clock synchronization protocol for networked measurement and control systems[S],2002.

[4] AQSIQ,SAC. GB/T25931-2010. Precision clock synchronization protocol for networked measurement and control systems[S]. Beijing: Standards Press of China,2011.

[5] CUI Q S, BAI X M, HUANG B Y, et al. Test method for the resident time error of P2P transparent clock and its implementation on the clock tester[J]. Power System technology,2017,41(1):322-327.

[6] CHEN S, ZHAO S. Analysis on IEEE1588v2 time synchronization principles and application [J]. High Speed Railway Technology,2015,6(3):80-85.

[7] GUO M F, LIN S M, LIN B, et al. MINS/GPS tightly-coupled integrated navigation system[J]. Journal of Chinese Inertial Technology,2011,19(2):214-219.

[8] ZHANG H W. Feature comparison among commonly used topological strutures in Industrial ethernet[J]. Inner Mongolia Electric Power,2006(24):93-96.

作者简介:

严思哲 男,(1995-),硕士研究生. 研究方向为嵌入式系统. E-mail:15129933391@163.com.

龚龙庆 男,(1962-),研究员,博士生导师. 研究方向为空间计算机技术.

刘 露 女,(1986-),硕士,高级工程师. 研究方向为实时以太网通信技术.

徐丹妮 女,(1989-),硕士,工程师. 研究方向为实时以太网通信技术.