跨域协同仿真工具总体设计文档

（版本 1.0）

OpenTSN 开源项目组

2022 年 2 月

目录

1 引言

1.1 编写目的

本文档为跨域协同仿真工具实现的概要设计文档，旨在对跨域协同仿真工具实现的总体设计进行概述。

1.2术语定义

- 仿真网络：由运行在仿真软件中的网络设备的硬件代码所构建的仿真的网络；
- 网络系统仿真环境：由端系统应用程序和仿真网络组成的网络系统环境；
- 软件域：运行在操作系统上的软件应用程序，模拟端系统上的控制或应用行为；
- 仿真域：运行在仿真软件中的硬件代码程序，模拟网络中的硬件设备行为；
- 交互文本：用于软件域内应用程序间、或软件域与仿真域间通信的文本；
- 跨域协同仿真工具：包含软件域、仿真域和交互文本的网络系统仿真环境；
- 软件域系统时间（t_{cpu_time}）：软件应用程序获取的当前系统时间；
- 仿真域系统时间（t_{mod_time}）：运行在仿真软件中的晶振所产生的时间计数信息；

- sync_pkt(t_{cpu_time})：同步报文，且携带有软件域系统时间（t_{cpu_time}）信息；
- sync_pkt(t_{mod_time})：同步报文，且携带有仿真域系统时间（t_{mod_time}）信息；
- 软件时间文本（cpu_time_file）：记录软件域系统时间（t_{cpu_time}）的文本；
- 异常事件文本（trap_file）：记录硬件设备异常事件的文本；
- tx_file：记录软件域的应用程序向仿真域发送的报文；
- rx_file：记录软件域的应用程序接收仿真域发送的报文；
- 时间信息文本：记录软件域与仿真域之间交互的同步报文；
- 应用数据文本：记录软件域与仿真域之间交互的业务数据报文；
- sim_enable：时间使能信号；
- ticker_counter：晶振时间计数器；
- gettimeofdaytxt()函数：OpenTSN 项目组自定义函数，基于软件时间文本获取软件域的系统时间；
- gettimeofday()函数：Linux 系统函数，获取 CPU 的系统时间信息。

2 跨域协同仿真工具概述

为解决网络芯片测试验证阶段上板调试周期长、调试复杂等问题，OpenTSN 项目组拟设计并开发一套跨域协同仿真工具。

典型的网络系统仿真环境组成如图 1 所示，包括应用程序和仿真网络。应用程序是软件实现的，模拟端系统上的应用。仿真网络是由运行在仿真软件中的网络设备的硬件代码所构建的仿真的网络，模拟网络中的硬件设备行为。网络系统仿真环境需要将软件应用程序与仿真网络集成在一起，从而实现整个网络系统的仿真。

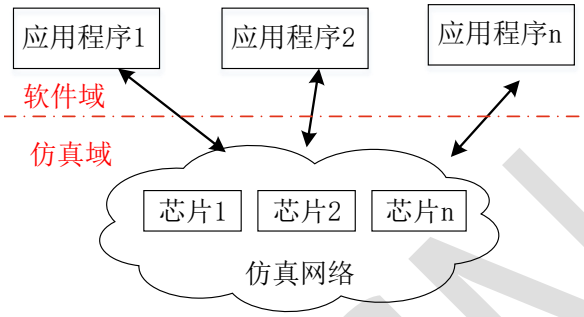

图 1 典型的网络系统仿真环境组成

关于网络系统仿真环境的实现，有两个关键的问题需要解决：

（1）仿真环境中软件应用程序与仿真网络之间如何进行通信

一般来说，软件域中的软件应用程序是使用 C 语言等高级编程语言实现，以进程的形式运行在操作系统中。仿真域中的仿真网络是使用 verilog 等硬件编程语言实现，运行在网络仿真软件中。软件应用程序与仿真网络集成在一起形成网络系统的仿真，软件应用程序与仿真网络之间如何通信交互是关键。

（2）如何保证仿真环境中软件域与仿真域的系统时间能同步，且同步精度在Δt 之内

在网络系统仿真环中，软件域中的软件应用程序是运行在真实的操作系统中，因此软件域中的系统时间为 CPU 主板时间。仿真域中的仿真网络是运行在网络仿真软件中，仿真域的系统时间是网络仿真软件的时间。CUP 主板时间和网络仿真软件的时间，一般来说相差比较大，在普通的 PC 机上，咫尺比大概是 100:1。也就是说软件域中的 100 秒，在仿真域中仅是 1 秒钟。

在一些应用场景中，如软件应用需周期性的往网络中发送报文。这种情况下，软件域是需要与仿真域的系统时间保持同步的，否则就会造成网络拥塞。因此，网络系统仿真环境中如何保证软件域与仿真域的系统时间能同步，且同步精度在Δt 之内，也是很关键。

针对上述两个问题，OpenTSN 项目组所设计的跨域协同仿真工具的解决方案是：（1）仿真环境中软件应用程序与仿真网络之间通过文本进行交互通信；（2）软件域与仿真域的系统时间通过“互锁机制”实现跨域时间同步。

3 总体设计

3.1工作原理

3.1.1 跨域协同仿真工具模型

跨域协同仿真工具模型如图 2 所示，跨域协同仿真工具可分为三部分：软件域、仿真域和交互文本。软件域包括时间管理进程以及一些其他的软件应用程序，仿真域包括时间同步管理控制和仿真网络。软件域的软件应用程序模拟端系统上的控制或应用行为，仿真域的仿真网络模拟网络中的硬件设备行为。软件域与仿真域之间通过文本进行交互通信。

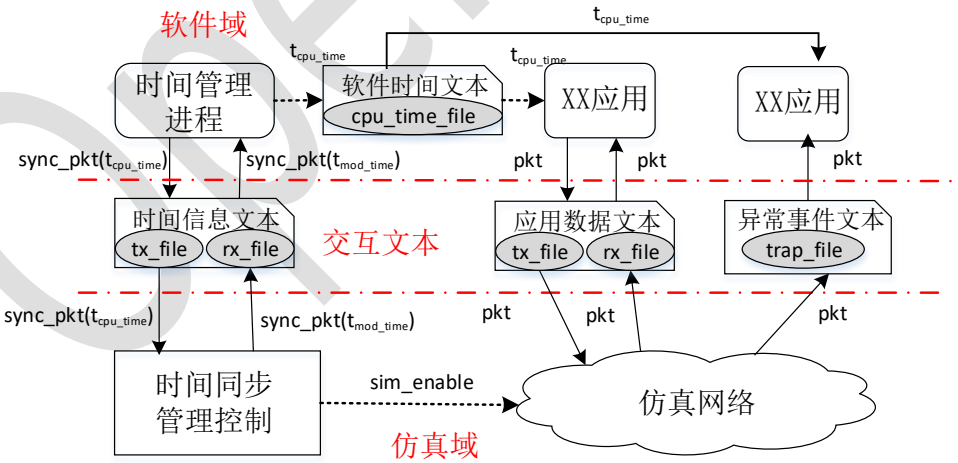

图 2 跨域协同仿真工具模型

相比典型的网络系统仿真环境，跨域协同仿真工具增加了交互文本，以实现仿真环境中软件域与仿真域之间的交互通信。软件域增加时间管理进程，仿真域中增加时间同步管理控制逻辑，以保证跨域协

同仿真工具中的软件域与仿真域的系统时间能同步，且同步精度在Δt之内。

3.1.2 系统时间同步流程

软件域和仿真域系统时间同步流程大致如下：

（1）仿真域中的时间同步管理控制逻辑，周期性的往时间信息文本（rx_file）写入携带有仿真域系统时间（t_{mod_time}）信息的同步报文sync_pkt(t_{mod_time})；

（2）软件域中的时间管理进程从时间信息文本（rx_file）中读取同步报文sync_pkt(t_{mod_time})，提取仿真域系统时间（t_{mod_time}）信息，然后将仿真域系统时间（t_{mod_time}）信息写入软件时间文本（cpu_time_file）中作为软件域系统时间（t_{cpu_time}），供其他的软件应用程序获取当前系统时间。再将当前的软件域的系统时间（t_{cpu_time}）封装成报文sync_pkt(t_{cpu_time}）写入时间信息文本（tx_file）；

（3）仿真域中的时间同步管理控制逻辑从时间信息文本（tx_file）中读取报文sync_pkt(t_{cpu_time}），提取软件域的系统时间信息（t_{cpu_time}）。然后将软件域的系统时间（t_{cpu_time}）与仿真域系统时间（t_{mod_time}）进行比较。若$|\ t_{mod_time} - t_{cpu_time}\ |$差值在Δt之内，不做处理；若大于Δt，则时间同步管理控制逻辑向仿真网络发送时间停止信号，直到差值小于Δt。

3.1.3 软硬件交互

软件域与仿真域之间、软件域内的时间管理进程与其他的应用程序之间，都是通过交互文本进行通信。交互文本具有如下特点：

（1）交互文本共分为四类文本，分别是软件时间文本、异常事件文本、时间信息文本和应用数据文本。软件时间文本记录软件域系统时间，异常事件文本记录硬件设备异常事件，时间信息文本记录软件域与仿真域之间交互的同步报文，应用数据文本记录软件域与仿真域之间交互的业务数据报文。

（2）端系统与仿真网络设备相连的每个端口，都有两个交互文本。tx_file文本为软件域的应用程序向仿真域发送报文，rx_file文本为软件域的应用程序接收仿真域发送的报文；

（3）针对一个端系统上运行多个应用程序场景，由于多个端系统应用程序将共用同一个文本发送报文，共用同一个文本接收报文。因此，应用程序在往 tx_file 文本写入报文时，需要加文件锁；在往 rx_file 文本读取报文时，需要增加对报文的过滤判断。

3.2仿真域的设计

3.2.1 系统结构

如图 3 所示，仿真域主要包含两大部分：同步控制逻辑和仿真网络。同步控制逻辑负责实现跨域协同仿真工具中的软件域与仿真域的系统时间同步，仿真网络负责模拟网络中的硬件设备行为。

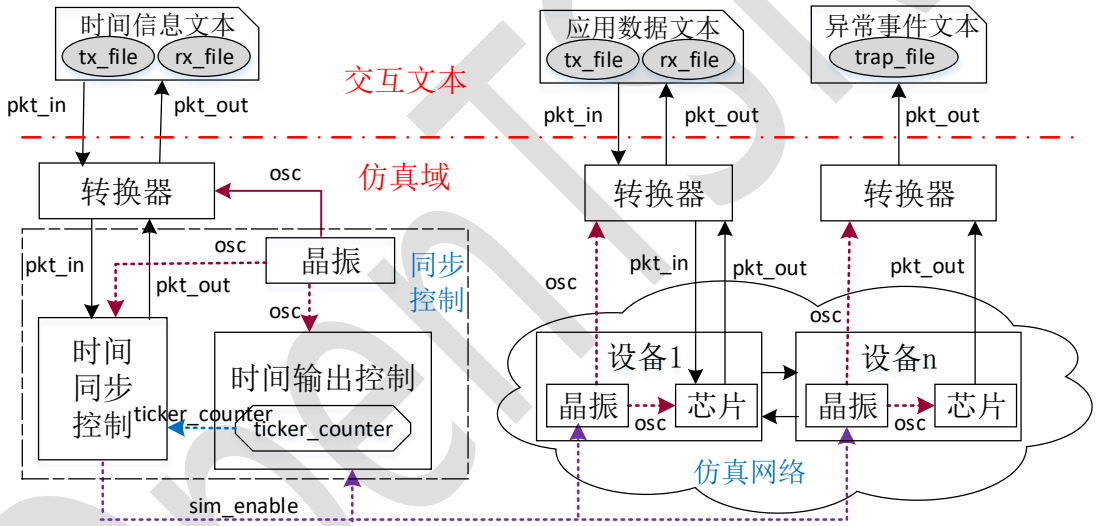

图 3 仿真域工作原理示意图

图 3 中的信号描述如表 1。

表5-1. 表 1 信号含义

接口信号	含义
osc	晶振时间
pkt_in	接收报文
pkt_out	发送报文

接口信号	含义
ticker_counter	时间计数器
sim_enable	时间信号使能

3.2.2 功能描述

如图 3 所示，同步控制逻辑主要包含四个模块，分别是转换器、晶振、时间同步控制和时间输出控制。各个模块的功能描述如下：

（1）转换器模块负责文本字符与报文格式的转换。一方面从时间信息文本（tx_file）中读取报文发送给时间同步控制模块，另一方面把时间同步控制模块要发送的报文写入时间信息文本(rx_file)。

（2）晶振模块负责给转换器模块、时间同步控制模块和时间输出模块提供时间晶振。

（3）时间同步控制模块负责跨域协同仿真工具中的软件域与仿真域的系统时间同步控制。一方面，接收时间同步控制模块输出的时间计数器 ticker_counter 值作为仿真域系统时间（t_{mod_time}）信息，并封装成报文 sync_pkt(t_{mod_time})发送给转换器模块；另外一方面，接收转换器模块发来的 sync_pkt(t_{cpu_time}),提取软件域的系统时间信息(t_{cpu_time})。然后将软件域的系统时间（t_{cpu_time}）与仿真域系统时间（t_{mod_time}）进行比较。若$|\ t_{mod_time} - t_{cpu_time}\ |$差值在Δt 之内，不做处理；若大于Δt，则时间同步控制模块向仿真网络和时间输出控制模块发送时间停止信号，直到差值小于Δt。

（4）时间输出控制模块维护寄存器 ticker_counter，并周期性的向时间同步控制模块发送 ticker_counter 值。寄存器 ticker_counter 受 sim_enable 信号的影响，若 sim_enable 信号为 unenable 值，则寄存器 ticker_counte 停止计数。

仿真网络由若干个网络设备节点组成，每个设备节点有自己的晶振。若仿真网络接收到时间同步控制模块发送的时间停止信号，则整个仿真网络的所有设备节点都停止工作，直到收到时间恢复信号。

3.3软件域的设计

3.3.1 软件域内的通信

如图 4 所示，软件域包括时间管理进程和一些其他的软件应用程序。软件域内的所有软件程序，需要通过交互文本与仿真域进行通信。

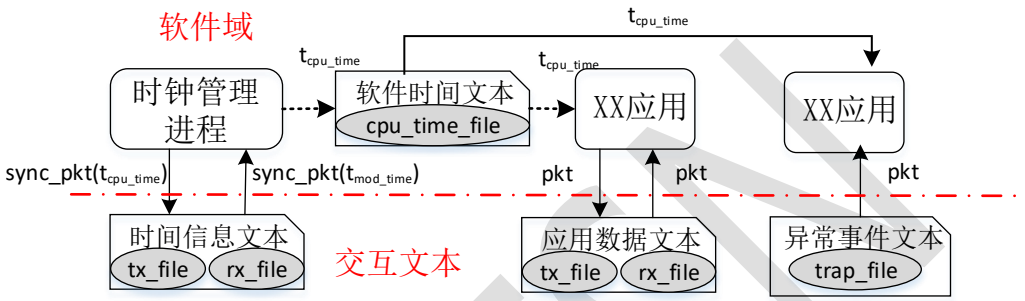

图 4 软件域工作原理示意图

关于时间管理进程与软件域内其他的应用程序之间的通信，为了简化软件域的实现复杂性，在初始实现阶段通过软件时间文本进行交互。也就是说，软件域内其他的应用程序需要从软件时间文本中获取软件域的系统时间，使用 OpenTSN 项目组自定义的 gettimeofdaytxt()函数代替 Linux 系统提供的 gettimeofday()函数。因此，在初始实现阶段，软件域内的应用程序仅能通过应用程序自身主动去查询软件域系统时间以实现定时功能。

未来 OpenTSN 团队在后续的优化工作中，将会扩展时间管理进程与软件域内其他的应用程序之间的通信方式，比如 socket 通信等。时间管理进程与软件域内其他的应用程序之间的通信可实时动态通信，因此软件域内的应用程序可向时间管理进程注册定时周期，时间管理进程也可向应用程序发送定时消息，模拟应用程序向操作系统注册定时并接收定时中断过程。

3.3.2 时间管理进程

时间管理进程的主要功能是实现跨域协同仿真工具中的软件域与仿真域的系统时间同步控制，为软件域内其他应用程序提供软件域的系统时间。时间管理进程的工作流程如图 5 所示。

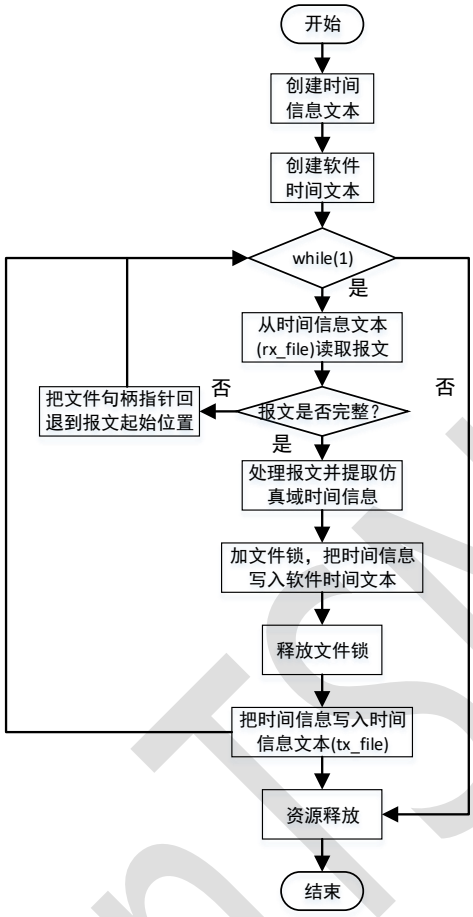

图 5 时间管理进程工作流程图

时间管理进程的工作流程描述如下：

（1）创建时间文本信息 tx_file 和 rx_file，用于与仿真域进行同步报文的交互；

（2）创建软件时间文本，用于为软件域内其他的软件应用程序提供软件域的系统时间；

（3）循环读取时间信息文本（rx_file）；

（4）判读读写的报文是否完整，若不完整则将文件指针回退到报文起始位置，转至步骤（3）；若报文完整，则转至步骤（5）；

（5）从同步报文中提取仿真域的系统时间；

（6）加软件时间文本锁，然后将仿真域的系统时间作为软件域系统时间，写入软件时间文本；

（7）释放软件时间文本锁；

（8）把软件域系统时间封装成报文，写入时间信息文本（tx_file）；

（9）若程序异常，则释放资源并结束；否则转至步骤（3）。

3.3.3 基础 API 库

软件域内的所有软件程序，需要通过交互文本与仿真域进行通信。软件域内其他的应用程序需要从软件时间文本中获取软件域的系统时间。为此 OpenTSN 项目组提供基础 API 库，包括系统时间获取 API、接收报文 API 和发送报文 AP。

1）系统时间获取 API

函数定义	struct timeval gettimeofdaytxt()
功能描述	从软件时间文本中获取软件域的系统时间
输入参数	无
输出参数	若成功返回软件域系统时间，若失败返回全 0

2）接收报文 API

接收报文 API 主要包括数据报文接收初始化 API、数据报文接收处理 API 和数据报文接收销毁 API。

（1）数据报文接收初始化 API

函数定义	int data_pkt_receive_init(u8* rule)
功能描述	完成数据报文接收资源的初始化。包括初始化 buff 空间、打开文本、设置过滤规则等
输入参数	过滤规则字符串指针，示例：u8* rule= "ether[3:1]=0x01 and ether[12:2]=0xff01";
返回结果	成功返回 0，失败返回-1

（2）数据报文接收处理 API (每次抓一个包）

函数定义	u8 *data_pkt_receive_dispatch_1(u16 *len_len);

功能描述	每次从文本中读取一个报文，然后对报文字符进行格式处理，再输出报文首地址指针和报文长度
输入参数	报文长度指针
返回结果	成功返回报文首地址指针，失败返回 NULL

（3）数据报文接收销毁 API

函数定义	int data_pkt_receive_destroy ( )
功能描述	完成数据报文接收资源的销毁，包括关闭文本句柄，释放 buff 空间等
输入参数	无
返回结果	成功返回 0，失败返回-1

3）发送报文 API

发送报文 API 主要包括数据报文发送初始化 API、数据报文发送处理 API 和数据报文发送销毁 API。

（1）数据报文发送初始化 API

函数定义	int data_pkt_send_init()
功能描述	完成数据报文发送资源的初始化。包括初始化 buff 空间、打开文本句柄等
输入参数	无
返回结果	成功返回 0，失败返回-1

（2）数据报文发送处理 API

函数定义	int data_pkt_send_handle(u8* pkt,u16 len)
功能描述	先对字符串进行文本格式的转换，然后再将数据写入文本，完成数据报文的发送
输入参数	数据报文指针、数据报文长度
返回结果	成功返回 0，失败返回-1
功能描述	完成数据报文的发送处理

（3）数据报文发送销毁 API

函数定义	int data_pkt_send_destroy()
功能描述	完成数据报文发送相关资源的销毁，包括关闭文本句柄、释放 buff 空间等
输入参数	无
返回结果	成功返回 0，失败返回-1

4 应用示例

4.1 组网拓扑

应用示例的组网拓扑如图 7 所示，由 3 台主机、2 个网卡和 1 个交换机组成，p0、p1 和 p2 分别为交换机的第 0、1 和 2 号端口。主机 1 和主机 3 部署端系统应用程序，主机 1 上运行 BE 流量发包器应用程序，主机 3 上运行 BE 流量接收器应用程序。主机 2 部署网络管理程序，运行集中式控制器程序 TSNLight。

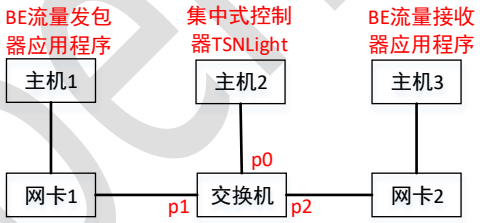

图 7 应用示例组网拓扑

4.2 仿真环境实现

应用示例在跨域协同仿真工具中的实现如图 8 所示。图 8 中的左边部分(标蓝)用于实现软件域和仿真域系统时间同步,与应用无关。图 8 中的右边部分与应用相关,软件域中的 BE 流量发包器、TSNLight 和 BE 流量接收器，对应着应用示例中的主机 1、主机 2 和主机 3 上的软件应用程序；仿真网络对应着应用示例中的网卡 1、交换机和网卡 2 组成的交换网络。

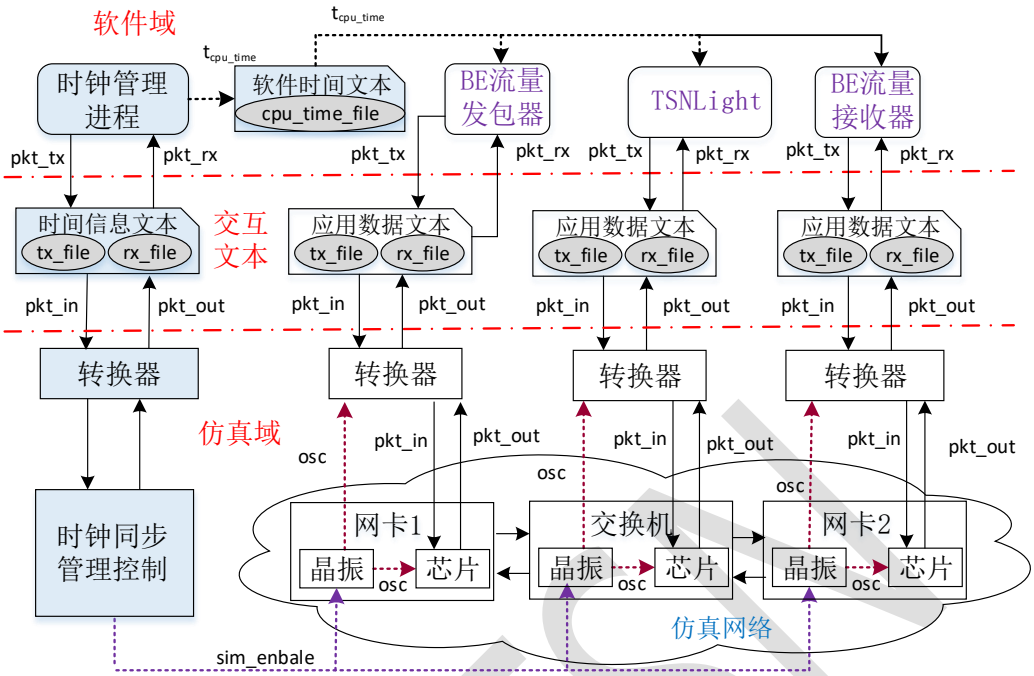

图 8 应用示例在仿真环境中的实现

图 8 中的左边部分（标蓝）、转换器以及应用数据文本读写，由 OpenTSN 项目组实现。软件应用程序、仿真网络，与应用相关，由用户实现。

对于软件应用程序，如 BE 流量发包器、TSNLight 或 BE 流量接收器，关于收发报文的接口和获取软件域系统时间的接口，需要调用 OpenTSN 项目组所提供的基础 API 库函数，其他逻辑由用户按需实现。