TSN Centralized Controller User Manual

(Single process version 1.0)

OpenTSN Open Source Project Team

May 2021

Version history

Version	revision time	revision content	Revised file	ID
1.0	**2021.5.18**	According to the actual test situation Write the first draft. Populate network initialization, configuration, Description of **cnc_api**	Sun Yinhan Li Junshuai	**TSNLight3.0** single process Centralized Controller manual

content

1 Introduction

This document is the user manual of the OpenTSN 3.0 version of the centralized controller, which mainly describes the Controller operating environment, file descriptions, compilation and running steps, networking examples, users can refer to Refer to this document to use the network controller.

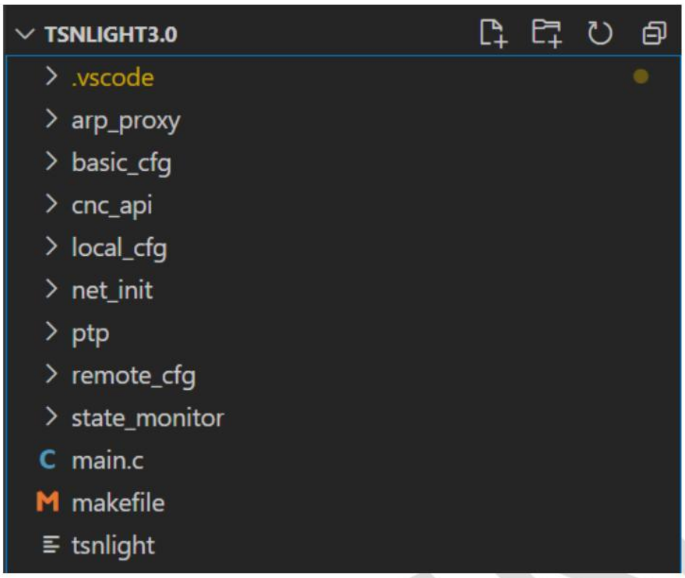
The most notable change in this version controller to the previous version is the way the implementation changed from multiprocessing to Single process, which can improve control in resource-constrained scenarios such as embedded devices with single-core processors performance of the device. At the same time, the controller also adds communication with the NETCONF agent to obtain control information and configure functions such as networking.

The main functions of the current version of the controller include network initialization, local configuration, remote configuration configuration, ARP proxy, basic network configuration, status monitoring and PTP time synchronization, all functions It needs to work with hardware to function properly.

2. Detailed description

1. Document Description

Classified according to different functions, the main directory of the project folder contains a total of 8 subfolders and 3 file. As shown below.



TSNLIGHT3.0
- .vscode
- arp_proxy
- basic_cfg
- cnc_api
- local_cfg
- net_init
- ptp
- remote_cfg
- state_monitor
- main.c
- makefile
- tsnlight

Figure 1-1 Project file directory

(1) arp_proxy

This folder contains code files related to the ARP proxy.

Table 2-1 ARP proxy code file description

file name	content
arp.h	Declare the main data structures and programming interfaces of the ARP proxy function
arp.c	Concrete implementation of programming interface, global variable declaration
When the arp_table.xml code runs, it will read this file, obtain the ARP table and store it in the corresponding data structure	

(2) basic_cfg

This folder contains code files related to the base configuration.

Table 2-2 Basic configuration code file description

file name	content
basic_cfg.h	The header file of the basic configuration, declaring the main data structure and programming interface of the basic configuration
basic_cfg.c	The source file of the basic configuration, which mainly implements the functions of the basic configuration

(3) cnc_api

This folder contains code files related to common APIs, and this folder contains include folder and src folder.

The include file mainly contains the header files of the general function library

Table 2-3 General API header file description

file name	content
beacon_report.h	Declare the data structure of the reported information, including the format of the reported register, the format of the reported statistical information, and the data format of the status information
cnc_api.h	Contains common header files, definitions of individual states, and declarations of common APIs
reg_cfg.h	Define the address of each configuration register and table entry, the data format of the configuration content, the data format of the configuration message mainly declares about the tsmp protocol, including the redefinition
tsmp_protocol.h	of signed and unsigned, the definition of TSMP subtype, the definition of TSNTag and the TSMP header Definition

The src file mainly contains the source files of the general function library

Table 2-4 General API source file description

file name	content
beacon_report.c	The source program of the Beacon report message, currently only the function to get the report type
data_rec_engine.c	The source file of data reception mainly includes the data reception initialization function and the data reception function. The source file of data transmission mainly includes the data transmission initialization function and the data transmission
data_send_engine.c	processing function. The source file of network configuration includes the register configuration function, table item configuration function, hcp Configuration function, and endian conversion function
reg_cfg.c	
tsmp_protocol.c	tsmp protocol source file, including tsmp message construction and parsing functions

(4) local_cfg

This folder contains code files related to local configuration.

Table 2-5 Description of local configuration files

file name	content
local_cfg.h	Header files for local configuration, data structures that declare local configuration, and function declarations
local_cfg.c	The source file of local configuration to realize the function of local configuration
local_cfg_xml.xml	When the code runs, it will read the file, obtain the local configuration information of offline planning, and then Parse the content of the text and configure it into the hardware

(5) net_init

This folder contains code files related to network initialization.

Table 2-6 Description of network initialization files

file name	content
net_init.h	Declare the main data structures and programming interfaces for network initialization
net_init.c	Concrete implementation of network initialization programming interface, global variable declaration
init_cfg_xml.xml	When the code runs, the file will be read to obtain the content of the basic network configuration. The content is specific Contains node register parameters, forwarding table. (The purpose of this section is to enable configuration packets to Reaching each node, the report message can reach the controller).

(6) ppt

This folder contains code files related to ptp time synchronization.

Table 2-7 PTP time synchronization file description

file name	content
ptp_single_process.h	declares the main data structure and programming interface of the time synchronization function
	Specific implementation of ptp_single_process.h programming interface, global variable declaration

(7) remote_cfg

This folder contains code files related to remote configuration.

Table 2-8 Description of remote configuration files

file name	content
remote_cfg.h	declares the main data structures and programming interfaces for remote configuration
remote_cfg.c	Concrete implementation of remote configuration programming interface, global variable declaration

(8) state_monitor

This folder contains code files related to condition monitoring.

Table 2-9 Description of status monitoring files

file name	content
state_monitor.h	Declare the main data structures and programming interfaces of the state monitoring function
state_monitor.c	Concrete implementation of programming interface, global variable declaration
topology_info_xml.xml	When the code runs, the file will be read, and the network topology information will be obtained and stored in the corresponding data structure.

(9) main.c

This file contains the main function of the controller, which realizes the operation of the entire controller.

State jumps and processes.

(10) makefiles

This file is the compilation file makefile of the entire project, execute make in the main folder

Instructions can be compiled to generate the executable tsnlight. When it is necessary to clear the compiled intermediate files and the final file, execute the make clean command in the main folder.

2. Program running

First, execute the command make in the main file directory to generate an executable file

tsnlight;

```
make
```

Then execute sudo ./tsnlight [interface], note that this step requires an administrator

Permission, the parameter is the name of the network card used to receive and send packets.

```
sudo ./tsnlight enp1s0
```

Appendix **A: Libxml2** Library Installation Tutorial

ÿ Copy the library folder to the Linux machine ÿ Go

to the libxml2-2.6.2 folder and open it in the terminal ÿ Execute ./configure ÿ make

ÿ make install

Appendix B: Problem Log

ÿ When executing ./tsnlight enp0s17, there is no executable permission, you need to execute chmod 777 tsnlight to give the tsnlight file executable permission