ENVISIONING • EMPOWERING • EXCELLING

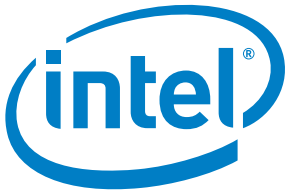
In 2020, MaxLinear acquired the
Connected Home Division business of Intel Corporation,
including the former Intel® product/s referenced in
the title of the attached material.

The MaxLinear logo will be added to the
attached material upon its next revision.

MaxLinear is now the manufacturer of this product, direct any questions and product support requests to your MaxLinear sales contact, MaxLinear Sales Representative or Distributor, or login to your myMxL account and create a new support ticket.

MaxLinear

Corporate Headquarters:
5966 La Place Court
Suite 100
Carlsbad, CA 92008
Tel.:+1 (760) 692-0711
Fax: +1 (760) 444-8598
www.maxlinear.com



Intel® 10G PON Chipset

Intel® 10G PON Development Kit EASY PRX321 REF BOARD

PRX321 SFU Reference Board V1.2/V1.3

Getting Started

Revision 1.0, 2018-11-06
Intel Confidential
Reference ID 617831

Legal Notice

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document. Intel disclaims all warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

All information provided here is subject to change without notice. Intel may make changes to its test conditions and internal reliability goals at any time. Contact your Intel representative to obtain the latest Intel product specifications and road-maps.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Revision History

Current:	**Revision 1.0, 2018-11-06**
Previous:	**None**

Page	Major changes since previous revision

Table of Contents

List of Figures

Preface

This is the Getting Started document for the Intel® 10G PON Development Kit EASY PRX321 REF BOARD V1.2/V1.3, which is a demonstration platform to show the Intel® 10G PON Chipset PRX321 device used in a business 10G SFU application.

To simplify matters, the following synonym is used:

EASY PRX321 REF BOARD

Synonym used for the Intel® 10G PON Development Kit EASY PRX321 REF BOARD. The board version can be also be added if required.

Organization of this Document

This document is organized as follows:

- **Chapter 1, Introduction**
  This chapter describes the contents of the package, gives an overview of the board connectors and configuration jumpers, and explains how to connect to the serial interface of the board.
- **Chapter 2, Bootloader Preparation**
  This chapter describes how to install the boot loader and the operating system image on the board.
- **Chapter 3, Basic Operation**
  This chapter describes how to connect the optical interface, and gives some command line examples.
- **Literature References**

1 Introduction

The EASY PRX321 REF BOARD is a reference and demonstration platform for the Intel® 10G PON Chipset PRX321 device. This document is a Getting Started document and describes all the steps for initial operation of the board.

1.1 Contents of Package

The package contains:

- EASY PRX321 REF BOARD V1.2 or EASY PRX321 REF BOARD V1.3
- Power supply: 12 V/110 V - 240 V
- Micro Flat USB cable
- XGSPON optical transceiver (SC/UPC Connector Bi-directional, “Blue”)

1.2 Board Overview

Figure 1 shows the EASY PRX321 REF BOARD V1.2:

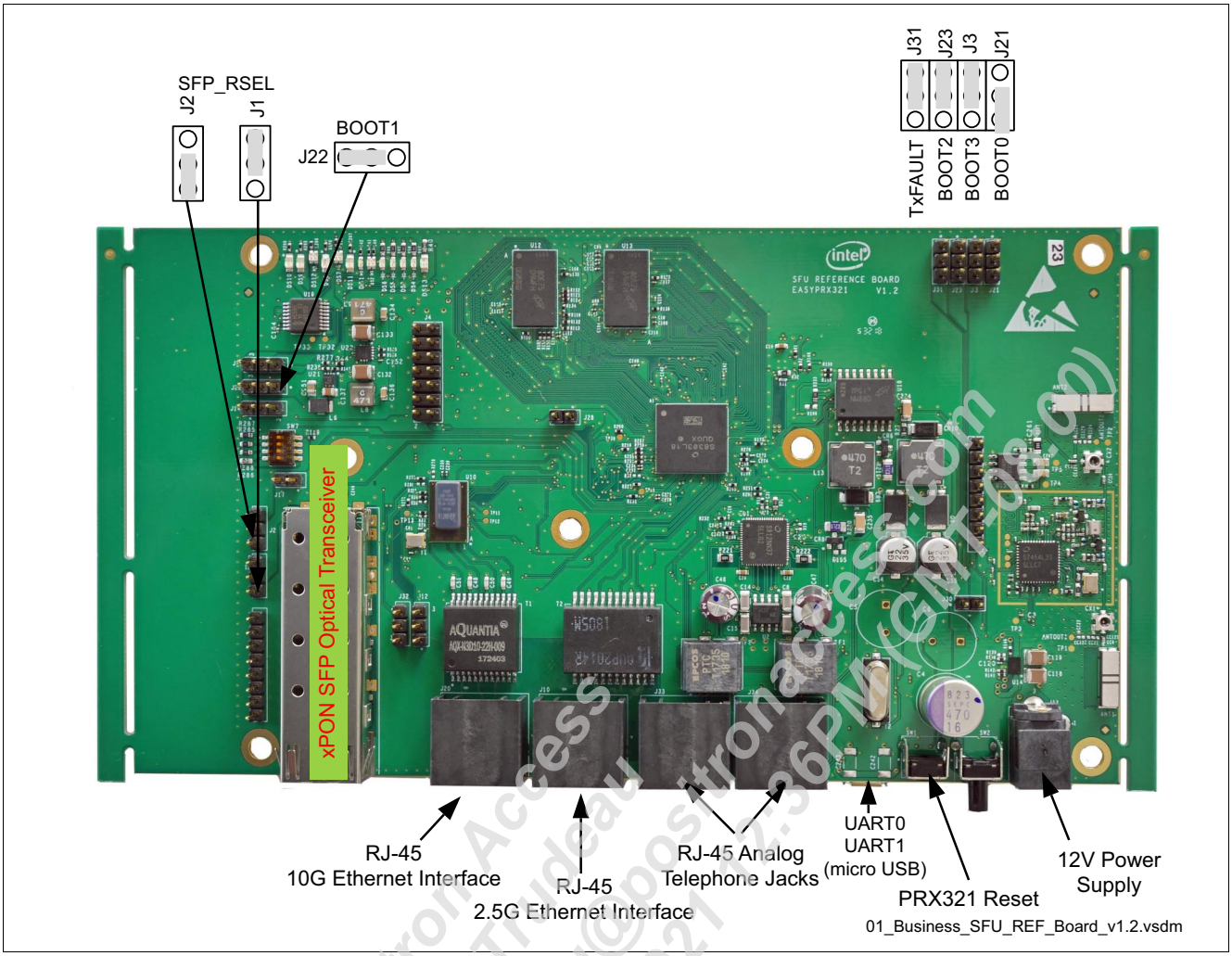


Figure 1 Overview of the EASY PRX321 REF BOARD V1.2

The optical transceiver SFP is used as the up-link to the OLT (Optical Line Termination). Two different Ethernet interfaces are dedicated to user traffic, software image update or control interfaces (for example: telnet, TFTP). Two analog telephones can be connected to the POTS interfaces. For debugging purposes, 2 serial ports, UART0 and UART1, are accessible via a micro USB, FTDI FT2232D device.

The EASY PRX321 REF BOARD V1.2 and V1.3 are pre-configured in QSPI flash mode. This is especially relevant when selecting the software images to be used.

Note: EASY PRX321 REF BOARD V1.3 is equivalent to V1.2 with some bug fixes. Same procedures apply for both boards.

1.3 Connecting the Board

In order to use the serial port, a valid FTDI software driver supporting FT2232D is required, which can be downloaded from the FTDI website. UART0 and UART1 will be detected as two COM ports, as shown in **Figure 2**. The assigned COM numbers can vary in different cases.

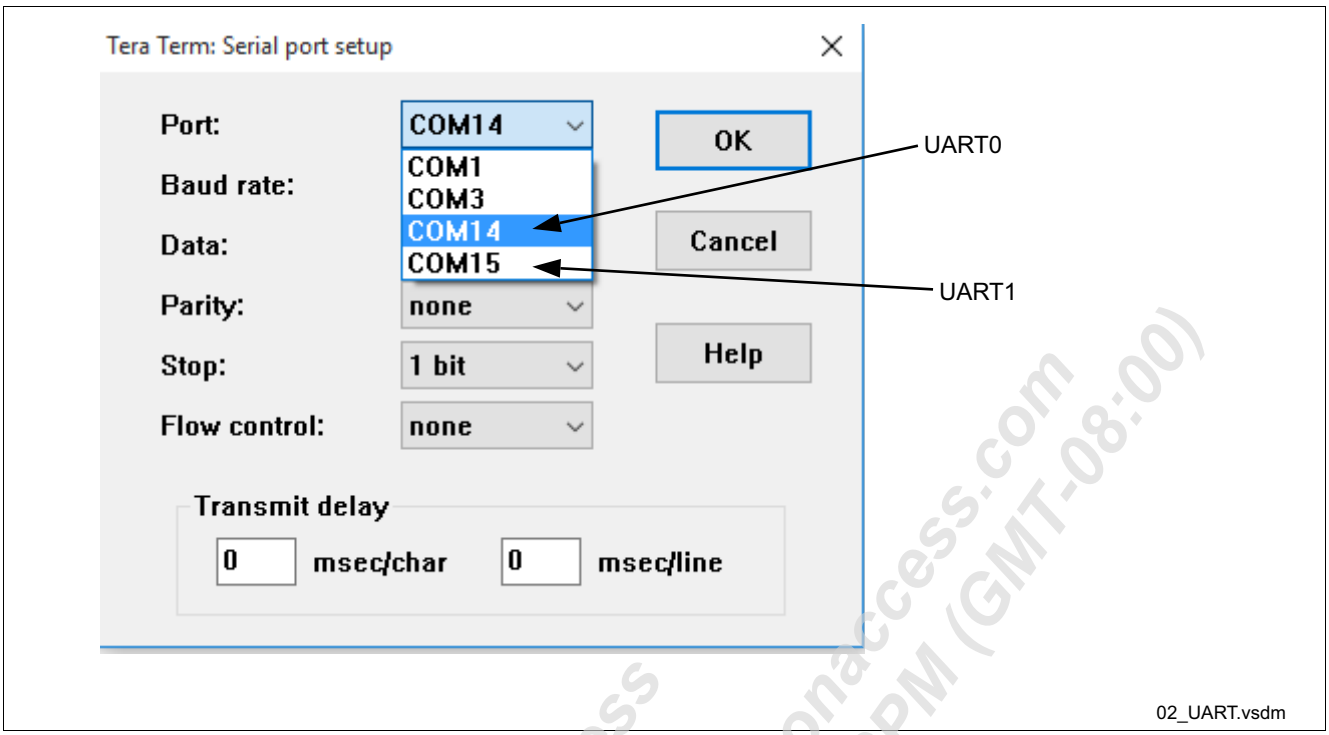


Figure 2 Example: UART0 and UART1 Detected as COM14 and COM15

For debugging purposes, connect the PC to the UART0 of the reference board. Use terminal software on the PC with the following parameters:

- 115200 baud
- 8-bit data
- No parity
- 1 stop bit
- No flow control

The board has a pre-installed Linux* software image. If the software needs to be updated, an Ethernet connection is required between the 10G Ethernet port and the PC/LAN. The 10G Ethernet Interface can operate in 1G mode if the mode is supported on the host PC, but not 100 Mbps or 10 Mbps.

Connect the power socket to the power supply, and connect the power supply to an AC outlet.

Note: The terminal needs to be reconnected to the COM port after each power-cycle of the board.

2 Bootloader Preparation

This is needed in two main cases: either the board has never been flashed before, or the images just need to be updated. Please select U-Boot and Linux* image files depending on the board and configuration.

2.1 Boot Mode Selection

The correct boot mode is configured on the delivered boards. However, it may be necessary to modify the boot mode under certain circumstances.

2.1.1 Boot from UART

It may be necessary to initiate a U-Boot installation if the flash memory has not been programmed before, or if the boot partition has been damaged. In this case, update the U-Boot in legacy UART0 boot mode.

Configure jumpers to select UART0 boot mode“0x4”, see **[1]**:

```
0 boot0 with J21
0 boot1 with J22
1 boot2 with J23
0 boot3 with J3
```

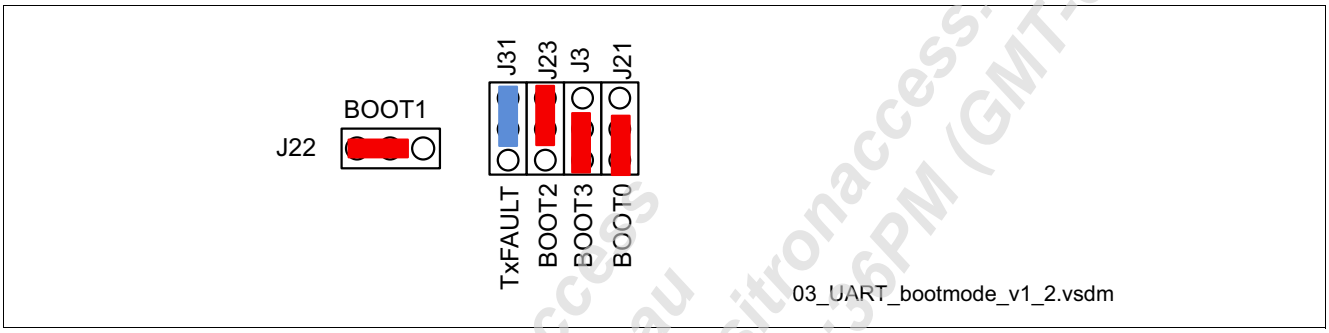
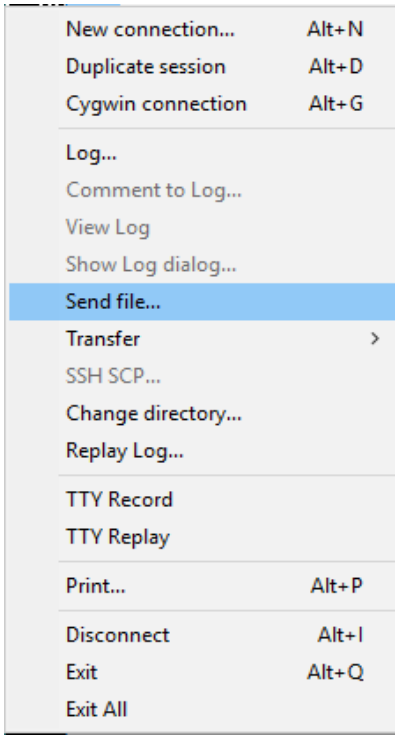


Figure 3 Jumper Settings for UART Boot of PRX321 SFU Reference Board V1.2 and V1.3

Power on the board and send `*.asc` file[1)] over the serial port as a text file:

1) u-boot.asc can be found in *<sdk_folder>/ugw_sw/openwrt/bin/targets/intel_mips/falcon_mountain/prx321_sfu/**. If the file is not generated, contact Intel for assistance.



03_UART_SendFile.vsdm

Figure 4 Sending File Over Terminal with Tera Term

The LOG:

```
ROM UART only
CFG 04
B
UART
************************DdrOk
<<< LINES INTENTIONALLY REMOVED >>>
*********************************************
U-Boot 2016.07-INTEL-v-3.1.102-00141-g925ecf8-dirty (Jun 19 2018 - 14:18:14 +0800)
4Kec
DRAM:  128 MiB
NAND:  device found, Manufacturer ID: 0x2c, Chip ID: 0x24
256 MiB
Bad block table found at page 131008, version 0x01
Bad block table found at page 130944, version 0x01
In:    serial
Out:   serial
Err:   serial
Net:   No ethernet found.
Type "run flash_nfs" to mount root filesystem over NFS
Hit any key to stop autoboot:  0
```

In U-Boot prompt, hit any key to interrupt the boot process. Continue with the steps described in **Chapter 2.2.1**, as U-boot is not yet stored in the flash memory.

2.1.2 Boot from Previously Installed U-Boot, QSPI Mode

This is the default mode of EASY PRX321 REF BOARD.

```
0 boot0 with J21
0 boot1 with J22
1 boot2 with J23
1 boot3 with J3
```

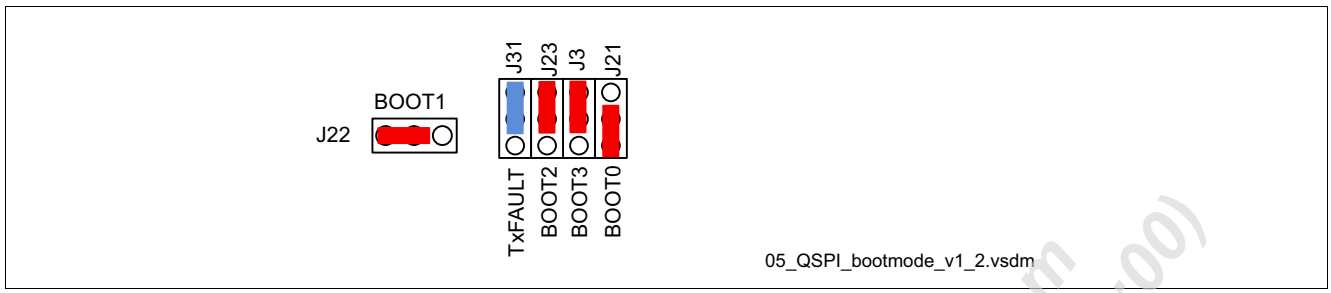


Figure 5 Jumper Settings for QSPI Boot of PRX321 SFU Reference Board V1.2 and V1.3

2.2 U-Boot Update

Once the initial U-Boot has been booted, using any of the boot modes described in **Chapter 2.1**, interrupt the U-Boot prompt by hitting any key:

```
Hit any key to stop autoboot: 5
```

There are two options, download new U-Boot via serial port, or via Ethernet port.

2.2.1 U-Boot Update via Serial Port

In order to download the U-Boot image via the serial port:

- Enable file reception:

```
FALCON # loadb 0xa0400000
## Ready for binary (kermit) download to 0xA0400000 at 115200 bps...
## Total Size      = 0x0002bfd0 = 180176 Bytes
## Start Addr      = 0xA0400000
```

- Transmit the u-boot-nand.bin file using Kermit protocol. For example use Tera Term (File -> Transfer -> Kermit -> Send)
- When the binary file is successfully transferred to the DDR memory, erase the flash:

```
FALCON # nand erase 0 40000
NAND erase: device 0 offset 0x0, size 0x40000
Erasing at 0x20000 -- 100% complete.
OK
```

- Flash the U-Boot to the NAND flash:

```
FALCON # nand write a0400000 0 40000
NAND write: device 0 offset 0x0, size 0x40000
 262144 bytes written: OK
```

- Switch off the board, change the boot mode if needed, and switch on.

2.2.2 U-Boot Update via Ethernet

Attention: Update via Ethernet not supported from uboot.asc.

In order to download the U-Boot image via the Ethernet port:

- Adapt your IP address parameters to your network, for example:

```
FALCON # setenv ethaddr 00:50:f1:50:ec:72
FALCON # setenv ipaddr 10.91.185.7
FALCON # setenv serverip 10.91.162.11
FALCON # setenv netmask 255.255.224.0
FALCON # setenv gatewayip 10.91.160.1
FALCON # setenv tftppath
FALCON # saveenv
```

- Connect the board 10G Ethernet port (connector J20) to the server PC. Note that the port also supports 1 Gbps mode, but not 100 Mbps or 10 Mbps.

- Verify connectivity:

```
FALCON # ping 10.91.162.11
Using Falcon-MX-Switch device
host 10.91.162.11 is alive
```

- Make sure that U-Boot variables *serverip* and *tftppath* are correctly configured and execute *run update_uboot*. Expected log:

```
FALCON # run update_uboot
Using FMX-Switch device
TFTP from server 10.91.162.11; our IP address is 10.91.185.7
Filename 'uboot-prx321-sfu-qspi-nand/u-boot-nand.bin'.
Load address: 0x82000000
Loading: ###############
         38.1 KiB/s
done
Bytes transferred = 206816 (327e0 hex)

NAND erase: device 0 offset 0x0, size 0x40000
Erasing at 0x20000 -- 100% complete.
OK

NAND write: device 0 offset 0x0, size 0x327e0
 206816 bytes written: OK
```

2.2.3 U-Boot Update via Ethernet (SDK 0.8.3 and older)

Additional steps are needed if you are using an U-Boot compiled from SDK0.8.3 or older. As a reference, it contains *2016.07-INTEL-v-3.1.116* U-Boot version.

Attention: Update via Ethernet not supported from uboot.asc.

In order to download the U-Boot image via the Ethernet port:

- Release 10G PHY reset with:

```
FALCON # sso init
FALCON # sso set 29 on 0xff
```

- Connect the board 10G Ethernet port (connector J20) to the server PC. Note that the port also supports 1 Gbps mode, but not 100 Mbps or 10 Mbps.

- If the U-boot is being flashed for the first time, or the U-boot variable environment was damaged, you will need to set the required variables:

```
FALCON # setenv aquantia_link_mode 10G_KR_MODE
FALCON # setenv xpcs-mode 10G_KR_MODE
```

- Adapt your IP address parameters to your network, for example:

```
FALCON # setenv ethaddr 00:50:f1:50:ec:72
FALCON # setenv ipaddr 10.91.185.7
FALCON # setenv serverip 10.91.162.11
FALCON # setenv netmask 255.255.224.0
FALCON # setenv gatewayip 10.91.160.1
```

- Verify connectivity:

```
FALCON # ping 10.91.162.11
```

```
Using Falcon-MX-Switch device
host 10.91.162.11 is alive
```

- Set up a tftp server on a PC to the U-Boot image file.
- Transfer the file:

```
FALCON # tftp a0400000 u-boot-nand.bin
Using Falcon-MX-Switch device
TFTP from server 10.91.162.11; our IP address is 10.91.187.7
Filename 'u-boot-nand.bin'.
Load address: 0xa0400000
Loading: #############
         34.2 KiB/s
done
Bytes transferred = 180176 (2bfd0 hex)
```

- When the binary file has been transferred to the DDR memory successfully, erase the flash:

```
FALCON # nand erase 0 40000
NAND erase: device 0 offset 0x0, size 0x40000
Erasing at 0x20000 -- 100% complete.
OK
```

- Copy the U-Boot from RAM to the NAND flash:

```
FALCON # nand write a0400000 0 40000
NAND write: device 0 offset 0x0, size 0x40000
 262144 bytes written: OK
```

2.3 U-Boot Environment Variables

U-Boot requires the environment variables to be set with correct values.

2.3.1 U-Boot Environment Reset

This is optional.

The variables only need to be reset if U-Boot is updated to a much newer version with different variables. If the environment partition is removed, U-Boot will detect and use default variables with correct values, which can then be saved to get an environment with correct values.

Note: ***Create a backup of the current variables before resetting the environment, show them using printenv***

```
FALCON # nand erase ${f_ubootconfig_addr} ${f_ubootconfig_range}; nand erase
${f_red_ubootconfig_addr} ${f_ubootconfig_range};

NAND erase: device 0 offset 0x100000, size 0x40000
Erasing at 0x120000 -- 100% complete.
OK

NAND erase: device 0 offset 0x140000, size 0x40000
Erasing at 0x160000 -- 100% complete.
OK
FALCON # reset
```

2.3.2 U-Boot Environment Initialization

If the U-Boot environment has been lost, the following configuration is required:

- Adapt your IP address parameters to your network, for example:

```
FALCON # setenv ipaddr 10.91.185.7
FALCON # setenv netmask 255.255.224.0
FALCON # setenv gatewayip 10.91.160.1
FALCON # setenv serverip 10.91.162.11
FALCON # setenv tftppath
```

- Save the environment to the flash memory:

```
FALCON # saveenv
```

Note: the following variables are only needed if the U-Boot environment variables are not present yet or the environment has been reset.

- Set a correct MAC address if the one set by Intel was lost. Please note that the MAC address must be unique. Example:

```
FALCON # setenv ethaddr 00:50:f1:50:ec:72
FALCON # saveenv
```

2.3.3 Additional U-Boot Variables (SDK 0.8.3 and older)

Additional settings required:

```
FALCON # setenv ubi_init 'setenv kernel_vol kernel; setenv rootfs_vol rootfs;setenv
firmware_vol firmware; setenv bootcore_vol bootcore; setenv rootfsname rootfs; ubi
part system_sw'
FALCON # setenv addmisc 'setenv bootargs ${bootargs} earlycon=lantiq,0x16380000
console=ttyLTQ0,115200 ethaddr=$(ethaddr)
ip=$(ipaddr):$(serverip):$(gatewayip):$(netmask):$(hostname):$(netdev):on panic=1
${mtdparts} maxcpus=4 nr_cpus=4 nocoherentio clk_ignore_unused mem=94M@512M
rcupdate.rcu_cpu_stall_suppress=1 loglevel=8'
FALCON # setenv mtdids 'nand0=nand.0'
FALCON # setenv mtdparts
'mtdparts=nand.0:1m(uboot),256k(ubootconfigA),256k(ubootconfigB),256k(gphyfirmware
),1m(calibration),16m(bootcore),108m(system_sw),-(res)'
FALCON # setenv flashargs 'setenv bootargs rootfsname=rootfs ro
ubi.mtd=system_sw,2048 rootfstype=squashfs'
FALCON # setenv aquantia_link_mode 10G_KR_MODE
FALCON # setenv xpcs-mode 10G_KR_MODE
FALCON # setenv loadaddr 0x81000000
FALCON # saveenv
```

2.4 Software Image

After U-Boot, the Linux kernel and file system image must be installed in the system.

2.4.1 Remove UBI Partitions

This is optional.

If there is an error, erase the flash partitions and try again.

```
FALCON # run ubi_init; ubi remove rootfs; ubi remove rootfs_data; ubi remove kernel
ubi0: attaching mtd1
ubi0: scanning is finished
ubi0: attached mtd1 (name "mtd=6", size 108 MiB)
ubi0: PEB size: 131072 bytes (128 KiB), LEB size: 126976 bytes
ubi0: min./max. I/O unit sizes: 2048/2048, sub-page size 2048
ubi0: VID header offset: 2048 (aligned 2048), data offset: 4096
ubi0: good PEBs: 864, bad PEBs: 0, corrupted PEBs: 0
ubi0: user volume: 7, internal volumes: 1, max. volumes count: 128
ubi0: max/mean erase counter: 8/4, WL threshold: 4096, image sequence number: 0
ubi0: available PEBs: 0, total reserved PEBs: 864, PEBs reserved for bad PEB
handling: 40
Remove UBI volume rootfs (id 4)
Remove UBI volume rootfs_data (id 5)
Remove UBI volume kernel (id 3)
FALCON #
```

2.4.2 Software Image Update via Ethernet

Command install the software image via the Ethernet port:

```
FALCON # run update_fullimage
ubi0: attaching mtd1
ubi0: scanning is finished
ubi0: attached mtd1 (name "mtd=6", size 108 MiB)
ubi0: PEB size: 131072 bytes (128 KiB), LEB size: 126976 bytes
ubi0: min./max. I/O unit sizes: 2048/2048, sub-page size 2048
ubi0: VID header offset: 2048 (aligned 2048), data offset: 4096
ubi0: good PEBs: 864, bad PEBs: 0, corrupted PEBs: 0
ubi0: user volume: 2, internal volumes: 1, max. volumes count: 128
ubi0: max/mean erase counter: 2/1, WL threshold: 4096, image sequence number: 0
ubi0: available PEBs: 780, total reserved PEBs: 84, PEBs reserved for bad PEB
handling: 40
Using FMX-Switch device
TFTP from server 10.91.161.128; our IP address is 10.91.187.149
Filename 'lede-intel_mips-falcon_mountain-PRX321_SFU_QSPI_PON-squashfs-
fullimage.img'.
Load address: 0x82000000
Loading: ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##################
         1.3 MiB/s
```

```
done
Bytes transferred = 5030376 (4cc1e8 hex)
Image contains header with name [SFU_PON_0.9.0]
Volume kernelB not found!
Creating dynamic volume kernelB of size 2135482
Erasing NAND...
Erasing at 0x120000 -- 100% complete.
Writing to NAND... OK
Image contains header with name [LEDE RootFS]
Volume rootfsB not found!
Creating dynamic volume rootfsB of size 2894752
Erasing redundant NAND...
Erasing at 0x160000 -- 100% complete.
Writing to redundant NAND... OK
Image contains header with name [SFU_PON_0.9.0]
Volume kernelA not found!
Creating dynamic volume kernelA of size 2135482
Erasing NAND...
Erasing at 0x120000 -- 100% complete.
Writing to NAND... OK
Image contains header with name [LEDE RootFS]
Volume rootfsA not found!
Creating dynamic volume rootfsA of size 2894752
Erasing redundant NAND...
Erasing at 0x160000 -- 100% complete.
Writing to redundant NAND... OK
```

If you see "UBI Error -28 creating volume" run:

```
FALCON # run reset_sysconfig
ubi0: attaching mtd1
ubi0: scanning is finished
ubi0: attached mtd1 (name "mtd=6", size 108 MiB)
ubi0: PEB size: 131072 bytes (128 KiB), LEB size: 126976 bytes
ubi0: min./max. I/O unit sizes: 2048/2048, sub-page size 2048
ubi0: VID header offset: 2048 (aligned 2048), data offset: 4096
ubi0: good PEBs: 864, bad PEBs: 0, corrupted PEBs: 0
ubi0: user volume: 3, internal volumes: 1, max. volumes count: 128
ubi0: max/mean erase counter: 2/1, WL threshold: 4096, image sequence number: 0
ubi0: available PEBs: 2, total reserved PEBs: 862, PEBs reserved for bad PEB
handling: 40
Remove UBI volume rootfs_data (id 2)
```

Finally, boot the system:

```
FALCON # run flash_flash
```

2.4.3 Software Image Update via Ethernet (SDK 0.8.3 and older)

Steps to install the software image via the Ethernet port in board with U-boot v3.1.116 or older:

- Release 10G PHY from reset to also be used in U-Boot:

```
FALCON # sso init
FALCON # sso set 29 on 0xff
```

- Connect the board 10G Ethernet port to the server PC (port supports 1 Gbps mode as well, but not 100 Mbps or 10 Mbps).
- Verify connectivity:

```
FALCON # ping 10.91.162.11
Using Falcon-MX-Switch device
host 10.91.162.11 is alive
```

- Set up the TFTP server in the PC to point to the software image file folder containing the correct image[1].
- Transfer the desired file:

```
FALCON # tftp $(loadaddr) lede-intel_mips-falcon_mountain-PRX321_SFU_QSPI_PON-
squashfs-fullimage.img

Using Falcon-MX-Switch device
TFTP from server 10.91.162.11; our IP address is 10.91.187.7
Filename 'lede-intel_mips-falcon_mountain-PRX321_SFU_QSPI_PON-squashfs-
fullimage.img'.
Load address: 0x81000000
Loading: ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##################################################
         3.8 MiB/s
done
Bytes transferred = 4515272 (44e5c8 hex)
```

- Store the image to the flash memory.

```
FALCON # run ubi_init; upgrade ${loadaddr} ${filesize}
ubi0: attaching mtd1
ubi0: scanning is finished
ubi0: attached mtd1 (name "mtd=6", size 108 MiB)
ubi0: PEB size: 131072 bytes (128 KiB), LEB size: 126976 bytes
ubi0: min./max. I/O unit sizes: 2048/2048, sub-page size 2048
ubi0: VID header offset: 2048 (aligned 2048), data offset: 4096
ubi0: good PEBs: 864, bad PEBs: 0, corrupted PEBs: 0
ubi0: user volume: 3, internal volumes: 1, max. volumes count: 128
ubi0: max/mean erase counter: 2/1, WL threshold: 4096, image sequence number: 0
ubi0: available PEBs: 0, total reserved PEBs: 864, PEBs reserved for bad PEB
handling: 40
Image contains header with name [SFU_PON_0.7.0.124]
Remove UBI volume kernel (id 1)
Creating dynamic volume kernel of size 2075220
```


1) Images can be found in *<sdk_folder>/ugw_sw/openwrt/bin/targets/intel_mips/falcon_mountain/prx321_sfu*

```
Erasing NAND...
Erasing at 0x120000 -- 100% complete.
Writing to NAND... OK
Image contains header with name [LEDE RootFS]
Remove UBI volume rootfs (id 0)
Creating dynamic volume rootfs of size 2439890
Erasing redundant NAND...
Erasing at 0x160000 -- 100% complete.
Writing to redundant NAND... OK
```

- If a *UBI Error -28* occurs while creating volume. There is not enough space in the ubi volume, execute the following command:

```
FALCON # run ubi_init ; ubi remove rootfs_data
ubi0: detaching mtd1
ubi0: mtd1 is detached
ubi0: attaching mtd1
ubi0: scanning is finished
ubi0: attached mtd1 (name "mtd=6", size 108 MiB)
ubi0: PEB size: 131072 bytes (128 KiB), LEB size: 126976 bytes
ubi0: min./max. I/O unit sizes: 2048/2048, sub-page size 2048
ubi0: VID header offset: 2048 (aligned 2048), data offset: 4096
ubi0: good PEBs: 864, bad PEBs: 0, corrupted PEBs: 0
ubi0: user volume: 3, internal volumes: 1, max. volumes count: 128
ubi0: max/mean erase counter: 2/1, WL threshold: 4096, image sequence number: 0
ubi0: available PEBs: 0, total reserved PEBs: 864, PEBs reserved for bad PEB
handling: 40
Remove UBI volume rootfs_data (id 2)
```

- Repeat the command:

```
FALCON # run ubi_init; upgrade  ${loadaddr} ${filesize}
....
```

- Power-cycle the board, press the reset button (SW1) or type *reset*.

3 Basic Operation

Plug the optical module into the SFP cage and connect the EASY PRX321 REF BOARD to the OLT.

If the board is equipped with the XGSPON optical transceiver, which is delivered with the package, please use the SC/UPC connector shown in **Figure 6**.

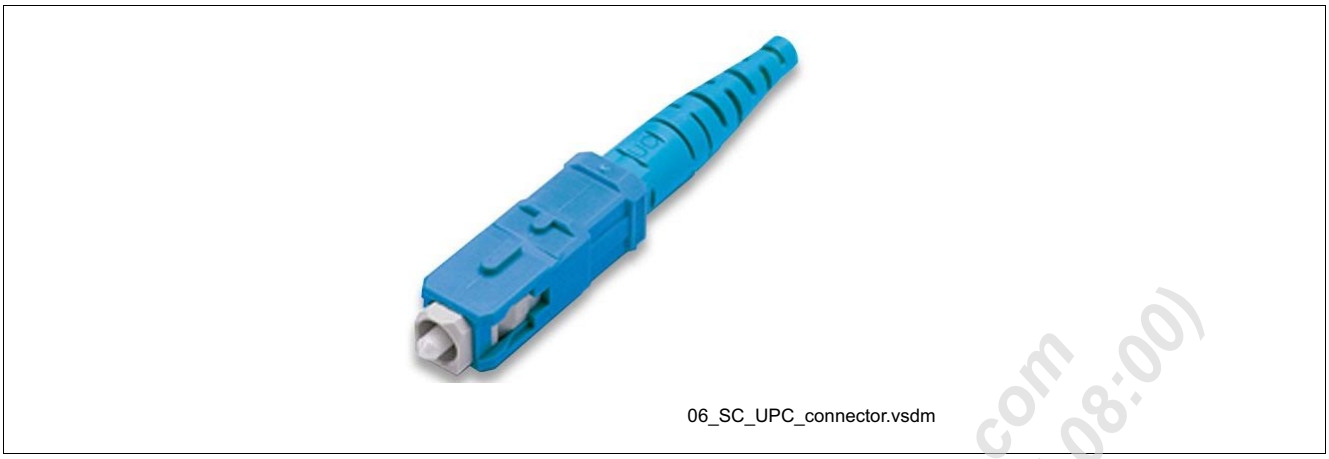


Figure 6 SC/UPC Connector

Attention: Always use 10 dB of attenuation or more.

Power up the board.

To stop U-Boot from auto booting the operating system, hit a key during the U-Boot start-up. If not, the Operating System will return to enter a Linux* console.

Example

To verify the PLOAM state, enter:

```
root@easy:/# pon ploam_state_get
errorcode=0 current=50 previous=40 time_prev=6120
```

Example

Monitor the PLOAM states using the *pond* application. Run the application in the background and reconnect the fiber to observe the changes:

```
root@easy:/# pond&
root@easy:/# alarm Loss of GEM channel delineation set
alarm Loss of GEM channel delineation cleared
alarm Loss of GEM channel delineation set
alarm Loss of GEM channel delineation cleared
alarm Loss of GEM channel delineation set
alarm Loss of GEM channel delineation cleared
alarm Loss of signal cleared
alarm Loss of downstream synchronization set
ploam state: previous - 50, current - 60
ploam state: previous - 60, current - 11
alarm Loss of signal set
alarm Loss of signal cleared
alarm Loss of signal set
alarm Loss of downstream synchronization cleared
ploam state: previous - 11, current - 12
```

```
ploam state: previous - 12, current - 23
ploam state: previous - 23, current - 40
ploam state: previous - 40, current - 50
```

Example

To print out the serial number of the ONU, use:

```
root@easy:/# pon serial_number_get
errorcode=0 serial_no="INTCF150EC2B"
```

Example

If needed, change the vendor ID as a part of the serial number using:

```
root@easy:/# vi /etc/config/gpon
....
option nSerial 'INTCF150EC2B'
```

Example

Use the PTOP tool to check current status. Select the option, the following options are available:

```
root@easy:/# ptop
....
w       Active alarms                g-s     GEM/XGEM Port Status
g-c     GEM/XGEM Port Counters       g-d     GEM/XGEM port DS Counters
g-u     GEM/XGEM port US Counters    f       FEC Status & Counters
t       GTC/XGTC Status & Counters   p-s     Power Save Status
p-c     PSM Configuration            a-c     Allocation Counters
p-d     PLOAM Downstream Counters    p-u     PLOAM Upstream Counters
o-s     Optical Interface Status     o-i     Optical Interface Info
d-b     Debug Burst Profile
```

Example of the option “o-i”:

```
SFP+ information                              Status
Vendor name                                 : SOURCEPHOTONICS
Vendor oui                                  :
Part number                                 : SPPS27XER3CDFD
Revision                                    : 1
Serial number                               : I462006526
Date code                                   : 180305
Wavelength                                  : 1270 nm

Options
------------
Power level declaration                     : Power Level 1
Paging implemented indicator                : No
Retimer / CDR indicator                     : No
Cooled Transceiver declaration              : No
Linear Receiver Output implemented          : No
Receiver decision threshold implemented     : No
Tunable transmitter technology              : No
RATE_SELECT functionality implemented       : No
```

```
TX_DISABLE implemented                              : No
TX_FAULT implemented                                : Yes
Inverted loss of signal implemented                 : No
Loss of signal implemented                          : Yes

DMI type
-------------
Digital monitoring implemented                      : Yes
Internally calibrated                               : Yes
Externally calibrated                               : No
Received power measurement type                     : Average
Address change required                             : No

Enhanced options
---------------------
Optional Alarm/Warnings flags implemented           : Yes
Soft TX_DISABLE control and monitoring implemented  : Yes
Soft TX_FAULT monitoring implemented                : Yes
Soft RX_LOS monitoring implemented                  : Yes
Soft RATE_SELECT ctrl and monitoring implemented    : No
Application select control implemented              : No
Soft RATE_SELECT control implemented                : No

Compliance                                          : SFF-8472 Rev 9.5
```

Example of the option o-i:

```
GEM Index GEM ID   u/s packets u/s bytes d/s packets  d/s bytes       Key Errors
0         1        254         12192     250         12000           0
1         1022     0           0         0           0               0
```

Example of the option g-s:

```
GEM Index  GEM ID  Alloc Id  Alloc Id st.  Data/OMCI  Max.Size  Encr. k.r. Direction
0          1       1          Valid        OMCI       1980      None       DS + US
1          1022    1025       Valid        Ethernet   1628      None       DS + US
```

Literature References

[1] Intel® 10G PON Chipset PRX321 (PRX321A1BI) Preliminary Data Sheet Rev. 1.1, in preparation

[2] Intel® 10G PON Chipset 10G PON Subsystem Preliminary User's Manual Programmer's Guide Rev. 1.2

[3] Intel® 10G PON Development Kit EASY PRX321 REF BOARD V1.2/V1.3 (SFU) User's Manual Hardware Description Rev. 1.0

Attention: Please refer to the latest revisions of the documents.