OpenTSN网络控制器使用手册

（版本1.0）

OpenTSN开源项目组

2020年12月

目录

[OpenTSN网络控制器使用手册 1](#_Toc63676806)

[**1 引言** 3](#_Toc63676807)

[**2运行环境** 3](#_Toc63676808)

[**3 文件说明** 3](#_Toc63676809)

[3.1 ARP应用 3](#_Toc63676810)

[3.2 通用函数库 4](#_Toc63676811)

[3.3 网络初始化应用 5](#_Toc63676812)

[3.3 状态监测应用 5](#_Toc63676813)

[3.4 PTP时间同步应用 6](#_Toc63676814)

[**4 编译和运行步骤** 6](#_Toc63676815)

[4.1 编译 6](#_Toc63676816)

[4.2 运行 7](#_Toc63676817)

[**5 组网示例** 8](#_Toc63676818)

[5.1组网拓扑 8](#_Toc63676819)

[5.2配置文本 8](#_Toc63676820)

[5.3运行顺序 13](#_Toc63676821)

[**附录一：Libxml2库安装教程** 14](#_Toc63676822)

[**附录二：问题记录** 14](#_Toc63676823)

**1 引言**

本文档为TSN集中控制器使用手册，主要描述控制器运行环境、文件说明、编译和运行步骤、组网示例，用户可以参考该文档使用网络控制器。

网络控制器主要功能包含ARP代理、网络初始配置、状态监测和PTP时间同步，所有功能需要与硬件配合才能实现。

**2运行环境**

网络控制器的运行环境为Linux系统，需要安装以下库支撑程序运行

* 需要安装libpcap和libnet库，用于收包和发包
* 需要安装libxml2库，用于进行xml文件解析（xml库中需要支持连续解析多个xml文本，会在提供的压缩包中包含xml库文件，用户可以进行安装和使用）
* Linux设备需要把网口开启混杂模式

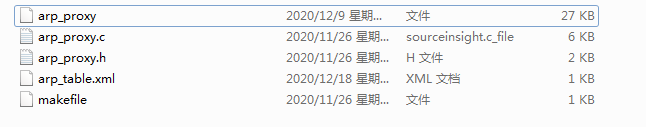
**3 文件说明**

网络控制器包含的源文件如下图所示，arp文件夹下存放的是arp应用程序，cnc\_api文件下存放通用基础库，cnc\_ptp文件下存放PTP时间同步应用程序，net\_init存放网络初始化应用程序，state\_monitor文件夹存放状态监测应用程序。



## 3.1 ARP应用

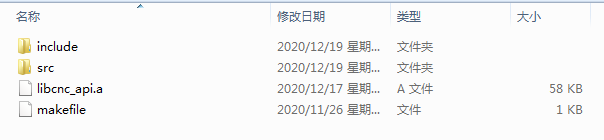
arp文件夹存放ARP代理应用程序，用于响应ARP请求报文，包含的文件如下图所示



* arp\_proxy：进行编译后生成的可执行文件
* arp\_proxy.c：arp代理的源文件，所有的函数在该文件中实现
* arp\_proxy.h：arp代理的头文件，arp数据结构定义在该文件中
* arp\_table.xml：用于存放arp表项
* makefile：可以进行编译的文件

## 3.2 通用函数库

cnc\_api文件夹为基础函数库，包含include文件夹（主要存放通用函数的api头文件）和src文件夹，包含的文件如下图所示



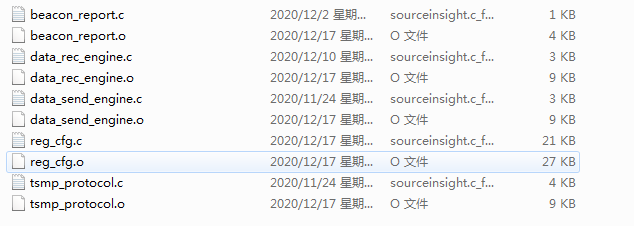
* include：主要存放通用函数的API头文件和通用的数据结构定义
* src：通用API的实现
* libcnc\_api.a：编译基础函数生成的动态库文件
* makefile：可以进行编译的文件

include文件夹中包含的文件



* beacon\_report.h：芯片上报报文的数据结构定义
* cnc\_api.h：通用API的头文件
* reg\_cfg.h：芯片配置和HCP配置报文的数据结构定义
* tsmp\_protocol.h：tsmp协议的数据结构定义

src文件夹中包含的文件



* beacon\_report.c：解析芯片上报报文和HCP上报报文的通用函数实现
* beacon\_report.o：编译beacon\_report.c生成的目标文件
* data\_rec\_engine.c：数据接收通用函数实现
* data\_rec\_engine.o：编译数据接收文件生成的目标文件
* data\_send\_engine.c数据发送通用函数实现
* data\_send\_engine.o：编译数据发送文件生成的目标文件
* reg\_cfg.c：寄存器配置通用函数实现
* reg\_cfg.o：编译寄存器配置文件生成的目标文件
* tsmp\_protocol.c：tsmp协议通用函数实现
* tsmp\_protocol.o: 编译tsmp协议文件生成的目标文件

## 3.3 网络初始化应用

net\_init文件夹存放网络初始化应用程序



* init：编译网络初始化进行生成的可以执行文件
* init\_cfg\_xml.xml：初始配置的xml文本，根据该文本进行网络探测
* makefile：make该文件生成可以执行文件init
* net\_init.c：主函数实现以及解析和进行初始配置的函数实现
* net\_init.h：网络初始配置的头文件
* offline\_plan\_xml.xml：离线规划的xml文本，用于进行离线规划配置
* parse\_offline\_plan\_file.c: 解析离线规划文本各进行离线规划配置的函数实现
* parse\_offline\_plan\_file.h：离线规划配置的头文件。

## 3.3 状态监测应用

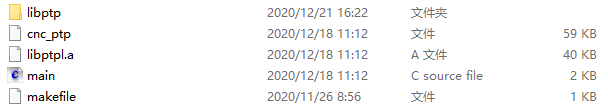
state\_monitor文件夹存放状态监测应用程序



* makefile：make该文件生成可执行文件monitor
* monitor：生成的可执行文件
* state\_monitor.c：状态监测模块的主函数和关键函数的实现
* state\_monitor.h：状态监测的头文件
* topolopy\_info\_xml.xml：拓扑信息的xml文本

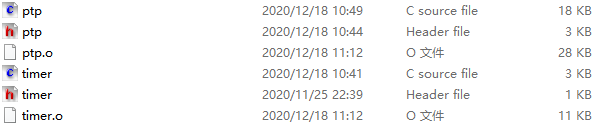
## 3.4 PTP时间同步应用

cnc\_ptp文件夹为PTP时间同步进程的实现



* libptp：主要存放时间同步相关函数的API头文件和数据结构定义以及时间同步相关函数API的实现
* cnc\_ptp：生成的可执行文件
* Libptpl.a：编译PTP时间同步函数生成的动态库文件
* main：时间同步主函数
* makefile：make该文件生成可执行文件cnc\_ptp

libptp文件夹中包含的文件



* Ptp.c：PTP同步处理模块的关键函数实现
* Ptp.h：PTP同步处理的头文件
* Ptp.o：编译时间同步处理文件生成的目标文件
* Timer.c：定时器模块的关键函数实现
* Timer.h：定时器处理的头文件
* Timer.o：编译定时器处理文件生成的目标文件

**4 编译和运行步骤**

## 4.1 编译

* 首先编译cnc\_api文件夹下的库文件，在cnc\_api文件夹下执行

make clean

make

* 编译net\_init文件夹下的网络初始化进程

make clean

make

* 编译state\_monitor文件夹下的状态监测进程

make clean

make

* 编译arp文件夹下的arp代理进程

make clean

make

* 编译cnc\_ptp文件夹下的ptp时间同步代理进程

make clean

make

## 4.2 运行

注：在root权限下执行

* 网络初始化进程（网络接口名为linux网络接口名字，使用ifconfig可以查看）

在net\_init目录下执行：./init 网络接口名

本设备的网络接口名为enp0s17，因此在执行时输入为 ./init enp0s17



* 运行状态监测进程

在state\_monitor目录下执行：./monitor 网络接口名

* 运行arp代理进程

在arp目录下执行：./arp\_proxy 网络接口名

* 运行PTP时间同步进程

在cnc\_ptp目录下执行：./cnc\_ptp

**附录一：Libxml2库安装教程**

* 拷贝库文件夹到Linux机器中
* 进入libxml2-2.6.2文件夹目录下，在终端中打开
* 使用root权限执行以下命令

make distclean

./configure

make

make install

* 安装完成。

**附录二：问题记录**

* 执行./init enp0s17时，出现没有可执行权限，需要执行chmod 777 init，赋予init文件可执行权限