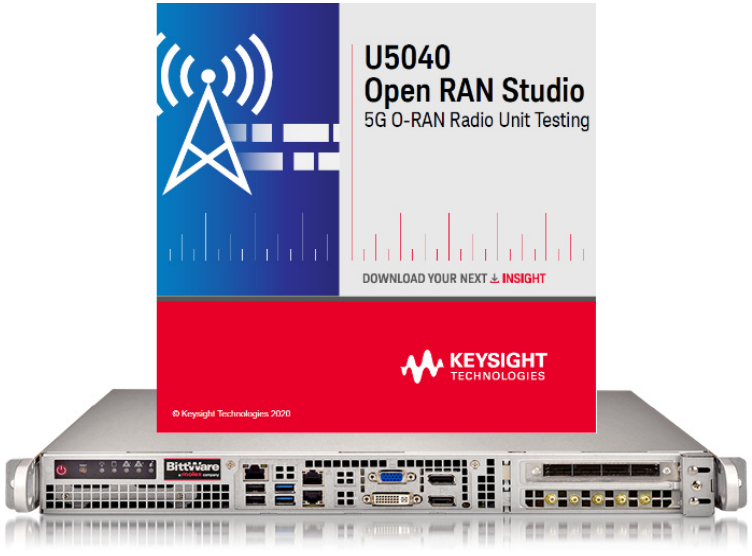
Implementing M-Plane & M-Plane Passthrough in ORAN Studio Solution (v1.1)

Reference Guide



KEYSIGHT
TECHNOLOGIES

Management (M) Plane – Overview

The Management Plane (M-Plane) is the part of a network system for configuration, monitoring, management and distribution of various services to all layers of the network stack and other parts of the system.

NOTE

Standardization with M-Plane

M-plane is the standardization of the configuration of O-RUs, without the need of any obscure and proprietary interfaces.

M-Plane characteristics

The M-Plane is used to:

- Handle operations in networks with multiple O-DUs and O-RUs.
- Define operations and functions to configure parameters required by the CUS-planes.

M-Plane architecture

The M-Plane architectures support two models:

- Hierarchical model

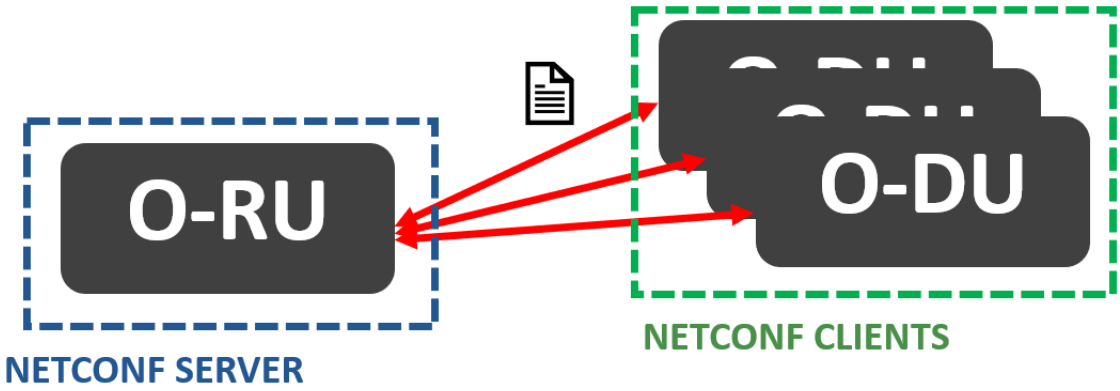


Figure 1 Block diagram for Hierarchical model of M-Plane architecture

- Hybrid model

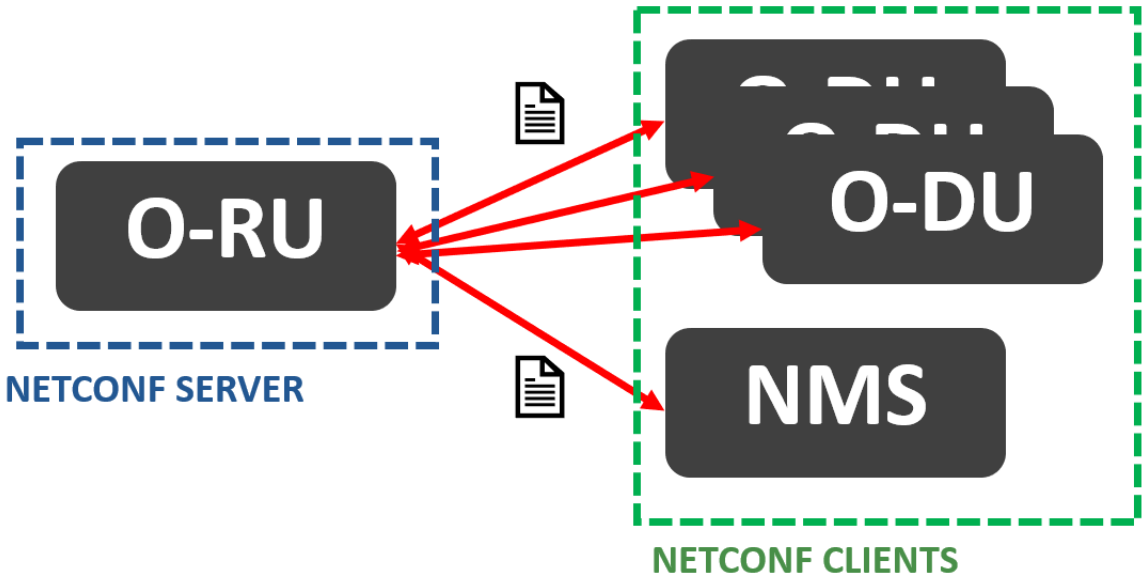


Figure 2 Block diagram for Hybrid model of M-Plane architecture

For efficient management of network operations, the key features in M-Plane are:

- NETCONF (Network Configuration): Network protocol for management
- YANG (Yet Another Next Generation): Data exchange language in NETCONF

Network Configuration (NETCONF) protocol

NETCONF is a network management protocol developed and standardized by the IETF. It was published in 2006 as RFC 4741 and later revised in 2011 and published as RFC 6241.

NETCONF provides mechanisms to:

- Install configuration of network devices
- Manage configuration of network devices
- Delete configuration of network devices

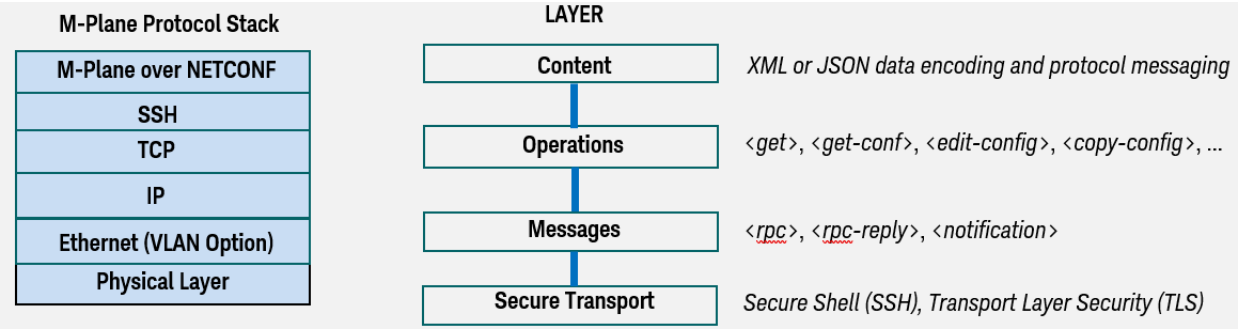


Figure 3 Block diagram for M-Plane protocol stack and NETCONF layer

Yet Another Next Gen (YANG) model

YANG was published as RFC 6020 in October 2010 by Internet Engineering Task Force (IETF).

YANG is a data model language by itself and not an encoding data language such as XML or JSON, although instances of the YANG model can be encoded into these formats for transport over the network. This model is used in networks management protocols, that is, NETCONF, to define the structure of configuration and state data (also, YANG modules) on network devices.

The YANG models are tree-based, such as:

module (name.yang)

container

group

leaf

...

Refer to *Annex D* in the *O-RAN.WG4.MP.0-v0X.00* specification for latest YANG models.

Figure 4 List of xRAN Fronthaul M-Plane YANG modules

M-Plane Functions

- Start Up installation
  - Establishes the M-plane connections between NETCONF clients and O-RU.
  - Setting of Transport Layer address via either manual, Dynamic Host Configuration Protocol (DHCP) and State-Less Address Auto-Configuration (SLAAC).
- SW Management
  - SW inventory
  - SW download
  - SW installation
  - SW activation
- Configuration Management
  - Sets the parameters required by O-RU for operation
  - Gets equipment status information
- Fault Management
  - Monitors potential errors and faults in the network
  - Contains notifications from O-RU such as ID, location of fault occurrence, locations affected by fault, severity, and so on

Contents

Implementing M-Plane in ORAN Studio Solution

Reference Guide

1. M-Plane in Keysight Open RAN Studio Solution

KEYSIGHT TECHNOLOGIES

Section 1.1: Keysight Open RAN Studio Solution - Overview

Figure 5 shows a comprehensive solution, comprising of BittWare server and U5040A Open RAN Studio software, for O-RU (Radio Unit) testing.

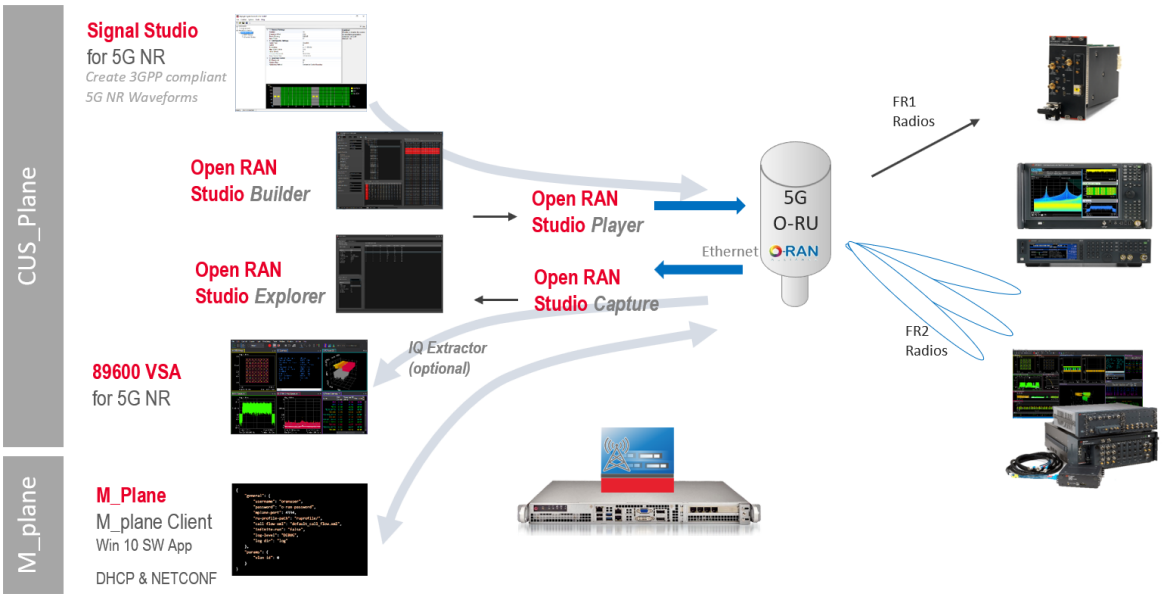


Figure 5 Key elements in the Open RAN Studio solution

1.1.1: Hardware platform for Open RAN Studio software

Keysight provides software applications running on a Windows based platform, including an FPGA controlled NIC card from BittWare (third-party company). The FPGA ensures time-deterministic generation and analysis. It also includes clock, trigger and PPS sharing to ensure that the entire system is operating under the same time and frequency reference.

System setup overview and features for the M_Plane client software	
Overview	▪ Host based emulation software ▪ M-Plane on independent Ethernet port ▪ Passthrough capabilities to merge M_plane with CUS_plane (FPGA module acts as a switch)
Features	▪ DHCP server with DU profile ▪ M_Plane NETCONF client ▪ Configurable Call Flows

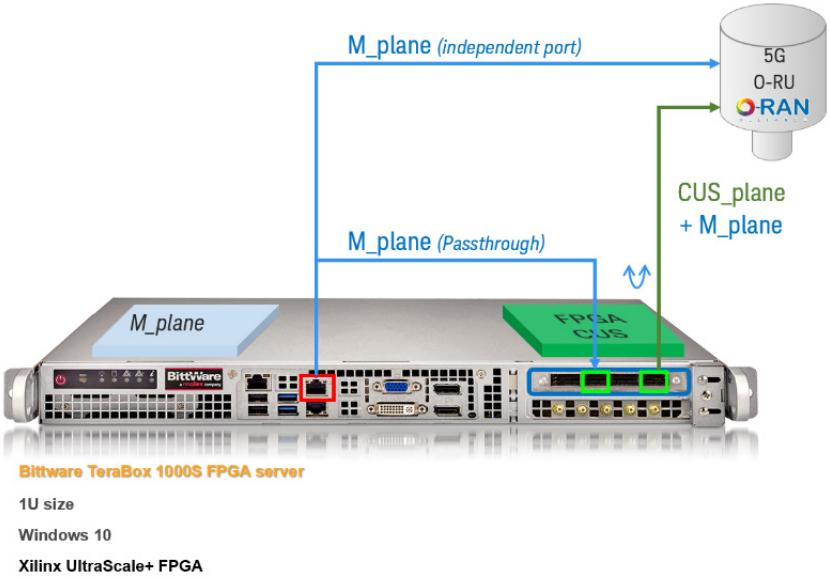


Figure 6 Setup for M_Plane client SW

1.1.2: M-Plane Support in Open RAN Studio solution

M-Plane features (supported and unsupported)	
Supported	▪ Based on O-RAN M-Plane Version 1.0 ▪ Most functions supported for Installation startup ▪ Hierarchical model ▪ IPv4 and DHCP
Not Supported	▪ Hybrid Model ▪ IPv6 ▪ VLAN support depends on Window NIC (currently not supported on BittWare HW) ▪ Antenna Line Devices (ALD) - may be proprietary ▪ Extended Input/Output ports ▪ LBM - Ethernet OAM loopback

M-Plane startup sequence support*

No.	Step	Support	Open RAN Studio components
1	Transport Layer Initialization	Yes	DHCP server
2	O-RU synchronization to primary reference clock	Yes	Open RAN Studio / BittWare server
3	O-RU Calls home to NETCONF clients	Yes	NETCONF client
4	SSH Secure Shell Connection Established	Yes	NETCONF client
5	NETCONF Capability Discovery	Yes	NETCONF client
6	Optional provisioning of new management accounts	TBD	
7	Supervision of NETCONF connection	Yes	NETCONF client
8	Retrieval of O-RU Information	Yes	NETCONF client
9	SW Management	No	-
10	CU-Plane transport connectivity check	No	-
11	U-Plane configuration	Yes	NETCONF client
12	Recovery of O-RU Delay Profile	No	-
13	Fault Management	Yes	NETCONF client

No.	Step	Support	Open RAN Studio components
14	Performance Management	No	-
15	Retrieval of O-RU State	Yes	NETCONF client
16	Configuring the O-RU operational parameters	Yes	NETCONF client

* Refer to Figure 1.4 in the ORAN-WG4-MP-0-V01.00 specification

1.1.3: M-Plane features available in ORAN solution

Release Version	M-Plane features
1.2.10801.0 (U5040A Open RAN Studio software download page)	▪ M-Plane Client Software ▪ M-Plane FPGA Pass-through (on BittWare HW)
Future versions	▪ M-Plane conformance testing ▪ M-Plane Client API (to support automated testing and conformance)

1.1.4: Installing M-Plane Client software

The M-Plane client software is installed, by default, with the U5040A Open RAN Studio software. You can download the software from the U5040A Open RAN Studio software download page. You may install the software either on the BittWare server or on your local PC. While the option to install “M-Plane Toolkit” is selected by default in both cases, the differences in the appearance of the “Custom Setup” window are shown in the images below. On the local machine, the M-Plane client is installed with the ‘offline’ mode of the U5040A Open RAN Studio software.

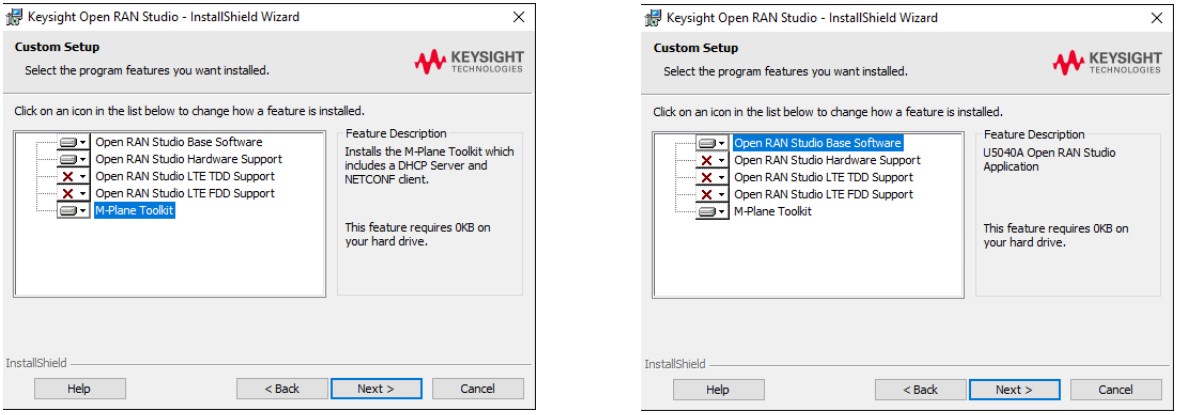


Figure 7 Custom setup on BittWare HW (left) and on a standard Win 10 PC (right)

1.1.5: Capabilities of the M-Plane client software

M-Plane capabilities	
DHCP server	**1** IP address assignment **2** Includes M-Plane specific parameters ▪ Sends Option 43: Vendor Specific Information: encodes IP address of NETCONF controller ▪ Receives / Parses Option 60: Vendor Class Identifier (such as, “oran-ru/<vendor>”)
NETCONF	**1** Scripted through Call-flow.xml ▪ Transport and Handshake (IPv4) ▪ NETCONF session establishment ▪ NETCONF Subscription to Notifications ▪ M-Plane connection supervision ▪ Retrieve active alarm list ▪ Retrieve O-RU information elements (with and without filter) ▪ Edit / Merge Configuration ▪ Create VLAN ▪ Add Processing Element ▪ Endpoint Carrier Configuration ▪ Endpoint Carrier Link Configuration ▪ Activate Carrier **2** Allows custom sequences, easy to modify

Section 1.2: Using M-Plane software

The following block diagrams illustrate three scenarios, where M-Plane messages only are transmitted, the M-Plane software, which is installed with U5040A Open RAN Studio software, is used to transmit both M-Plane and CUS-Plane messages separately or M-Plane Passthrough is applied along with either a second NIC on the BittWare server or an external PC.

Setup 1: Transmitting M-Plane messages only using Windows 10 PC

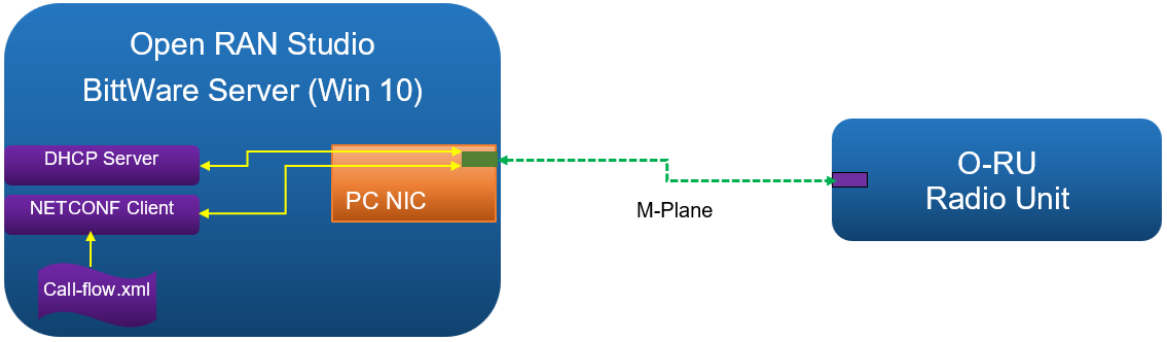


Figure 8 Transmitting M-Plane messages only using Win 10 PC

Setup 2: Transmitting M-Plane & CUS Plane messages using 2nd port on O-RU

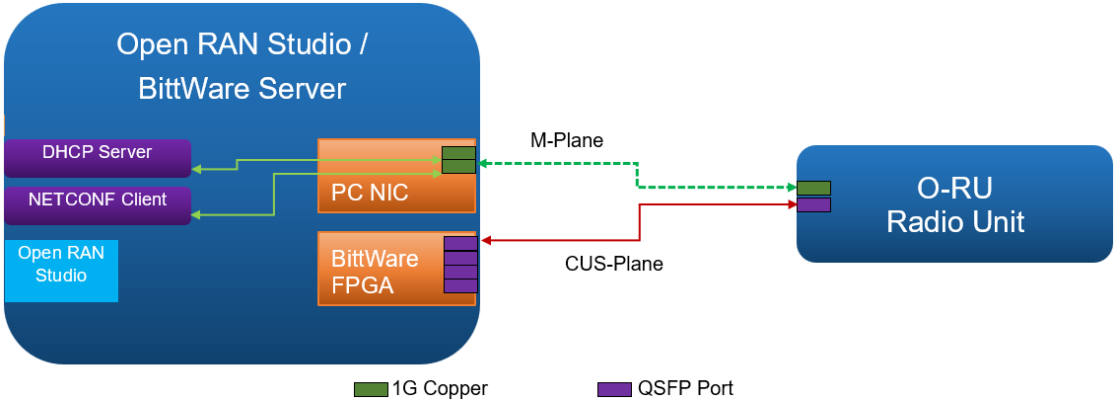


Figure 9 Transmitting M-Plane & CUS-Plane messages using M-Plane software

Setup 3: Transmitting CUS & M-Plane messages using M-Plane with FPGA Passthrough

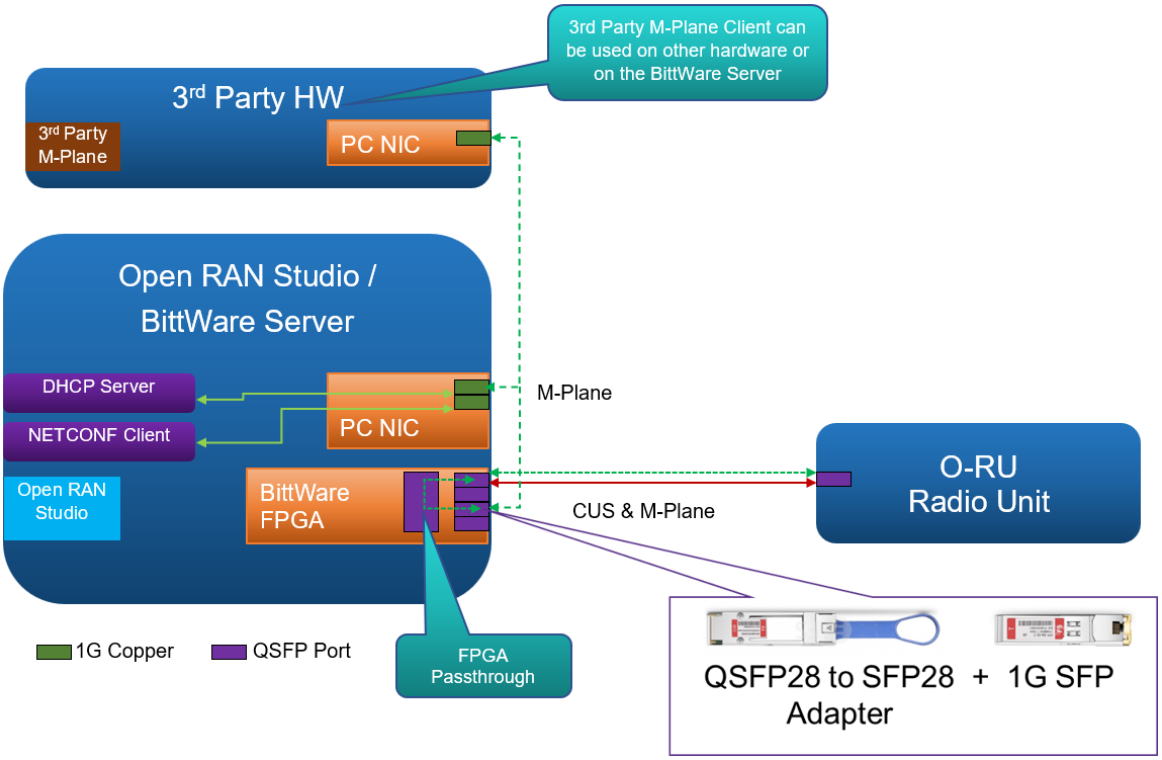


Figure 10 Transmitting CUS & M-Plane messages with BittWare 2nd NIC or ext. PC

Note that in the setup involving M-Plane with FPGA Passthrough, you must first insert a 1G SFP module into an QSFP28 to SFP28 adapter. Then, connect one end of a Cat-5 / Cat-6 Ethernet cable to the adapter and the other end to the secondary Ethernet NIC port on the BittWare server.

Figure 11 Setup requirements for M-Plane with FPGA Passthrough

Section 1.3: M-Plane configuration

To understand how to configure M-Plane, we must understand the configuration file structure.

1.3.1: Directory structure of the M-Plane client software

By default, the configuration files for the M-Plane client software can be found in: *C:\ProgramData\Keysight\MPlane\Client*.

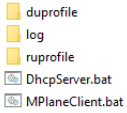


Figure 12 Directory structure of the M-Plane client

Folder / *File* name	Description
duprofile	Directory with DU configuration files
log	Directory with log files (this may not be created during the initial installation)
ruprofile	Directory with RU configuration files
DhcpServer.bat	Used to run a DHCP server, upon which, configuration can be provided to the O-RU
MPlaneClient.bat	Used to run the M-Plane NETCONF client

1.3.2: Flow of configuration files

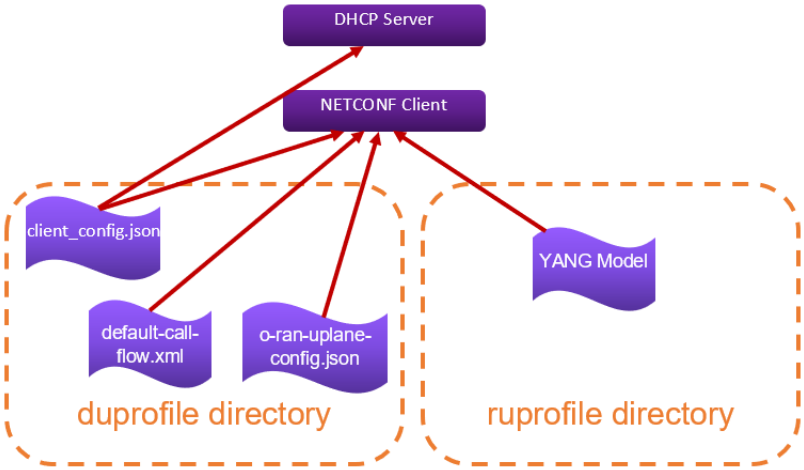


Figure 13 Flow of Configuration files in the M-Plane client

Config file name	Description
client_config.json	General client configuration
default-call-flow.xml	Defines the flow of actions that NETCONF client will perform, when the O-RU calls (it) home
o-ran-uplane-config.json	Defines the U-Plane configuration, which will be sent to the O-RU
YANG model	Described in Yet Another Next Gen (YANG) model.

Structure of the *client_config.json* file

```
{
  "general": {
    "username": "oranuser",
    "password": "o-ran-password",
    "infinite-run": "false",
    "log-level": "DEBUG",
    "log-dir": "log",
    "mplane-port": 4334,
    "call-flow-xml": "default_call_flow.xml"
  },
  "params": {
    "vlan-id": 0
  }
}
```

Parameters	Description
username	username for call-home session
password	password for call-home session
infinite-run	if true, the NETCONF client will keep the session open at the end of *default_call_flow.xml*
log-level	specifies logging level
log-dir	specifies directory for logging
mplane-port	TCP port for SSH / NETCONF session, default: 4334
call-flow-xml	name of file used for call flow script

Structure of the *o-ran-uplane-config.json* file

- This file defines the user plane carrier configuration, which is used in <edit-config/> operations.
- An example of this file populated can be found in *o-ran-uplane-conf.sample-filled.json*.

Snippet from the *default_call_flow.xml* file

```
<query>
    <get xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
        <filter
xmlns:yanglib="urn:ietf:params:xml:ns:yang:ietf-yang-libra
ry"
                type="xpath"
                select="/yanglib:*//."/>
    </get>
    <create-subscription
xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
        <stream>NETCONF</stream>
    </create-subscription>
    <supervision-watchdog-reset
xmlns="urn:o-ran:supervision:1.0"

action="start_supervision_thread">

<supervision-notification-interval>60</supervision-notific
ation-interval>
        <guard-timer-overhead>10</guard-timer-overhead>
    </supervision-watchdog-reset>
    <get xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
        <filter
xmlns:o-ran-module-cap="urn:o-ran:module-cap:1.0"
                type="xpath"
                select="/o-ran-module-cap:*//."/>
```

```
        <with-defaults
xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-with-defau
lts">report-all</with-defaults>
    </get>
…
</query>
```

This file defines a sequential list of Remote Procedure Calls (RPCs) that the NETCONF Client will perform when the O-RU calls home. This normally follows the of the O-RU startup sequence per the M-Plane specification. However, nothing prohibits using different sequences.

Note that a different filename can be specified in *client_config.json*.

Section 1.4: Understanding M-Plane commands

To launch the **M-Plane Client Command Prompt** window, click **Start** > **Keysight M-Plane Toolkit** > **M-Plane Client Command Prompt**.

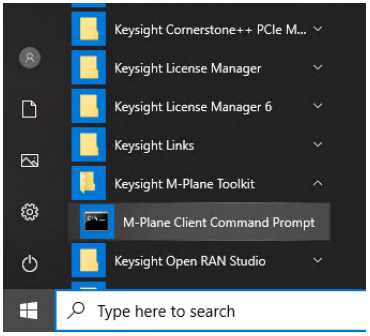


Figure 14 Launching M-Plane Client Command Prompt

The **M-Plane Client Command Prompt** window is launched, as shown in Figure 15.

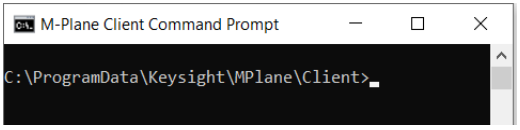


Figure 15 Default view of M-Plane Client Command Prompt window

The root folder *C:\ProgramData\Keysight\MPlane\Client* contains the configuration folders along with the *DhcpServer.bat* (to run the DHCP server) and *MPlaneClient.bat* (to run the NETCONF client) commands.

See M-Plane configuration on page 17 in this document for more details.

1.4.1: Description of the M-Plane commands

For DHCP server

- Filename: DHCPServer.bat
- Usage:

  DhcpServer.bat --ip <IP-ADDRESS-TO-BIND-ON> --duprofile <du_directory>
- Example:

  DhcpServer.bat --ip 192.168.2.1 --duprofile duprofile
- Parameters

1 --ip <IP-ADDRESS-TO-BIND-ON>
   - This is the IP Address of the windows Network Adapter that the DHCP Server will use. This should be a secondary Network Adapter configured with a static IP address.
   - WARNING! Running the DCHP server on the primary Adapter used to connect to a LAN may interfere with the DHCP server on the LAN and cause connectivity issues.
   - NOTE: There is currently a limitation that this IP address MUST end with **.1** through **.5**. This is because the DHCP server assumes those IP addresses are reserved for static IPs. DHCP clients will be given IP addresses **.6** or greater. Also, the subnet mask is assumed to be 255.255.255.0 (or /24 in CIDR notation).

2 --duprofile <du_directory>
   - <du_directory> specifies the directory with configuration files for the O-DU emulation (DHCP Server / NETCONF client).

For NETCONF client

- Filename: MPlaneClient.bat
- Usage:

  MPlaneClient.bat --ip <IP-ADDRESS-TO-BIND-ON> --duprofile <du_directory> --ruprofile <ru_directory>
- Example:

  MPlaneClient.bat --ip 192.168.2.1 --duprofile duprofile --ruprofile ruprofile
- Parameters

1 --ip <IP-ADDRESS-TO-BIND-ON>
   - This should be the same IP address used for the DHCP server (when using DHCP).
   - Specifies the IP Address that the NETCONF client will listen on (using the port number in <du_directory>/client_config.json, which is 4334, by default).

2 --duprofile <du_directory>
   - This specifies the directory with configuration files for the O-DU emulation (DHCP Server / NETCONF client). Normally, this should be set to “duprofile”.

3 --ruprofile <ru_directory>
   - -This specifies the directory with the O-RU configuration profile. Normally, this should be set to “ruprofile”. Note that this directory contains the YANG Models for the O-RU and they should not be modified.

Example for DHCP Server and NETCONF Client

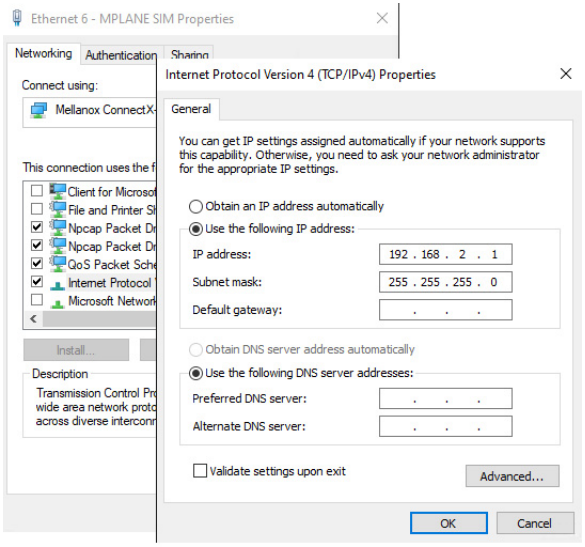


Figure 16 Configuring DHCP Server + NETCONF Client

1 Configure Secondary Network Connection
   - Configure Internet Protocol Version 4 with:
     - IP address: 192.168.2.1
     - Subnet mask: 255.255.255.0
     - Default gateway: <empty>
   - Disable IPv6
2 Launch two **M-Plane Client Command Prompt** windows.
   - On window 1: Run the DHCP Server
     - DhcpServer.bat --ip 192.168.2.1 --duprofile duprofile
   - On window 2: Run NETCONF Client

Section 1.5: Overview on RPC

The key elements in RPC are:

- get
- edit-config
- create-subscription
- supervision-watchdog-reset
- file-upload

1.5.1: RPC: get

Description

The “get” RPC retrieves running configuration information from the O-RU / NETCONF Server.

Sub-elements

- **Filter** (optional): specifies filtering criteria for the get RPC
  - Attribute: type = xpath | subtree
  - Attribute: select = subtree to select

```
EXAMPLE: get module capabilities

    <get xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">

        <filter
xmlns:o-ran-module-cap="urn:o-ran:module-cap:1.0"

                type="xpath"

                select="/o-ran-module-cap:*//."/>

        <with-defaults
xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-with-defau
lts">report-all</with-defaults>

    </get>
```

1.5.2: RPC: edit-config

Description

This RPC edits the existing configuration in the O-RU / NETCONF Server.

Attributes

- **xmlns**: defines the namespace
- **action** (optional): for some specific operations, an action is specified. See “Actions” for more details

Sub-elements

- **target**: specifies the datastore (that is, “running” or “startup”) to be modified.
- **default-operation**: specifies if the configuration should be a “merge” or a “replace”. In most cases, “merge” is used to merge into the existing configuration.
- **config**: specifies the configuration to merge.

Example: merge LBM configuration

```
    <edit-config
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
        <target>
            <running/>
        </target>
        <default-operation>merge</default-operation>
        <config>
           <md-data-definitions xmlns="urn:o-ran:lbm:1.0">
                <maintenance-domain>
                    <id>KEYS_xyz_LBM</id>
                    <name>level_7</name>
                    <md-level>7</md-level>
                </maintenance-domain>
            </md-data-definitions>
```

```
    </config>
</edit-config>
```

1.5.3: RPC: create-subscription

Description

This RPC creates a subscription to the specified stream.

Attributes

- **xmlns**: defines the namespace for the notification to subscribe to.

Sub-elements

- **stream**: specifies the stream to subscribe to. This can be “NETCONF”, or “supervision-notification”.

```
Example: Subscribe to NETCONF stream
    <create-subscription
xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
        <stream>NETCONF</stream>
    </create-subscription>
```

1.5.4: RPC: supervision-watchdog-reset

Description

This RPC creates a subscription to the specified stream.

Attributes

- **xmlns**: defines the namespace for the supervision watchdog reset.
- **action**: this should be set to “start_supervision_thread”

Sub-elements

- **stream**: specifies the stream to subscribe to.

See *ORAN-WG4.MP.0-v01.00 section 3.6* for more details.

```
Example:

    <supervision-watchdog-reset
xmlns="urn:o-ran:supervision:1.0"


action="start_supervision_thread">


<supervision-notification-interval>60</supervision-notific
ation-interval>

        <guard-timer-overhead>10</guard-timer-overhead>

    </supervision-watchdog-reset>
```

1.5.5: RPC: file-upload

Description

This RPC requests a file upload.

Attributes

- **xmlns**: defines the namespace for the supervision watchdog reset.

Sub-elements

- **local-logical-file-path**: specifies the local file on the NETCONF server to upload.
- **remote-file-path**: specifies the URL, where the file on the O-DU/NMS must be uploaded
- **password**: specifies the password for the file upload

See *ORAN-WG4.MP.0-v01.00 section 9.2* for more details.

```
<file-upload xmlns="urn:o-ran:file-management:1.0">

    <!-- This will be sent if only we receive success in
start query-->

        <local-logical-file-path/>

       <!-- Add the correct remote file path and password
to make this successful in
troubleshooting-log-location.json file -->
```

```
        <remote-file-path/>
        <password>
            <password/>
        </password>
    </file-upload>
```

1.5.6: RPC Actions

The following table lists various RPC actions along with their description.

RPC Actions	Description
start_supervision_thread	Should be used with <supervision-watchdog-reset/> to start a thread that will handle the supervision-notification events.
wait_for_locked_thread	Used in conjunction with getting <*sync xmlns="urn:o-ran:sync:1.0"*> to communicate, when the s-plane is locked.
create_new_vlan	Should be used with <edit-config/> RPCs that add new VLANs.
add_processing_element	Should be used with <edit-config/> RPCs that add new processing elements.
endpoint_carrier_configuration	Should be used with <edit-config/> RPCs that modify endpoint carrier configurations.
endpoint_carrier_link_configuration	Should be used with <edit-config/> RPCs that modify endpoint carrier link configurations.
activate_carrier	Should be used with <edit-config/> RPCs that activate carriers.
log_management_thread	Should be used with <start-troubleshooting-logs/> RPCs.
endpoint_carrier_configuration_negative	Used in negative tests for endpoint carrier configuration.

Section 1.6: M-Plane configuration examples

1.6.1: Supervisor Watchdog setup

```
<create-subscription
xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
        <stream>NETCONF</stream>
    </create-subscription>
    <supervision-watchdog-reset
xmlns="urn:o-ran:supervision:1.0"

action="start_supervision_thread">

<supervision-notification-interval>60</supervision-notific
ation-interval>
        <guard-timer-overhead>10</guard-timer-overhead>
    </supervision-watchdog-reset>
```

1.6.2: Get Modules Capabilities

```
<get xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
        <filter
xmlns:o-ran-module-cap="urn:o-ran:module-cap:1.0"
                type="xpath"
                select="/o-ran-module-cap:*//."/>
        <with-defaults
xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-with-defau
lts">report-all</with-defaults>
    </get>
```

NOTE

Similar examples are in the *default-call-flow.xml* file for the following procedures:

1. Getting Transceivers
2. Getting Interfaces
3. Getting Hardware
4. Getting U-Plane configuration
5. Various other examples

1.6.3: Test for Alarm Generation

```
<get xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
action="wait_for_locked_thread">
        <filter type="subtree">
            <sync xmlns="urn:o-ran:sync:1.0">
                <sync-status/>
            </sync>
        </filter>
    </get>
```

1.6.4: Create new VLAN

```
<edit-config
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
                    action="create_new_vlan">
        <target>
            <running/>
        </target>
        <default-operation>merge</default-operation>
        <config>
```

```
            <interfaces
xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
                <interface>
                    <name>KEYS_if_xyz_vlan0</name>
                    <type
xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">i
anaift:l2vlan</type>
                    <mac-address
xmlns="urn:o-ran:interfaces:1.0">00:03:a2:0a:00:00</mac-ad
dress>
                    <vlan-id
xmlns="urn:o-ran:interfaces:1.0">2</vlan-id>
                    <base-interface
xmlns="urn:o-ran:interfaces:1.0">MAC0</base-interface>
                </interface>
            </interfaces>
        </config>
    </edit-config>
```

1.6.5: LBM Configuration

```
<edit-config
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
        <target>
            <running/>
        </target>
        <default-operation>merge</default-operation>
        <config>
           <md-data-definitions xmlns="urn:o-ran:lbm:1.0">
                <maintenance-domain>
                    <id>KEYS_xyz_LBM</id>
                    <name>level_7</name>
```

```
                        <md-level>7</md-level>
                    </maintenance-domain>
                </md-data-definitions>
            </config>
        </edit-config>
```

1.6.6: Configure mplane-info

```
<edit-config
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
        <target>
            <running/>
        </target>
        <default-operation>merge</default-operation>
        <config>
            <mplane-info
xmlns="urn:o-ran:mplane-interfaces:1.0">
                <searchable-mplane-access-vlans-info>
                    <vlan-range>
                        <lowest-vlan-id>1</lowest-vlan-id>
                      <highest-vlan-id>1</highest-vlan-id>
                    </vlan-range>
                </searchable-mplane-access-vlans-info>
                <m-plane-interfaces>
                    <m-plane-ssh-ports/>
                </m-plane-interfaces>
            </mplane-info>
        </config>
    </edit-config>
```

1.6.7: Add Processing Element

```
<edit-config
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
                  action="add_processing_element">
        <target>
            <running/>
        </target>
        <default-operation>merge</default-operation>
        <config>
            <processing-elements
xmlns="urn:o-ran:processing-element:1.0">
<transport-session-type>ETH-INTERFACE</transport-session-t
ype>
                <ru-elements>
<name>KEYS_xyz_processing-element01</name>
                    <transport-flow>
<interface-name>KEYS_if_xyz_vlan0</interface-name>
                        <eth-flow>
<ru-mac-address>a0:ce:c8:17:7a:41</ru-mac-address>
                            <vlan-id>2</vlan-id>
<o-du-mac-address>38:AF:D7:D5:CE:EF</o-du-mac-address>
                        </eth-flow>
                    </transport-flow>
                </ru-elements>
            </processing-elements>
        </config>
    </edit-config>
```

1.6.8: Endpoint Carrier Configuration

```
<edit-config
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
                    action="endpoint_carrier_configuration"
                  comment="All the user-plane-configuration
is filled using o-ran-uplane-conf.json file">
        <target>
            <running/>
        </target>
        <default-operation>merge</default-operation>
        <config>
            <user-plane-configuration
xmlns="urn:o-ran:uplane-conf:1.0">
            </user-plane-configuration>
        </config>
    </edit-config>
```

1.6.9: Endpoint Carrier Link Configuration

```
<edit-config
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"

action="endpoint_carrier_link_configuration"
                  comment="All the user-plane-configuration
is filled using o-ran-uplane-conf.json file">
        <target>
            <running/>
        </target>
        <default-operation>merge</default-operation>
        <config>
            <user-plane-configuration
xmlns="urn:o-ran:uplane-conf:1.0"/>
```

```
        </config>
    </edit-config>
```

1.6.10: Activate Carrier

```
<edit-config
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
                  action="activate_carrier"
                comment="All the user-plane-configuration
is filled using o-ran-uplane-conf.json file">
        <target>
            <running/>
        </target>
        <default-operation>merge</default-operation>
        <config>
            <user-plane-configuration
xmlns="urn:o-ran:uplane-conf:1.0"/>
        </config>
    </edit-config>
```

1.6.11: Log Management

```
<start-troubleshooting-logs
xmlns="urn:o-ran:troubleshooting:1.0"

action="log_management_thread"/>
    <file-upload xmlns="urn:o-ran:file-management:1.0">
    <!-- This will be sent if only we receive success in
start query-->
        <local-logical-file-path/>
       <!-- Add the correct remote file path and password
to make this successful in
            troubleshooting-log-location.json file -->
```

```
        <remote-file-path/>
        <password>
            <password/>
        </password>
    </file-upload>
```

1.6.12: Example: flow for M-Plane RPC configuration

The following steps show how to configure M-Plane RPC, based on *Fig. 24* in *ORAN-WG4-MP-0-V01.00* specification.

1 NETCONF Client / O-RU Controller sets parameters of objects on the NETCONF Server / O-RU

```
<rpc> <edit-config><source><running/><…>
```

2 Upon receiving a value change request from the O-RU Controller, the O-RU automatically updates the values.

3 NETCONF Server / O-RU sends an acknowledgement to the NETCONF Client / O-RU Controller

```
<rpc-reply> <OK>
<rpc-reply> <rpc-error> <error-xxx>
```

Section 1.7: References

The following documents were referred to, for the implementation of M-Plane and to prepare this document:

- ORAN-WG4.MP.0-v01.00 - O-RAN Alliance Working Group 4 Management Plane Specification
- RFC 6241, “Network Configuration Protocol (NETCONF)”, IETF, June 2011
- RFC 7950, “The YANG 1.1 Data Modeling Language”, IETF, August 2016

Section 1.8: Contact us

For more information on Keysight Technologies' products, applications or services, contact your local Keysight office. The complete list is available at: www.keysight.com/find/contactus.

This page has been intentionally left blank.

Edition 1.0, April 2021

www.keysight.com

KEYSIGHT
TECHNOLOGIES