YANG Instance File for 60802 IA-Station

Martin Mittelberger (Siemens AG)
01/2023
v02

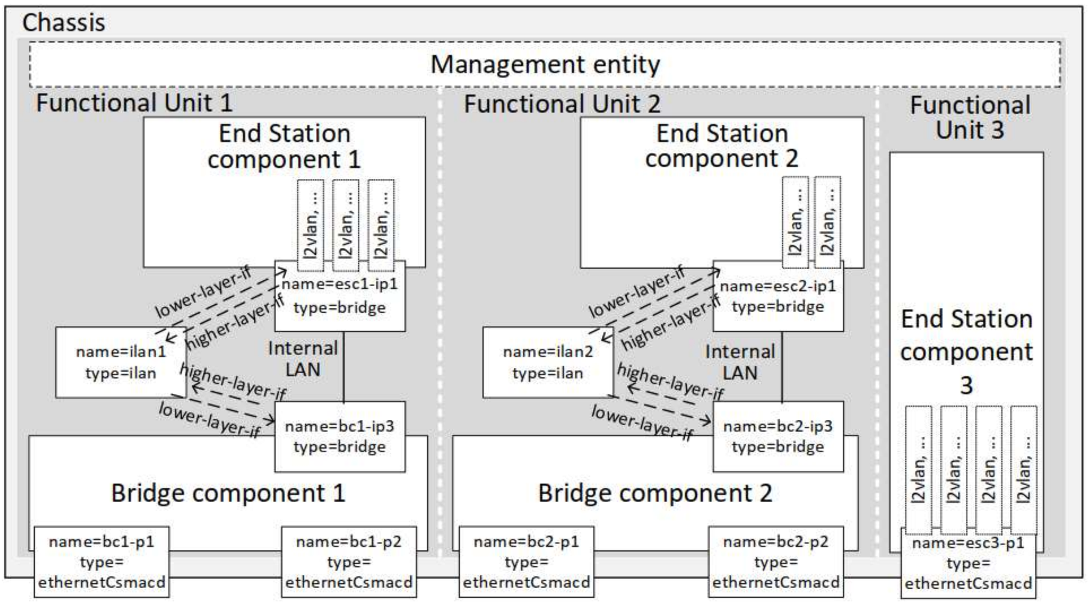
This contribution shows one possible way how to produce an instance file for a 60802 IA station. A LINUX system with installed pyang, yanglint and expect tools is required.

Example-files are appended to this pdf document.

Structure of the example device

The structure of this example is according to figure 21 in 60802 d1.4.

The device contains only functional unit 3, as it is the same device as described in <60802-Mittelberger-Time-Sync-Configuration-0123-v01.pdf>



Security relevant modules, gate-parameter-table and ietf-hardware are not yet included in this example.

Generate skeleton xml files from YANG modules

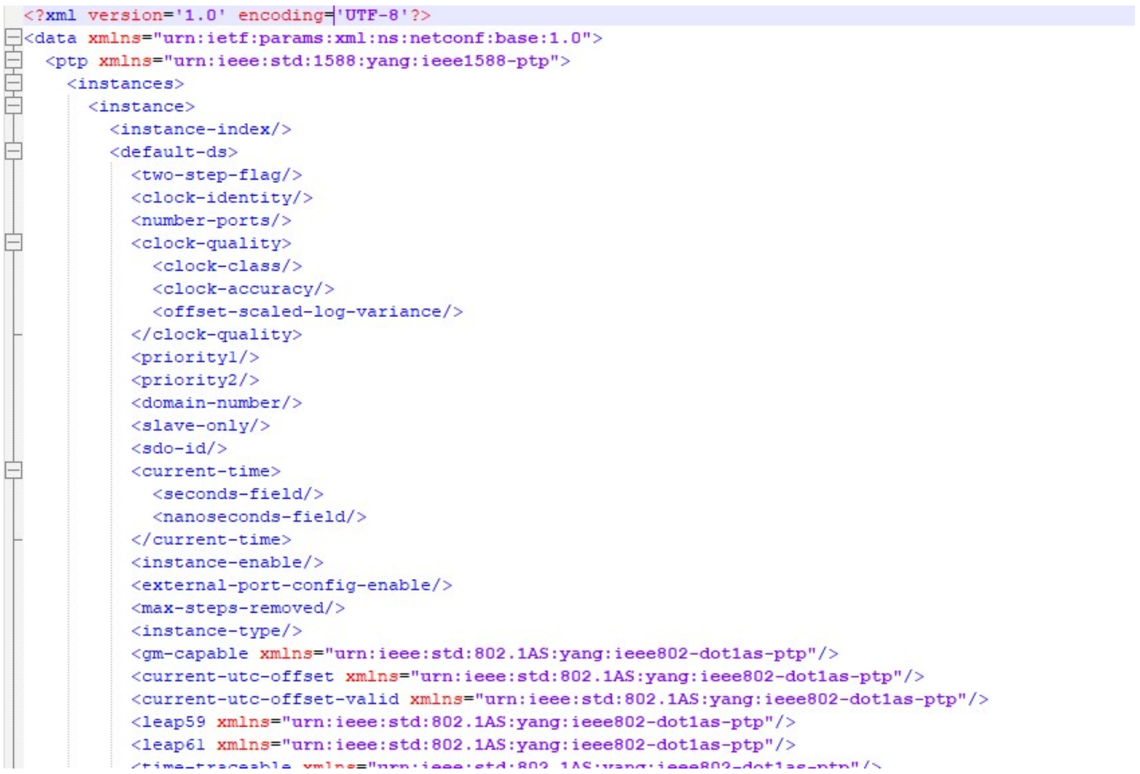
The pyang tool can be used to generate skeleton XML files out of YANG modules. In this example the YANG files **“ieee1588-ptp.yang”** and **“ieee802-dot1as-ptp.yang”** are used to generate a skeleton file for the time sync configuration.

As a precondition all both yang files and all yang files which are included by them must exist in the working directory.

The following command (see **“gen-timesync.sh”**) generates a file **“skeleton-timesync.xml”** (which already contains default values if the yang modules contains defaults):

```
pyang --format=sample-xml-skeleton --sample-xml-skeleton-doctype=data --sample-xml-skeleton-
defaults -o skeleton-timesync.xml ieee1588-ptp.yang ieee802-dot1as-ptp.yang
```

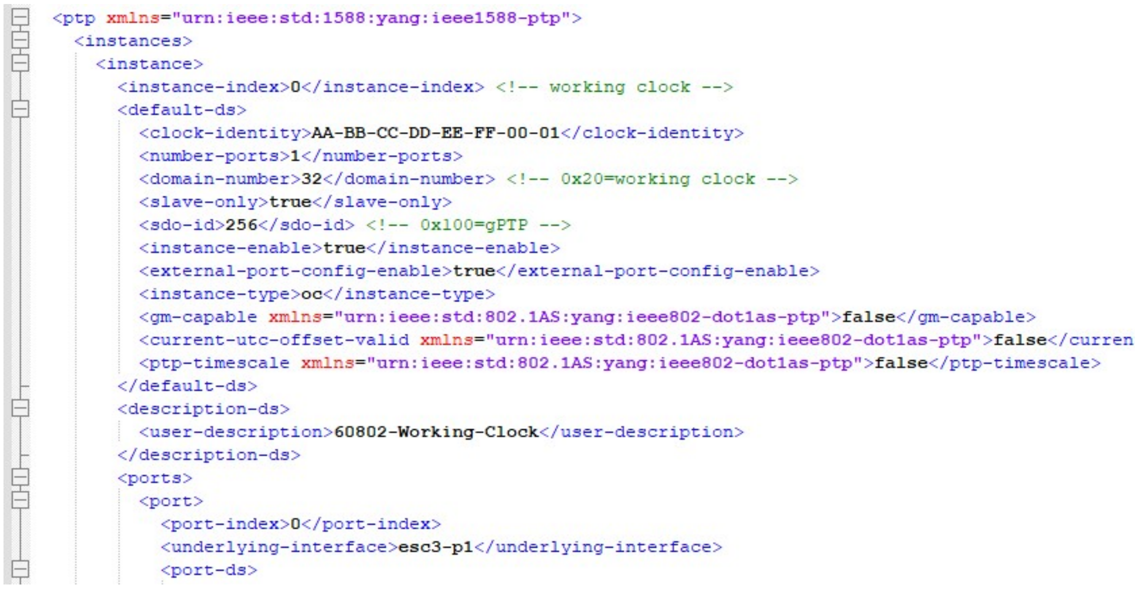
The output file is a skeleton XML file (see **“skeleton-timesync.xml”**) with both 1588 and augmented 802.1AS YANG elements:



```
<?xml version='1.0' encoding='UTF-8'?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ptp xmlns="urn:ieee:std:1588:yang:ieee1588-ptp">
    <instances>
      <instance>
        <instance-index/>
        <default-ds>
          <two-step-flag/>
          <clock-identity/>
          <number-ports/>
          <clock-quality>
            <clock-class/>
            <clock-accuracy/>
            <offset-scaled-log-variance/>
          </clock-quality>
          <priority1/>
          <priority2/>
          <domain-number/>
          <slave-only/>
          <sdo-id/>
          <current-time>
            <seconds-field/>
            <nanoseconds-field/>
          </current-time>
          <instance-enable/>
          <external-port-config-enable/>
          <max-steps-removed/>
          <instance-type/>
          <gm-capable xmlns="urn:ieee:std:802.1AS:yang:ieee802-dot1as-ptp"/>
          <current-utc-offset xmlns="urn:ieee:std:802.1AS:yang:ieee802-dot1as-ptp"/>
          <current-utc-offset-valid xmlns="urn:ieee:std:802.1AS:yang:ieee802-dot1as-ptp"/>
          <leap59 xmlns="urn:ieee:std:802.1AS:yang:ieee802-dot1as-ptp"/>
          <leap61 xmlns="urn:ieee:std:802.1AS:yang:ieee802-dot1as-ptp"/>
          <time-traceable xmlns="urn:ieee:std:802.1AS:yang:ieee802-dot1as-ptp"/>
```

This file must then be modified manually to represent the wanted device instance.

Following is a fragment of the XML file which represents a 60802 IA end station. In this file (**“60806-End-station.xml”**) exemplary values have been inserted and unnecessary leaves have been removed:



```xml
<ptp xmlns="urn:ieee:std:1588:yang:ieee1588-ptp">
  <instances>
    <instance>
      <instance-index>0</instance-index> <!-- working clock -->
      <default-ds>
        <clock-identity>AA-BB-CC-DD-EE-FF-00-01</clock-identity>
        <number-ports>1</number-ports>
        <domain-number>32</domain-number> <!-- 0x20=working clock -->
        <slave-only>true</slave-only>
        <sdo-id>256</sdo-id> <!-- 0x100=gPTP -->
        <instance-enable>true</instance-enable>
        <external-port-config-enable>true</external-port-config-enable>
        <instance-type>oc</instance-type>
        <gm-capable xmlns="urn:ieee:std:802.1AS:yang:ieee802-dot1as-ptp">false</gm-capable>
        <current-utc-offset-valid xmlns="urn:ieee:std:802.1AS:yang:ieee802-dot1as-ptp">false</curren
        <ptp-timescale xmlns="urn:ieee:std:802.1AS:yang:ieee802-dot1as-ptp">false</ptp-timescale>
      </default-ds>
      <description-ds>
        <user-description>60802-Working-Clock</user-description>
      </description-ds>
      <ports>
        <port>
          <port-index>0</port-index>
          <underlying-interface>esc3-p1</underlying-interface>
          <port-ds>
```

The manually populated XML file can then be checked with yanglint.

The following input file (**“60802-End-station.in”**) for the expect tool loads all necessary yang modules into yanglint and checks the syntax of the XML file. In this example a JSON file is also generated if the check is successfully:

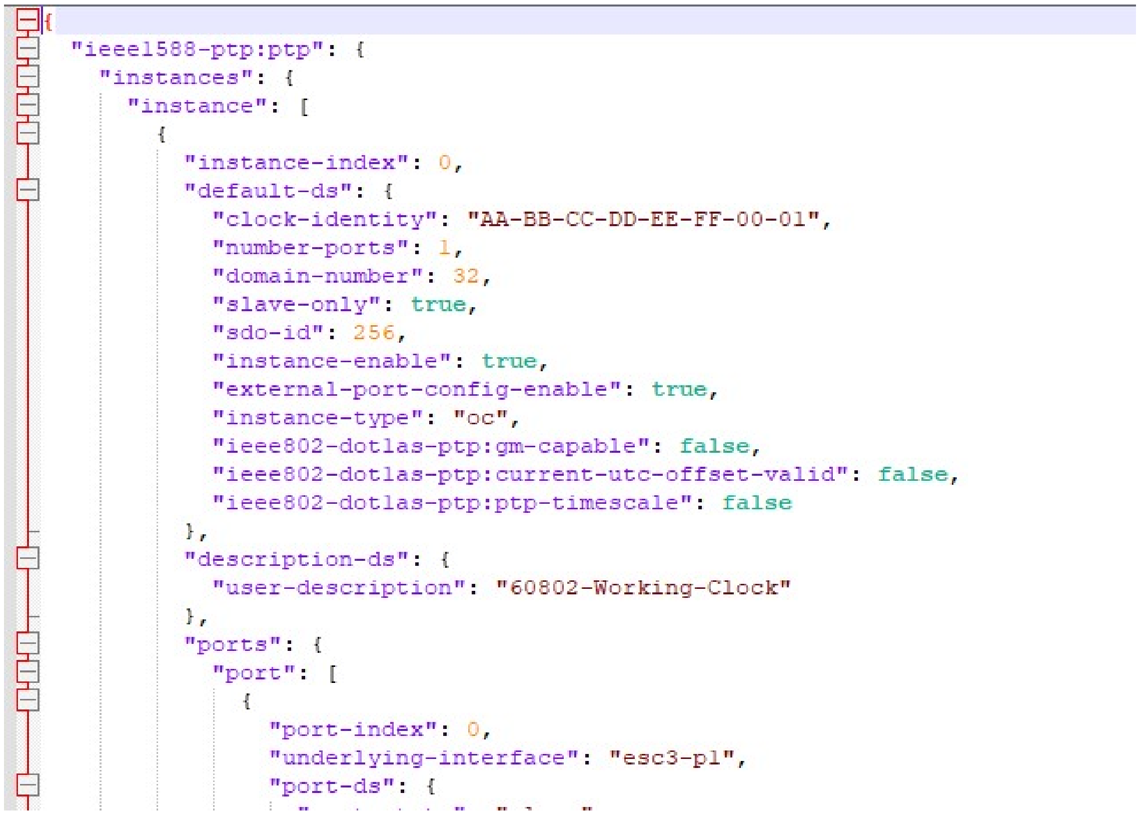
```
#! /usr/bin/expect -f
spawn /usr/local/bin/yanglint
expect ">"
send "load ietf-hardware@2018-03-13\r"
expect ">"
send "load iana-hardware\r"
expect ">"
send "load iana-if-type\r"
expect ">"
send "load ieee802-dot1q-types\r"
expect ">"
send "load ietf-system@2014-08-06\r"
expect ">"
send "load ietf-interfaces@2018-02-20\r"
expect ">"
send "load ietf-routing\r"
expect ">"
send "load ieee802-dot1q-bridge\r"
expect ">"
send "load ieee802-ethernet-interface\r"
expect ">"
send "load ieee1588-ptp --features=ieee1588-ptp:external-port-config,cmlds\r"
expect ">"
send "load ieee802-dot1as-ptp\r"
expect ">"
send "load ieee802-dot1ab-lldp\r"
expect ">"
send "load ieee802-dot1cb-stream-identification\r"
expect ">"
send "load ieee802-dot1cb-frer\r"
expect ">"
send "load ieee802-dot1q-sched\r"
expect ">"
send "load ieee802-dot1q-preemption\r"
expect ">"
send "load ietf-ip\r"
expect ">"
send "load ietf-tcp-common --features=ietf-tcp-common:keepalives-supported\r"
expect ">"
send "load ietf-tcp\r"
expect ">"
send "load ietf-dhcp --features=ietf-dhcp:client\r"
expect ">"
send "data -e -t data -f json --output=60802-End-station.json 60802-End-station.xml\r"
expect ">"
send "quit\r"
expect eof
```

Please note, that “features” of the YANG models can also be specified to represent the simulated IA-station (in this example: ieee1588-ptp:external-port-config and cmlds, and tcp-common:keepalives-supported).

The following commands starts the execution of yanglint:

```
expect 60802-End-station.in
```

A fragment of the generated JSON output (**“60802-End-station.json”**)file is shown here:



```
{
  "ieee1588-ptp:ptp": {
    "instances": {
      "instance": [
        {
          "instance-index": 0,
          "default-ds": {
            "clock-identity": "AA-BB-CC-DD-EE-FF-00-01",
            "number-ports": 1,
            "domain-number": 32,
            "slave-only": true,
            "sdo-id": 256,
            "instance-enable": true,
            "external-port-config-enable": true,
            "instance-type": "oc",
            "ieee802-dot1as-ptp:gm-capable": false,
            "ieee802-dot1as-ptp:current-utc-offset-valid": false,
            "ieee802-dot1as-ptp:ptp-timescale": false
          },
          "description-ds": {
            "user-description": "60802-Working-Clock"
          },
          "ports": {
            "port": [
              {
                "port-index": 0,
                "underlying-interface": "esc3-p1",
                "port-ds": {
```