文章编号:1001 – 893X(2012)05 – 0800 – 04

嵌入式 Linux 下 IEEE 1588 时间同步的实现*

易胜蓝

(中国西南电子技术研究所,成都 610036)

摘 要:在分析 IEEE 1588 精确时间同步协议(PTP)原理的基础上,设计了包括最佳主时钟算法和 PTP 协议的时钟同步模型。针对嵌入式 Linux 操作系统,提出了在 Linux 的网络驱动层通过在收发以太网帧时完成时间戳的接收和添加的方法。实验结果表明,通过该方法能够达到10 μs量级的同步精度,较好地实现了时钟同步的效果。

关键词:嵌入式 Linux;时间同步;PTP;IEEE 1588;最佳时钟算法

中图分类号:TN915;TP393 **文献标志码**:A doi:10.3969/j.issn.1001 – 893x.2012.05.039

Implementation of IEEE 1588 Time Synchronization under Embedded Linux

YI Sheng-lan

(Southwest China Institute of Electronic Technology, Chengdu 610036, China)

Abstract: The IEEE 1588 Precise Time synchronization Protocol(PTP) is analysed, then a time synchronization model which includes the best master clock algorithm and PTP protocol is given. Based on Embedded Linux Operation System, a method of adding time stamp when the MAC frame is received in the driver layer is proposed. The experimental results show that this method meets the demands of time synchronization and 10 μs level synchronization precision can be achieved.

Key words: embedded Linux; time synchronization; PTP; IEEE 1588; best master clock algorithm

1 引 言

目前,用于时间同步的协议中使用得较多的是网络时间协议(Network Time Protocol,NTP),这是一种低成本的网络对时协议。但是,由于网络时间协议的时间标记(Time Stamp)是在协议的应用层中获得的,无法满足太高的时间精度要求,只能提供毫秒级别的对时精度[1]。与 NTP 协议不同的是,IEEE 1588精确时钟同步协议(Precision Time Synchronization Protocol,PTP)可以在不增加硬件成本的条件下,达到几微秒到几十微秒级别的时间同步精度[2],通过网络接口在协议的底层获取时间标记,达到时钟同步的效果。如果需要更高的精度要求,可以使用专门的硬件模块获取时间标记,将时间戳的获取点移至物理层,由硬件完成时间戳的获取,则可以将时钟同步精度提高到亚微秒级[3]。本文在对 IEEE 1588 的同步原理详细分析的基础上,提出一种在嵌入式 Linux 系统中以软件方式实现精准时间协议(PTP)的方法,并给出了相应的应用平台实现。

2 时钟同步原理

IEEE 1588 通过时间报文交换的方法实现分布

* 收稿日期:2011 – 11 – 01;修回日期:2012 – 04 – 09

式系统中各节点的时间同步。在系统中往往会有一个主时钟(master),它提供整个系统的时间基准,而系统中的其他时钟则为从时钟(或叫做子时钟,slave)。在系统中,每个时钟状态的确定是通过最佳时钟算法确定的。主时钟和从时钟通过交换报文的方式来确定主从时钟之间的时间偏移以及报文传输的网络延迟,PTP 协议通过两次包含有时间戳的报文的发送,真正精确的发送时间被正确记录了下来,用来最终计算出主时钟和从时钟之间的时间差。主时钟每隔一段时间将本地时间发布到网络上,从时钟进行时间戳的接收并更新本地时钟,同时从时钟不断地进行线路延时的计算,以保证时钟的精确[4]。

如图 1 所示,主时钟节点按照定义好的时间间隔,将时间同步报文(Sync) 发送给网络上所有的从时钟节点,该时间同步报文发送的精确时间戳 T_{M1} 被同时记录了下来,该同步时间戳在随后发送的跟进报文(Follow - up)中被发送给从时钟节点。从时钟节点在收到该时间同步报文后,接收时间戳 T_{S1} 被记录了下来,T_{M1}、T_{S1}这两者的时间偏差包括传输线路中的报文传输延迟和主从时钟的偏差。由于分布式系统中各个节点的布线长度、布线方式和该节点在网络中位置的不同,时钟报文的传输延迟也将会变得不同。为了提高时钟同步的精度,必须通过测量的方式消除该时钟传输延迟。从时钟节点以固定的时间间隔将延迟请求(Delay Request)报文发送给主时钟节点,同时将该报文的发送时间戳 T_{S2} 记录下来,主时钟节点在收到该延迟请求报文的同时,将接收时间戳 T_{M2} 记录下来,然后主时钟节点在随后的延迟响应 (Delay Response)报文中将该时间戳发送给相应的从时钟节点。

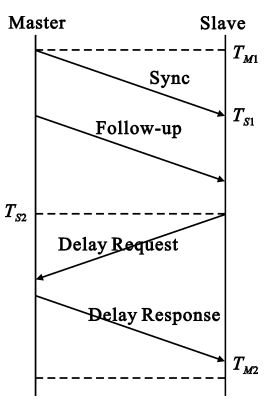

图 1　主从时钟同步原理

Fig. 1 Time synchronization principle of master and slave

在计算延迟的过程中,假设网络传输介质是对称均匀的,因此可以认为主时钟到从时钟的时间延迟跟从时钟到主时钟的时间延迟是对称相等的。因此,可以得出以下两个式子:

$$T_{S1} - T_{M1} = T_d + T_\Delta \tag{1}$$

$$T_{M2} - T_{S2} = T_d - T_\Delta \tag{2}$$

式中,T_Δ 为从时钟相对主时钟的偏差,T_d 为报文的网络传输延迟。

由式(1)和式(2)可以求得:

$$T_\Delta = [(T_{S1} - T_{M1}) - (T_{M2} - T_{S2})]/2 \tag{3}$$

$$T_d = [(T_{S1} - T_{M1}) + (T_{M2} - T_{S2})]/2 \tag{4}$$

根据计算出来的偏差 T_Δ,调节从时钟,即可实现时间同步的功能。

3　最佳时钟选择算法设计

在 IEEE 1588 精确时钟同步协议中,首先要从众多时钟当中找出一个时钟,作为系统的主时钟,其余时钟为从时钟。这个功能需要使用最佳主时钟算法来实现,最佳主时钟算法模块主要用于选择本地网络中的最佳时钟作为主时钟,同时决定本地时钟所应处的状态。最佳主时钟算法(Best Master Clock Algorithm) 由状态定断算法和数据集比较算法[1]这两部分算法组成。

数据集比较算法首先在所有同步报文的不同数据集中挑选出合格的同步报文,然后对这些同步报文进行筛选,以选出可用的最佳报文。其实现的流程如图 2 所示:它的作用是根据同步报文的不同数据集,通过从合格的同步报文里筛选,获得最佳的报文。在收到报文的时候,首先判断该报文是否是同步报文,是的话则继续运行;接着使用 Identifier_Compare()函数对时钟层和时钟标志位进行比较,如果这两种属性不同,则可以由此得出较好的时钟,如果这两者属性相同,则需要继续往下运行;进行时钟属性比较,如果时钟属性不同,可以得到较好的时钟,如果时钟属性也相同,则继续往下运行,使用 UUID_comparision()函数对不同的 UUID 进行比较,这样经过层层比较,所有端口都运行该算法,最终可以选出最佳的时钟。

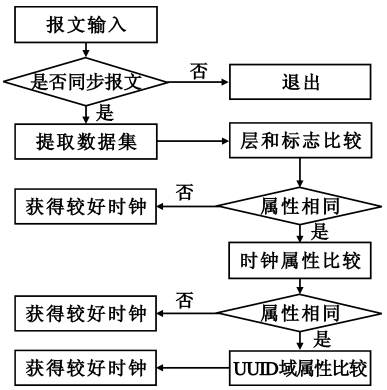

图 2 数据集比较算法流程

Fig.2 Data set comparation algorithm flow

状态定断算法主要进行状态的判断和数据集的更新,在确定最佳主时钟后,状态定断算法根据不同数据集的信息计算出系统中每个时钟的各个 PTP 端口的推荐状态,具体共有 8 种推荐状态:未校正、从时钟、待机、主时钟、监听、禁止、故障和初始化等状态,本地时钟将根据结果相应地调整状态。状态定断算法根据运算分析的结果,动态调整各个时钟和端口的状态,所以当主时钟出现性能下降或产生故障的时候,系统能够自动选择其他更合适的时钟作为主时钟。

4 嵌入式 Linux 操作系统移植

系统的应用平台选用基于 ARM9 的嵌入式系统,处理器芯片为三星的 ARM9 芯片 S3C2440,应用平台带有一块 256MB 的 Nand Flash 芯片用于存储 Bootloader、Linux 内核和根文件系统。嵌入式 Linux 操作系统是文中 IEEE 1588 协议实现的基础,为协议的运行提供运行的基础。

首先,根据应用平台的特性移植 Bootloader,本系统使用的 Bootloader 为 u-boot 1.1.6,其完成的主要任务有:为 Flash 分区,分配好系统的存储空间,分别为 u-boot 区、Linux 内核分区和根文件系统分区;初始化处理器和外设的硬件资源配置;传递启动参数给 Linux 内核,调用 Linux 操作系统。U-boot 的实现非常依赖于具体硬件,需要根据硬件配置将 U-boot 移植到嵌入式系统中。然后,根据应用平台的硬件特性配置 Linux 内核,并将 Linux 内核编译为 zImage 格式,应用系统选用的内核版本为 2.6.30.4,可以提供最新的应用特性。最后,使用 BusyBox 工具制作所需要的根文件系统。BusyBox 将系统中的许多功能模块集成到一个名叫 BusyBox 的可执行文件中,并通过使用不同的命令名称来调用相应的功能模块,十分容易定制自己的根文件系统。

将 Bootloader、Linux 内核和根文件系统编译完成后,即可通过 JTAG 工具烧写到 Nand Flash 的对应分区中。

5 PTP 协议软件实现

如图 3 所示为所设计网络时间同步系统的结构图,主要包含 PTP 协议处理模块、PTP 发送模块、PTP 接收模块和时间戳添加模块。其中 PTP 协议处理模块位于应用层中,使用 Linux 操作系统的网络 API 实现。PTP 发送模块和 PTP 接收模块使用 UDP 传输协议进行时间同步报文的发送和接收工作,时间戳的添加和读取工作在驱动层完成。

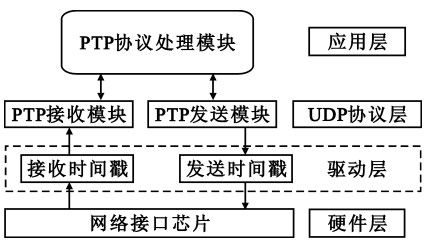

图 3 网络时间同步结构图

Fig.3 Network time synchronization structure

5.1 PTP 协议处理模块实现

PTP 协议处理模块是实际的运行模块,是整个精确时间同步协议实现部分的核心,它针对主从时钟节点的运行状态不同执行不同的任务。PTP 协议的接收、发送控制部分运行于 UDP 传输协议之上以实现 PTP 报文的接收和发送,主程序模块在应用层需要完成协议状态机的转化、报文计算、时钟同步计算、几个数据集和接收发送模块的控制。

为了在不使用专门硬件的情况下,能够达到最高的时间记录精度,记录时间戳的功能被添加在网络接口的驱动程序中,在网络接口的驱动程序中实现时间戳的精确添加。其中,主时钟节点按照定义好的时间间隔将同步时间报文(Sync)和 Follow-Up 报文发送给网络上所有的从时钟节点,并且对从时钟节点的延迟请求做出响应。同样,从时钟节点需要按照固定时间间隔将延迟请求报文发送给主时钟节点并接收来自主时钟节点的响应报文,根据该响应报文中的时间戳计算主从时钟的时间偏移和主从时钟节点间网络传输延迟,并根据时间偏移和传输

延迟更新本地时钟。其中时间同步报文(Sync)和延迟请求报文(Delay – Request)采用组播方式发送,跟进报文(Follow – Up)和延迟相应报文(Delay – Response)采用广播方式发送。

5.2　时间戳获取

PTP 协议处理模块实现的精度主要受到时间戳精度的影响,而报文时间戳的精确程度又主要是受同步算法、系统硬件等因素的影响。在与硬件特性相关的方面,网卡芯片的效率、晶振稳定程度、内存访问速度和 CPU 主频等都会对时间戳精度造成影响。在特定的硬件平台基础上,时间戳获取的位置又很大程度上影响了时间戳的准确性,时间戳可以在应用层、驱动层或者 Mac 与 PHY 之间获取。通过前面的报文传输过程分析可以看出,时间戳的获取越接近硬件底层,时间戳获取的时间误差就越小。因此,在精确时间协议实现的时候,时间戳的标记应该做到尽量接近硬件底层。

本文通过添加 Linux 操作系统的网络驱动层函数的方法来获取和添加时间戳。在同步报文发送的出口帧交给 MAC 控制器的时刻加上出口时间戳并保存起来,在同步报文接收的入口帧到达网络接口中断服务子程序的入口处加上时间戳并保存起来。

8	6	6	2	46-1 500	4
前导码	目的地址	源地址	类型	数据	校验码

图 4　以太网帧结构

Fig.4 Ethernet frame structure

以太网帧的典型结构如图 4 所示,包括源地址、目的地址、类型、数据、校验码和前导码。以太网帧是 OSI 参考模型数据链路层的封装,网络层的数据包加上以太网帧头和帧尾,构成可由数据链路层识别的数据帧。因此可以在收到本机网络层传递过来的数据后,在网络数据报头部加上时间戳的标记,在收到对方的以太网帧的时候,获取对方发送的时间戳参数。接收和发送时间戳任务的实现主要通过 add_stamp 和 get_stamp 这两个函数完成。add_stamp 为添加时间戳的函数,用于在发送同步报文的时候添加时间戳;add_stamp 的关键实现部分如下所示:

```
a = get_utime();
add_to_frame(time_t a,frame);
```

其中,get_utime 函数用于从系统中获得以 μs 为单位的时间,add_to_frame 函数将获得的时间参数以时间戳的方式添加到以太网帧中。

get_stamp 为获取时间戳的函数,在网络驱动层接收到同步报文的时候,用于从同步报文中获取对方发来的时间戳。其关键实现的函数为 utime_from_frame,该函数用于将从收到的以太网帧中获得的时间参数提取出来。

实际的测试表明,通过在网络驱动层添加时间戳的方法能够达到10 μs量级的同步精度。

6　结束语

本文从 IEEE 1588 协议出发,在嵌入式 Linux 操作系统中,对时钟同步协议进行详细分析,设计实现了最佳时钟选择算法和 PTP 协议处理模块,并对如何在网络驱动层添加时间戳进行了介绍。实验结果表明,通过该方法能够达到10 μs量级的同步精度,较好地实现了时钟同步的效果。

参考文献:

[1]　王晓冬,阚德涛,张志武.以太网的时钟同步技术[J].电子工程师,2008,34(9):47 – 50.
WANG Xiao – dong, KAN De – tao, ZHANG Zhi – wu. Clock synchronization technology for ethernet[J]. Electronics Engineer,2008, 34(9):47 – 50.(in Chinese)

[2]　鲁骏,张向利,范晓峰.嵌入式 Linux 下时钟同步系统的分析与实现[J].仪表技术与传感器,2009(3):64 – 66.
LU Jun, ZHANG Xiang – li, FAN Xiao – feng. Analysis and realize of clock synchronization under embedded Linux[J]. Instrument Technology and Sensor,2009(3):64 – 66.(in Chinese)

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

[4]　李聪,高丽.基于 IEEE 1588 的时钟同步技术在分布式系统中的应用[J].电子设计工程,2009,17(12):54 – 56.
LI Cong, GAO Li. Application of time synchronization technology based on IEEE1588 in distributed system[J]. Electronic Design Engineering,2009,17(12) 54 – 56. (in Chinese)

[5]　戴宝峰,崔少辉,常健.IEEE 1588 最佳主时钟算法的分析与实现[J].仪表技术,2008(2):29 – 31.
DAI Bao – feng, CUI Shao – hui, CHANG Jian. Analysis and realize of IEEE 1588 best main clock algorithm[J]. Instrument Technology, 2008(2):29 – 31.(in Chinese)

作者简介:

易胜蓝(1981—),女,湖南人,2003 年获工学学士学位,现为工程师,主要从事航空通信领域的研究工作。

YI Sheng – lan was born in Hunan Province, in 1981. She received the B.S. degree in 2003. She is now an engineer. Her research concerns aviation communication.

Email:sly_lan@163.com