

# **Intel Atom® Processor E3900 Series/ Intel Atom® Processor E3800 Product IPSO 6LoWPAN IoT Software 3.0 for Yocto Project\***

**User Guide**

---

*July 2017*



You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

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.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting: <http://www.intel.com/design/literature.htm>

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at <http://www.intel.com/> or from the OEM or retailer.

No computer system can be absolutely secure.

Intel, Atom, and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

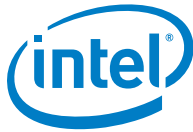
\*Other names and brands may be claimed as the property of others.

Copyright © 2017, Intel Corporation. All rights reserved.



# Contents

<b>1.0</b>	<b>Introduction.....</b>	<b>7</b>
1.1	Terminology .....	7
1.1	Reference Documents .....	7
<b>2.0</b>	<b>Before You Begin .....</b>	<b>9</b>
<b>3.0</b>	<b>6LoWPAN IoT Software Overview .....</b>	<b>11</b>
<b>4.0</b>	<b>Hardware and Software Compatibility.....</b>	<b>13</b>
4.1	Hardware Supported .....	13
4.2	Software Package.....	14
<b>5.0</b>	<b>Build Image with 6LoWPAN Software Stack .....</b>	<b>15</b>
5.1	Prepare Yocto BSP Image.....	15
5.1.1	Build Base Image for Apollo Lake-I .....	15
5.1.2	Build Base Image for Baytrail-I .....	15
5.2	6LoWPAN Setup Procedure.....	17
5.3	Build Yocto Project* Image with 6LoWPAN Software Stack.....	18
5.4	Prepare SD Card or USB as Bootable Media.....	19
<b>6.0</b>	<b>Building the Firmware for Serial Radio and Node.....</b>	<b>21</b>
6.1	Build the Firmware for Node.....	21
6.2	Build the Firmware for Serial Radio.....	23
6.2.1	Flashing Firmware into Hardware Device .....	24
<b>7.0</b>	<b>Running the 6LoWPAN IoT Software.....</b>	<b>26</b>
7.1	Preparation .....	26
7.2	Setting up Network Configuration .....	26
7.2.1	Set up WiFi/LAN Connection with Static IPv6 for Apollo Lake-I .....	26
7.2.2	Set up LAN Connection with Static IPv6 for Baytrail-I .....	27
7.3	Starting Network Border Router .....	27
7.4	Using Static IPv6 Method.....	29
7.5	Checking the Nodes Registered to Serial Radio.....	30
<b>8.0</b>	<b>Accessing the 6LoWPAN IoT Software .....</b>	<b>31</b>
8.1	Access 6LoWPAN Network with CoAP Client.....	31
8.1.1	Firefox Web Browser GUI CoAP Client .....	31
8.1.2	Command-line-Interface CoAP Client.....	32
8.2	Lightweight Machine to Machine (LWM2M) .....	33
8.2.1	Configuring and Running LWM2M Bootstrap Server .....	34
8.2.2	Running and Using LWM2M Server to Access the LWM2M Clients.....	35
8.2.3	LWM2M Network with Datagram Transport Layer Security (DTLS) Encryption.....	39



8.3	IoT WebDemo Server.....	40
8.3.1	Start the 6LoWPAN IoT WebDemo Server.....	40
8.3.2	Pinging a 6LoWPAN Node.....	42
8.3.3	Toggle LEDs.....	43
8.3.4	Reading Temperature.....	43
8.3.5	HTTP Server.....	44
8.3.6	Network Topology View.....	44
8.3.7	Configuration Page.....	45
8.3.8	RSSI Scanner.....	46
8.3.9	802.15.4 Sniffer.....	47
9.0	Over-The-Air (OTA) Firmware Upgrade Tool.....	49
9.1	Using the 6LoWPAN Over-the-Air Firmware Upgrade Tool.....	49
<b>Appendix A.....</b>		<b>50</b>
A.1	UART Connection.....	50

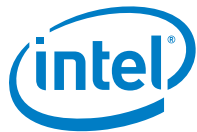
## Figures

Figure 1.	Overview of the Software Architecture on Target System.....	11
Figure 2.	IPv6 Address of the Serial Radio.....	30
Figure 3.	List of Nodes Connected to the Serial Radio.....	30
Figure 4.	CoAP GUI Interface.....	32
Figure 5.	LWM2M Protocol Stack.....	39
Figure 6.	Device List for all 6LoWPAN Nodes.....	42
Figure 7.	Example of Pinging Node using IoT WebDemo Server.....	43
Figure 8.	Example of Temperature Reading.....	43
Figure 9.	IPv6 Address of the Nodes on 6LoWPAN Network.....	44
Figure 10.	Network Topology View.....	45
Figure 11.	Configuration Page of NBR.....	46
Figure 12.	RSSI Scanner.....	47
Figure 13.	802.15.4 Sniffer.....	48
Figure 14.	UART Connection Block Diagram for Baytrail-I.....	50
Figure 15.	UART Connection Block Diagram for Apollo Lake -I.....	50



## Tables

Table 1. Terminology.....	7
Table 2. Reference Documents.....	7
Table 3. Link Layer Security Description .....	22
Table 4. Command Line Interface .....	33
Table 5. Objects Supported by IoT-U10.....	37
Table 6. Objects/Resources.....	38



## Revision History

---

Date	Revision	Description
July 2017	004	Added new platform (Intel Atom® E3900 SoC) that supports Wi-Fi and Ethernet connection between host and target.
March 2017	003	Software version 2 Release
September 2016	002	Added Section 6
August 2016	001	Initial release (Gold)



## 1.0 Introduction

This document assists in creating and integrating 6LoWPAN IoT features into Yocto\* BSP and other Linux\* platforms.

The document is intended for hardware and software engineers with experience in developing embedded applications. This document assists in setting up the gateway or network border router (NBR) to enable communication between the local/remote PC on the Ethernet network and the nodes on the 6LoWPAN network. Also covered are different methods to access the 6LoWPAN nodes with a local PC through HTTP and CoAP protocols. This document assumes some experience with IPv4 and IPv6 networking, Linux/Windows\* OSes, and using software/hardware tools to flash firmware into the hardware platform.

### 1.1 Terminology

Table 1. Terminology

Terminology Term	Description
LWM2M	Light Weight Machine to Machine
6LoWPAN	IPv6 Low-power Wireless Personal Area Network
CoAP	Constrained Application Protocol
GUI	Graphical User Interface
UART	Universal Asynchronous Receiver/ Transmitter
USB	Universal Serial Bus

### 1.1 Reference Documents

Table 2. Reference Documents

Document	Document No./Location
Intel Atom® Processor E3800 Product Family IPSO 6LoWPAN IoT Software for Yocto Project* Release Notes	334857-003
Intel RDC website for Intel® Atom® Processor E3900 Series, Intel® Celeron® Processor N3350, and Intel® Pentium® Processor N4200, Formerly Apollo Lake	<a href="https://www.intel.com/content/www/us/en/embedded/products/apollo-lake/technical-library.html">https://www.intel.com/content/www/us/en/embedded/products/apollo-lake/technical-library.html</a>



Intel RDC website for Intel® Atom™ Processor E3800 Product Family and Intel® Celeron® Processor N2807/N2930/J1900	<a href="https://www.intel.com/content/www/us/en/embedded/products/bay-trail/software-and-drivers.html">https://www.intel.com/content/www/us/en/embedded/products/bay-trail/software-and-drivers.html</a>
---	---

§





## 2.0 Before You Begin

---

1. Prepare a Host PC system with the following requirements:
  - 64-bit multi-core System
  - Running Linux\* Ubuntu\* 14.04 LTS
  - Minimum of 4GB RAM
  - A high-speed Internet connection to download third party sources
  - Minimum of 100 GB Storage
2. If your PC is behind a corporate network with proxy settings, use the guide at [https://wiki.yoctoproject.org/wiki/Working\\_Behind\\_a\\_Network\\_Proxy](https://wiki.yoctoproject.org/wiki/Working_Behind_a_Network_Proxy) to configure your proxy settings.
3. Install additional Linux\* packages into your host PC:

```
# sudo apt-get install gawk wget git-core diffstat unzip texinfo
gcc-multilib build-essential chrpath socat
```

```
# sudo apt-get install libssl1.2-dev xterm
```

```
# sudo apt-get install make xsltproc docbook-utils fop dblatex
xmlto
```

```
# sudo apt-get install autoconf automake libtool libglib2.0- dev
```

**Note:** The cmake that comes with Ubuntu 14.04 LTS has a problem building the LWM2M DTLS server. Intel recommends that you replace it with the latest version of cmake:

```
# sudo apt-get remove cmake
```

```
# wget https://cmake.org/files/v3.5/cmake-3.5.1.tar.gz
```

```
# tar -xvf cmake-3.5.1.tar.gz
```

```
# cd cmake-3.5.1
```

```
# ./bootstrap
```

```
# make
```

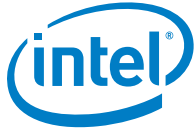
```
# sudo make install
```

Next install development toolchain and set up development environment for compiling and building firmware.

4. Download the toolchain into any directory (preferable in HOME directory)
 

```
# wget https://launchpad.net/gcc-arm-embedded/4.8/4.8-2014-q3-
update/+download/gcc-arm-none-eabi-4_8-2014q3-20140805-linux.tar.bz2
```
5. Untar the tarball
 

```
# tar xvf gcc-arm-none-eabi-4_8-2014q3-20140805-linux.tar.bz2
```



6. Include the downloaded gcc-arm-none-eabi toolchain's path in the .bashrc file

```
# gedit ~/.bashrc
```

```
export PATH=$PATH:<gcc_arm_toolchain_path>/gcc-arm-none-eabi-4_8-2014q3/bin
```

7. Source the environment

```
# source ~/.bashrc
```

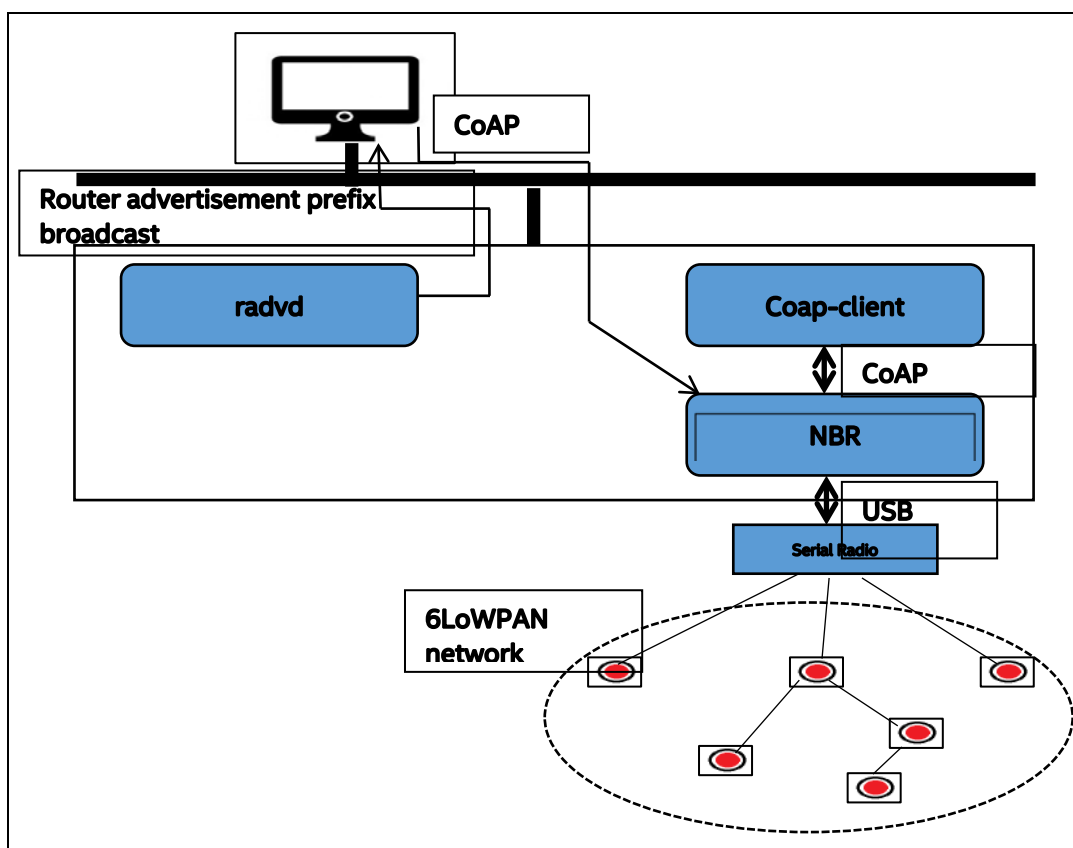
§

### 3.0 6LoWPAN IoT Software Overview

Below are the main software components and tools for 6LoWPAN support:

- Network border router (NBR)
- Serial radio firmware
- 6LoWPAN end node firmware

**Figure 1. Overview of the Software Architecture on Target System**



- Radvd and the CoAP-client tool are not part of the Native Border Router. Intel recommends that they be used together with Native Border Router.
- The NBR and the serial radio acts as the root node and forms the 6LoWPAN network with all the 6LoWPAN nodes.
- Serial radio must be attached to the system's USB port for the NBR to work. To experiment with the 6LoWPAN network, at least one 6LoWPAN node is needed.



- The system running the Yocto\* Project\* based Linux image becomes the gateway between all the IP (Internet Protocol) network connections (Wi-Fi, Ethernet, etc.). It allows other software such as the web browser and the CoAP client tool (coap-client) to access the 6LoWPAN nodes through serial radio.
- Users can also run the Linux IPv6 Router Advertisement Daemon (radvd) on the target system to broadcast the IPv6 prefix, which allows the local client to access the 6LoWPAN nodes by referring to their IPv6 address through CoAP protocol. However, radvd is not needed if users access the 6LoWPAN nodes with the IPv4 address of the HTTP server, which is running on the target system. Details on how the radvd works are out-of-scope in this document. See <http://www.litech.org/radvd/> or <https://www.ietf.org/rfc/rfc2461.txt> for more details.
- Users can access the 6LoWPAN nodes through the CoAP protocol by using the web browser on their local client. This procedure only works if the users' local client operating system supports IPv6 and the radvd is running on the target system. Firefox\* web browser can be used to access the 6LoWPAN nodes by installing Copper (Cu), a CoAP through the GUI. For advanced users, they can also access the 6LoWPAN nodes by using coap-client, which is a command line interface tool available in Linux. It is the user's responsibility to install the coap-client on their own machine (local client). Users can also use the pre-installed coap-client on the target system to access the 6LoWPAN nodes in the Linux shell.

§



## 4.0 Hardware and Software Compatibility

---

### 4.1 Hardware Supported

1. Intel Atom® Processor E3800 Product Family (Formerly Baytrail)
2. Intel Atom® Processor E3900 Series (Formerly Apollo Lake – I)
3. AzureWave AW-CB178NF for Apollo Lake I
  - Off-the-shelf M.2 Module with Wi-Fi and Bluetooth functions.
  - Software package can be downloaded from Intel RDC Website for Apollo Lake-I (refer to Section 1.2 reference table)
  - The software package must patch on top of Apollo lake-I Yocto Project\* base image.
4. Yanzi\* IoT-U10
  - Device with temperature sensor and LED that can act as Serial Radio or Node.
  - Contact your Intel representative for more details.
  - Only a USB connection is supported for this hardware.
5. Texas Instruments\* CC2538EM
  - Texas Instruments reference evaluation module that can act as Serial Radio or Node.
  - Off-the-shelf part from Texas Instruments.
  - Support USB connections.
  - Refer to Texas Instruments official website for details on the hardware.
6. Texas Instruments CC2538DK (SmartRF06 + CC2538 Evaluation Module)
  - Texas Instruments Development Kits.
  - Off-the-shelf parts from Texas Instruments.
  - Support UART connection and USB connection.
  - See the appendix for UART connection block diagrams.
7. Zolertia\* RE-MOTE
  - Off-the-shelf kit from Zolertia.
  - Dual-band (2.4GHz and onboard CC1200 for sub-GHz).
  - Support UART and USB connection.
  - Onboard RGB LED



## 4.2 Software Package

Software release package name: **6LoWPAN-3.0\_Software\_Stack\_0.8.3-2017-06-06.tar.bz2**

Inside this software package there is one folder:  
**6LoWPAN\_3.0\_Software\_Stack.**

This **6LoWPAN\_3.0\_Software\_Stack** is composed of:

1. apl-6LoWPAN-setup.sh
  - This setup script helps build a Yocto\* image for Apollo Lake I which includes the Network Border Router, IoT WebDemo Server, Over-the-Air firmware upgrade tool, LWM2M bootstrap and LWM2M server.
  - This setup script also sets up and clones repo from Sparrow project for compiling and building firmware for serial radio and end nodes.
2. byt-6LoWPAN-setup.sh
  - This setup script helps build a Yocto\* image for Baytrail which includes the Network Border Router, IoT WebDemo Server, Over-the-Air firmware upgrade tool, LWM2M bootstrap and LWM2M server.
  - This setup script also sets up and clones repo from Sparrow project for compiling and building firmware for serial radio and end nodes.
3. Meta-Wakaama
  - This is the Yocto recipe for LWM2M bootstrap and LWM2M server.
4. Meta-Netcontiki
  - Meta-Netcontiki is a Yocto recipe that provides 6LoWPAN Network Border Router (NBR).
5. Read Me
  - A quick user guide.

§



## 5.0 Build Image with 6LoWPAN Software Stack

---

If planning to use Network Border Router, there is a shell script provided to help build the complete Linux\* file system and kernel image, and bundled with the 6LoWPAN IoT Software components. Running this shell script downloads all the Linux\* components through the Internet connection and setting up ready for the subsequent build instructions.

The 6LoWPAN build package is **6LoWPAN-3.0\_SOFTWARE\_STACK\_0.8.3-2017-06-06.tar.bz2**. Set up the toolchain and Yocto BSP image before proceeding with other build and setup process.

### 5.1 Prepare Yocto BSP Image

#### 5.1.1 Build Base Image for Apollo Lake-I

To set up a 6LoWPAN3.0, user needs to set up a BSP base image. The Apollo Lake-I Yocto BSP release package can be found as below:

<https://github.com/01org/iotg-yocto-bsp-public/tree/e3900/master>

1. Git clone the Yocto BSP

```
# git clone https://github.com/01org/iotg-yocto-bsp-public.git -b e3900/master
```

2. Check out to the E3900-MR2 branch

```
# git checkout E3900-MR2
```

3. If first build; run:

```
# ./setup.sh
```

4. User is prompted, select core-image-sato to proceed.

**Note:** Time needed to build the image depend on host machine specification and Internet speed.

To enable BT/WiFi support for APL-I:

1. Put the software package same level as yocto\_build
2. Run BT/WiFi setup script

```
# ./ wifi_bt_setup.sh
```

#### 5.1.2 Build Base Image for Baytrail-I

To prepare a 6LoWPAN-3.0 for Baytrail, set up a Baytrail BSP base image. Download Yocto Project\* BSP release package for Baytrail from below:



<https://www.yoctoproject.org/downloads/bsps/jethro20/valley-island>

Steps to set up:

1. Make a new directory for Baytrail (to avoid confusion with APL-I Yocto image)

```
# mkdir byt-6lowpan
```

2. Download Poky build system into byt-6lowpan folder

```
# cd byt-6lowpan
```

```
# git clone -b jethro git://git.yoctoproject.org/poky.git
```

3. Download the corresponding BSP tarball from the BSP Downloads' page of the Yocto Project website: <https://www.yoctoproject.org/downloads/bsps/jethro20/valley-island>

Tarball: valleyisland-4.0-jethro-2.0.tar.bz2

4. Extract the corresponding tarball into meta-intel folder

```
# tar -xvjf valleyisland-4.0-jethro-2.0.tar.bz2
```

5. Rename valleyisland-4.0-jethro-2.0.tar.bz2 into meta-intel

```
# mv valleyisland-4.0-jethro-2.0.tar.bz2 meta-intel
```

6. Make a yocto\_build directory with same level as poky directory

```
# mkdir yocto_build
```

```
# cd yocto_build
```

7. Prepare build environment

```
# source ../poky/oe-init-build-env
```

8. Add the location of meta-intel to the bblayers.conf

```
# vi conf/bblayers.conf
```

```
<path>/meta-intel \
```

```
<path>/meta-intel/meta-isg/meta-valleyisland \
```

9. To build a 64-bit image, add "valleyisland-64" to MACHINE in local.conf:

```
# vi conf/local.conf
```

```
MACHINE = "valleyisland-64"
```

10. To build a 32-bit image, add "valleyisland-64" to MACHINE in local.conf:

```
# vi conf/local.conf
```





```
MACHINE = "valleyisland-32"
```

11. To build your image

```
# bitbake core-image-sato
```

**Note:** Time needed to build the image depend on host machine specification and Internet speed.

## 5.2 6LoWPAN Setup Procedure

Prerequisite:

User must build APL-I or BYT base image first before setup 6LoWPAN. Refer to Section 5.1 to prepare your Yocto Project\* image.

1. Untar 6LoWPAN software package into your directory in host machine.

```
# tar xvfj 6LoWPAN-3.0_Software_Stack_x.y.z-yyyy-mm-dd.tar.bz2
```

2. Run the shell script.

### **For Apollo Lake-I:**

a. Copy 6LoWPAN-setup folder to the same level as the APL-I *yocto\_build* and run the setup script.

```
# ./apl-6LoWPAN-setup.sh
```

b. Navigate to *yocto\_build* and source the environment

```
# cd <path>/yocto_build
```

```
# source oe-init-build-env
```

### **For Baytrail-I:**

a. Copy 6LoWPAN-setup folder inside byt-6lowpan folder

```
# cp -r 6LoWPAN-setup <path>/byt-6lowpan
```

b. Run setup script

```
# cd byt-6lowpan/6LoWPAN-setup
```

```
# ./byt-6LoWPAN-setup.sh
```

c. Navigate to *yocto\_build* and source the environment

```
# cd <path>/yocto_build
```

```
# source ../poky/oe-init-build-env
```

**Note:** These setup scripts take no parameters.



**Note:** If the setup procedure is interrupted, execute the setup script again.

It is now ready to build an image with 6LoWPAN software stack.

## 5.3 Build Yocto Project\* Image with 6LoWPAN Software Stack

A complete build may take several hours to complete, depending on the Internet connection speed and your build PC's specifications.

1. This command should be executed under the build folder /build for both Apollo Lake-I and Baytrail-I (to build whole image again).

```
# bitbake core-image-sato
```

**Note:** If the build procedure is somehow interrupted, execute the `bitbake core-image-sato` command again.

2. Final image location:

**For APL-I (64-bits only):**

```
<user_dir>/tmp/deploy/images/intel-corei7-64/
```

The output image file is

```
- core-image-sato-*.hddimg
```

**For BYT (32-bits or 64-bits):**

```
<user_dir>/tmp/deploy/images/valleyisland-32/
```

or

```
<user_dir>/tmp/deploy/images/valleyisland-64/
```

3. (Apply to Apollo Lake-I and Baytrail-I) To build only the Network Border Router application, user can run this command:

```
# bitbake native-border-router
```

4. (Apply to Apollo Lake-I and Baytrail-I) To build only the Network Border Router with *link layer security application*, user need to modify `.bb` in the build folder:

```
#cd <root setup folder>/meta-netcontiki/recipes-connectivity/router/
```

```
#vim native-border-router_1.5.0.bb
```

Add the following to make command:

```
MAKE_WITH_LLSEC_LEVEL=n (n = 0, 1, 2...7)
```

For example:

```
# To build NBR for USB SLIP radio, use this line
```



```
>-----make CC="$CC" LD="$CC" MAKE_WITH_LLSEC_LEVEL=7
```

User must to execute the build process again

#### **For Apollo Lake-I:**

```
# cd <root_setup_folder>/yocto_build
# source oe-init-build-env
```

#### **For Baytrail-I :**

```
# cd <root setup folder>/byt-6lowpan/yocto_build
# source ../poky/oe-init-build-env build/
```

(Apply to Apollo Lake-I and Baytrail-I) Compile Yocto image again.

```
# bitbake -f -c cleanall native-border-router
# bitbake native-border-router
```

(Apply to Apollo Lake-I and Baytrail-I) Application output:

- <root setup folder>/tmp/work/corei7-64-poky-linux/native-border-router/1.5.0.r0/image/opt/6lowpan/sbin
- -32- for 32 bit byt

5. After running 6LoWPAN IoT Software (in Section 5) user may need to change the `border-router.native-sparrow` (without security) into `border-router.native-sparrow_nonsec` to prevent any conflict.

```
# cd /opt/6lowpan/sbin
# mv border-router.native-sparrow border-router.native-sparrow_nonsec
```

6. Transfer the security link layer NBR from your Linux machine to BYT and APL.

```
# scp <user>@<your ip address>:<file source> <file destination>
```

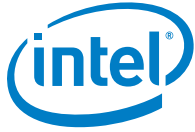
Example:

```
#scp user@10.22.227.48: tmp/work/corei7-64-poky-linux/native-border-router/1.5.0.r0/image/opt/6lowpan/sbin/border-router.native-sparrow /opt/6lowpan/sbin
```

## 5.4 Prepare SD Card or USB as Bootable Media

To write the bootable image to SD card or USB stick (assuming it is at `/dev/sdc`):

**Note:** Refer to Section 5.3 Build Procedure for Yocto Image Output directory



```
# sudo dd if=core-image-sato-*.hddimg of=/dev/sdc  
  
# sync  
  
# eject /dev/sdc
```

§



## 6.0 Building the Firmware for Serial Radio and Node

This local repository of firmware is generated after running the setup script as described in Section 5.0. After the setup is completed, navigate to the firmware directory to build firmware.

```
# cd <your root setup folder>/repo/sparrow
```

### 6.1 Build the Firmware for Node

#### 1. For **CC2538EM** board:

- To generate rescue-image, run the following commands:
 

```
# cd products/sparrow-dual-mode
# make clean TARGET=felicia BOARD=cc2538em
# make TARGET=felicia BOARD=cc2538em IMAGE=1
# make TARGET=felicia BOARD=cc2538em rescue-image
```

**Note:** Firmware for CC2538EM only supports automatic dual mode operation. The board operates as serial radio if it is attached to USB port with ttyACM USB ACM driver, otherwise it operates as a 6LoWPAN end node. It does not support CoAP and LWM2M due to lack of hardware sensors and LEDs.

- To generate .jar file extension for OTA upgrade tool, run the following commands:
 

```
#make clean TARGET+Felicia BOARD = cc2538em
#make TARGET=felicia BOARD=cc2538em MAKE_WITH_NETSCAN=1
images INC=0
```

#### 2. For **IoT-u10**:

```
# cd examples/felicia/iot-u10
# make clean
# make MAKE_WITH_NETSCAN=1 MAKE_WITH_WEBSERVER=1 MAKE_WITH_IPSO=1
images INC=0
```

**Note:** This step is used for generating firmware image with .jar file extension for OTA upgrade tool as described in Section 9. Take note that no rescue image is needed for IoT U-10 as the device comes with stock firmware.

#### 3. For **Zolertia RE-MOTE with dual-band**:

- To generate rescue-image, run the following commands:
 

```
# cd examples/zoul/remote
# make clean
```



To support sub-GHz:

```
# make MAKE_WITH_WEBSERVER=1 MAKE_WITH_IPSO=1 MAKE_WITH_920=1  
IMAGE=1
```

To support 2.4GHz:

```
# make MAKE_WITH_WEBSERVER=1 MAKE_WITH_IPSO=1
```

Finally, generate the binary for flashing.

```
# make rescue-image
```

- To generate node firmware with .jar file extension for flashing the Zolertia REMOTE with OTA firmware upgrade tool:

```
# cd examples/zoul/remote  
# make clean
```

To support sub-GHz:

```
# make MAKE_WITH_WEBSERVER=1 MAKE_WITH_IPSO=1 MAKE_WITH_920=1  
images INC=0
```

To support 2.4GHz:

```
# make MAKE_WITH_WEBSERVER=1 MAKE_WITH_IPSO=1 images INC=0
```

4. In order to build the firmware with link layer security, user needs to add `MAKE_WITH_LLSEC_LEVEL=n` compilation option to make command.

where  $n=0,1,2,...7$  and 7 is the highest level of security.

**Table 3. Link Layer Security Description**

Security level	Description	Mode
0	No security	None
1	Authentication only	AES-CBC-MAC-32
2	Authentication only	AES-CBC-MAC-64
3	Authentication only	AES-CBC-MAC-128
4	Encryption only	AES CTR
5	Encryption and authentication	AES-CCM-32
6	Encryption and authentication	AES-CCM-64
7	Encryption and authentication	AES-CCM-128

For example:

```
# make MAKE_WITH_WEBSERVER=1 MAKE_WITH_IPSO=1 MAKE_WITH_920=1  
IMAGE=1 MAKE_WITH_LLSEC_LEVEL=7
```



## 6.2 Build the Firmware for Serial Radio

### 1. For **CC2538EM**:

```
# cd products/sparrow-serial-radio
# make clean
# make TARGET=felicia BOARD=cc2538em IMAGE=1
# make TARGET=felicia BOARD=cc2538em rescue-image
```

To support UART with hardware flow control at 912.6kbps for CC2538EM:

```
# cd products/sparrow-serial-radio
# make clean
# make TARGET=felicia BOARD=cc2538em MAKE_WITH_UART_FLOW_CONTROL=1
IMAGE=1
# make TARGET=felicia BOARD=cc2538em rescue-image
```

### 2. For **IoT-u10**:

```
# cd products/sparrow-serial-radio
# make TARGET=felicia BOARD=iot-u10 clean
# make TARGET=felicia BOARD=iot-u10 images INC=0
```

**Note:** This step is used for generating firmware image with .jar file extension for OTA upgrade tool as described in Section 9.

### 3. For **Zolertia RE-MOTE with dual-band**:

- To generate rescue-image, run the following commands:

```
# cd products/sparrow-serial-radio
# make TARGET=zoul-sparrow BOARD=remote clean
# make TARGET=zoul-sparrow BOARD=remote MAKE_WITH_920=1
USE_DEBUG_PORT=1 IMAGE=1
# make TARGET=zoul-sparrow BOARD=remote rescue-image
```

To support SubGHz:

```
# cd products/sparrow-serial-radio

# make TARGET=zoul-sparrow BOARD=remote-revb clean

# make TARGET=zoul-sparrow BOARD=remote-revb MAKE_WITH_920=1
USE_DEBUG_PORT=1 images INC=0
```

To support 2.4GHz:

```
# cd products/sparrow-serial-radio

# make TARGET=zoul-sparrow BOARD=remote-revb clean

# make TARGET=zoul-sparrow BOARD=remote-revb USE_DEBUG_PORT=1
IMAGE=1

# make TARGET=zoul-sparrow BOARD=remote-revb rescue-image
```



- To generate serial radio firmware with .jar file extension for flashing the Zolertia RE-MOTE with OTA firmware upgrade tool:

To Support sub-GHz:

```
# cd products/sparrow-serial-radio
# make TARGET=zoul-sparrow BOARD=remote clean
# make TARGET=zoul-sparrow BOARD=remote MAKE_WITH_920=1
USE_DEBUG_PORT=1 images INC=0
```

To Support 2.4GHz:

```
# cd products/sparrow-serial-radio
# make TARGET=zoul-sparrow BOARD=remote-revb clean
# make TARGET=zoul-sparrow BOARD=remote-revb USE_DEBUG_PORT=1
images INC=0
```

If reflashing the node with newer firmware using OTA firmware upgrade tool in the same day, rebuild the firmware with higher **INC** number. For example, if building the previous firmware with **INC=0**, use any number higher than 0 when building new firmware. Intel recommends incrementing the **INC** number by 1 at a time.

4. Make sure to specify **MAKE\_WITH\_IPSO=1** and **MAKE\_WITH\_DTLS=1** in make when building firmware sensor for node to work as LWM2M client with DTLS support.

### 6.2.1 Flashing Firmware into Hardware Device

1. CC2538EM rescue-image is rescue-cc2538em.bin. Copy rescue-image.bin to your Windows machine and flash it to your CC2538EM board.
2. Refer to the Texas Instruments website for detailed guidelines to flash the CC2538EM node using the flash upgrade tool.
3. For flashing .jar firmware binary into the node or serial radio with OTA upgrade tool refer to [Section 9.0](#).
4. For Zolertia RE-MOTE rescue-image is rescue-remote.bin. For setting up and using flashing tool to flash Zolertia RE-MOTE. Refer to following:
5. Before using the cc2538-bsl tool for the first time, make sure to check the availability of cc2538-bsl tool with following instructions:

```
# cd <root setup folder>/contiki/tools/cc2538-bsl/
# ls
```

6. If the folder is empty means the cc2538-bsl tool is not presented, do the following:

```
# cd ../../
```





```
# git submodule init
# git submodule update
```

7. The cc2538-bsl is cloned from open-source project's repository and ready for use.
8. Flash the image using cc2538-bsl tool.

```
# cd <root setup folder>/contiki/tools/cc2538-bsl/
For serial radio:
# python cc2538-bsl.py -e -w -v -p /dev/ttyUSB0 <root setup
folder>/products/sparrow-serial-radio/rescue-remote.bin
For Node:
# python cc2538-bsl.py -e -w -v -p /dev/ttyUSB0 <root setup
folder>/examples/zoul/remote/rescue-remote.bin
```

## §



## 7.0 Running the 6LoWPAN IoT Software

---

This section provides step-by-step instruction on running the 6LoWPAN software on the target system platform. Start the target platform with the image created in Section 5.2.1 and log on the target through an SSH (Ethernet port) or serial console (UART port).

### 7.1 Preparation

What you need:

- A host PC as local/remote client running: Ubuntu\* Linux with IPv6 support is recommended.
- An Intel Architecture target PC.
- A web browser installed on the host PC. A Firefox\* browser is recommended.
- An Ethernet network with a DHCP server. This network connects the local client and target system through the LAN. Connection to the Internet and remote client are optional.
- A 6LoWPAN serial radio device.
- One or more 6LoWPAN node devices.

### 7.2 Setting up Network Configuration

#### 7.2.1 Set up WiFi/LAN Connection with Static IPv6 for Apollo Lake-I

1. To start Wi-Fi connection for Apollo Lake I platform

```
# rfkill unblock Wi-Fi  
  
# ifconfig <WIRELESS_NETWORK_INTERFACE> up
```

2. Edit /etc/wpa\_supplicant.conf, with the following:

```
# vi /etc/wpa_supplicant.conf  
  
ctrl_interface=/var/run/wpa_supplicant  
  
ctrl_interface_group=0  
  
Network={  
    ssid="<ROUTER_SSID>"  
  
    key_mgmt=NONE  
  
    scan_ssid=1  
}
```



3. Up the wireless interface with wpa\_supplicant:

```
# wpa_supplicant -B -i<WIRELESS_NETWORK_INTERFACE> -c
/etc/wpa_supplicant.conf

# wpa_supplicant -B -iwlp2s0 -c /etc/wpa_supplicant.conf
```

4. Check the status: "wpa\_state=COMPLETED"

```
# wpa_cli -p /var/run/wpa_supplicant status
```

5. Start dhcp to obtain ip address:

```
# udhcp -i <WIRELESS_NETWORK_INTERFACE>

# udhcp -i wlp2s0
```

6. Check interface for IP address:

```
# ifconfig
```

7. (Both LAN and Wi-Fi) To add Static IPv6 address to the APL-I (e.g, LAN – elp1s0; Wi-Fi – wlp2s0)

```
# ip -6 addr add <IPv6_ADDRESS>/<NETMASK> dev <NETWORK_INTERFACE>

# ip -6 addr add bbbb::6a05:caff:fe1c:12d1/64 dev wlp2s0
```

## 7.2.2 Set up LAN Connection with Static IPv6 for Baytrail-I

To add Static IPv6 address to the Baytrail, edit the /etc/network/interfaces as below:

```
# vi /etc/network/interfaces

pre-up modprobe ipv6
iface eth0 inet6 static
address bbbb::6a05:caff:fe1c:12d1
netmask 64
```

## 7.3 Starting Network Border Router

1. Attach the serial radio to the USB port on the target system. You should be able to see ttyACM0 in the /dev directory. Enter the following command to confirm the serial radio is detected by the OS:

### **For IoT-u10 and CC2538EM:**

```
# ls /dev/ttyA*

/dev/ttyACM0
```

**For CC2538EM using UART port**, /dev/ttySn is used. (Replace *n* with number depending on your UART hardware configurations)

### **For Zolertia RE-MOTE:**

```
# ls /dev/ttyUSB*
```

```
/dev/ttyUSB0
```

2. Attach the USB port label 'DBG' to host machine for the device to work as a serial radio.
3. If the serial radio is present, then you can start the NBR. The NBR executable binary is located at `/opt/6lowpan/sbin`. Enter the command as follows:

```
# cd /opt/6lowpan/sbin
```

- a. **For USB**, enter the command:

```
# ./border-router.native-sparrow -s /dev/tty* fd02::1/64
```

- b. **For UART**, enter the command:

With hardware flow control firmware:

```
# ./border-router.native-sparrow -B 460800 -H -s /dev/tty* fd02::1/64
```

Without hardware flow control firmware:

```
# ./border-router.native-sparrow -B 460800 -s /dev/tty* fd02::1/64
```

**Note:** User needs to change /dev/tty\* above to /dev/ttyACM0 for **IoT-U10 and CC2538EM** or /dev/ttyUSB0 for **Zolertia RE-MOTE**.

4. A tunnel interface tun0 with IPv6 address fd02::1 is created by NBR as shown below:

```
# ifconfig
```

```
tun0    Link encap:UNSPEC HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
```

```
inet6 addr: fd02::1/64 Scope:Global
```

UP POINTOPOINT RUNNING NOARP MULTICAST MTU:1500 Metric:1

RX packets:0 errors:0 dropped:0 overruns:0 frame:0

TX packets:0 errors:0 dropped:0 overruns:0 carrier:0

```
collisions:0 txqueuelen:500
```

RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)



**Note:** The NBR needs to run continuously. Users are advised to run other applications in new terminals to avoid disrupting the NBR. Please kill the NBR (CTRL+C) before dismantling the device connection.

5. The NBR is up and running.

## 7.4 Using Static IPv6 Method

1. Users must set IPv6 as a static address, as mentioned in Section 7.2. If static IPv6 address is successfully configured, user will see this as below:

```
# ifconfig Eth0

Eth0  Link encap: Ethernet HWaddr 98:4F:EE:01:6E:30

      inet addr:192.168.0.100 Bcast:172.30.66.255 Mask:255.255.255.0

      inet6 addr: bbbb::6a05:caff:fe1c:12d1/64 Scope:Global

      inet6 addr: fe80::984f:eeff:fe01:6e30/64 Scope:Link

      UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

      RX packets:37083 errors:0 dropped:0 overruns:0 frame:0

      TX packets:3907 errors:0 dropped:0 overruns:0 carrier:0

      collisions:0 txqueuelen:1000

      RX bytes:3118774 (2.9 MiB) TX bytes:554533 (541.5 KiB)

      Interrupt:41 Base address:0x8000
```

2. Next, to access 6LoWPAN network via target system (BYT/APL), user required to add an IPv6 route through a gateway from local client PC.

**Note:** The following command is needed in your client PC

```
# ip -6 route add default via <GATEWAY_ADDRESS> dev <NETWORK_INTERFACE>

# ip -6 route add default via bbbb::6a05:caff:fe1c:12d1 dev eth0
```

3. To enable routing in IPv6, issue the following command:

```
# sysctl -w net.ipv6.conf.all.forwarding=1
```

Now the target system acts as an IPv6 router, which enables the local client to access the 6LoWPAN nodes with IPv6 addresses.

## 7.5 Checking the Nodes Registered to Serial Radio

There are several ways to identify the nodes that are joining the NBR. This method requires the http web address obtained when the NBR was started.

Example:

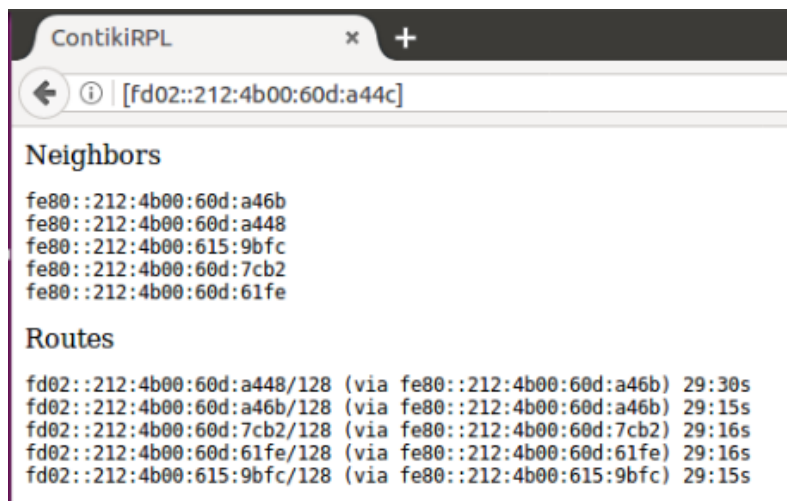
IPv6 Address of NBR is fd02::212::4b00::60d:a44c

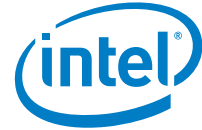
**Figure 2. IPv6 Address of the Serial Radio**

```
Server IPv6 addresses:
0x6bca80: => fd02::212:4b00:60d:a44c
0x6bcaa8: => fe80::212:4b00:60d:a44c
```

When the NBR is started, user is able to access the webpage via browser from end PC. The webpage displays the node that is connected to network border router. Use the IPv6 address of the NBR ([http://\[IPv6 Address of NBR\]/r](http://[IPv6 Address of NBR]/r) at port 80).

**Figure 3. List of Nodes Connected to the Serial Radio**





## 8.0 Accessing the 6LoWPAN IoT Software

---

This document covers two methods to access the 6LoWPAN Network:

- CoAP protocol between CoAP server and CoAP client
- IoT Webdemo Server

### 8.1 Access 6LoWPAN Network with CoAP Client

This section focuses on accessing the 6LoWPAN nodes directly with IPv6 addresses through CoAP protocol. Every node in 6LoWPAN is a CoAP server. The user accesses the nodes as a CoAP client either through a Firefox Web Browser GUI or command line.

#### 8.1.1 Firefox Web Browser GUI CoAP Client

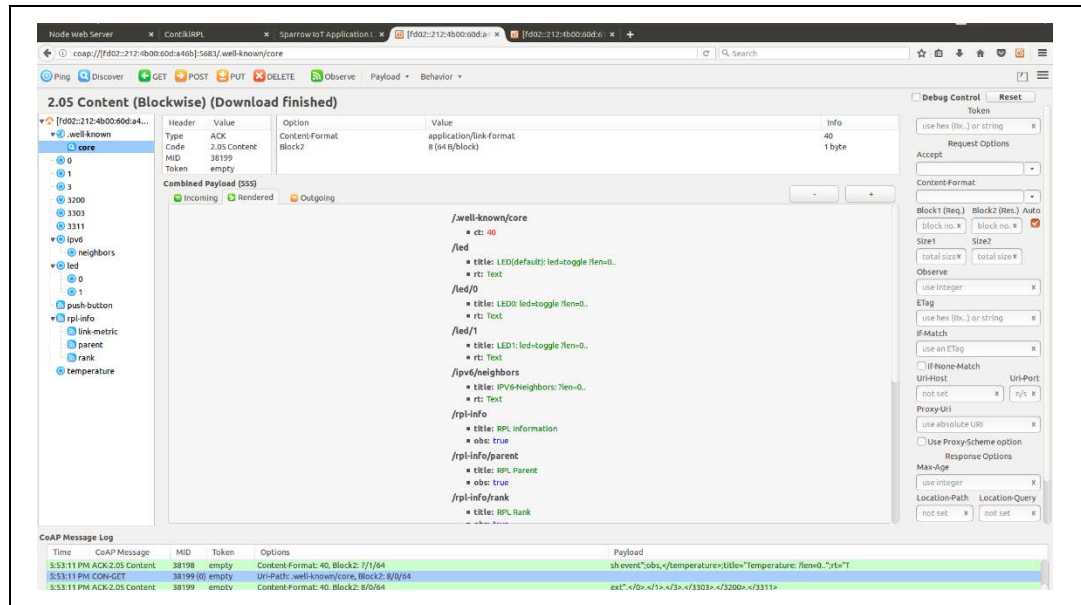
1. Use Firefox web browser and go to <https://addons.mozilla.org/en-US/firefox/addon/copper-270430/> to install the CoAP plug-in Copper (Cu) into it.
2. Ensure the plugin is installed properly and enabled in Firefox web browser.
3. Find out the IPv6 address of the 6LoWPAN node to access. For example, if the 6LoWPAN node's IPv6 address is fd02::212:4b00:60d:a46b enter the following CoAP URI into the Firefox address bar as below:

**coap://[fd02::212:4b00:60d:a46b]/.well-known/core**

A CoAP GUI interface is shown as seen in the screenshot below.



Figure 4. CoAP GUI Interface



Use the buttons **GET**, **POST**, **PUT**, **DELETE** to perform an action on the resources listed in the left pane on the webpage. For example, at a resource location (`coap://[fd02::212:4b00:60d:a46b]:5683/temperature`), use the **GET** button to read the temperature. The temperature reading is displayed in the browser. To toggle the LED on the 6LoWPAN node, click on the link "0" under the "led" in the left pane. You are redirected to URI `coap://[fd02::212:4b00:60d:a46b]:5683/led/0`. Click the **POST** button to toggle the state of LED. The state of LED is displayed in the browser.

## 8.1.2 Command-line-Interface CoAP Client

For users who prefer command-line interface, use the **coap-client** pre-installed in the target system. Below are examples using the command-line interface to access the 6LoWPAN node.





Table 4. Command Line Interface

CoAP Resource Name	Method	Command
/well-known/core	GET	<pre>root@valleyisland-64:~# coap-client -m get coap://[fd02::212:4b00:60d:a46b]/well-known/core v:1 t:0 tkl:0 c:1 id:49657 &lt;/well-known/core&gt;;ct=40,&lt;/led&gt;;title="LED(default): led=toggle ?len=0..";rt="Text",&lt;/led/0&gt;;title="LED0: led=toggle ?len=0..";rt="Text",&lt;/led/1&gt;;title="LED1: led=toggle ?len=0..";rt="Text",&lt;/ipv6/neighbors&gt;;title="IPV6- Neighbors: ?len=0..";rt="Text",&lt;/rplv:1 t:0 tkl:0 c:1 id:49658 l-info&gt;;title="RPL Information";obs,&lt;/rpl-info/parent&gt;;title="RPL Parent";obs,&lt;/rpl-info/rank&gt;;title="RPL Rank";obs,&lt;/rpl-info/link- metric&gt;;title="RPL Link Metric";obs,&lt;/push-button&gt;;title="Push event";obs,&lt;/temperature&gt;;title="Temperature: ?len=0..";rt="Tv:1 t:0 tkl:0 c:1 id:49659 ext"</pre>
/temperature (only applicable for IoT-U10 and RE-MOTE)	GET	<pre>root@valleyisland-64:~# coap-client -m get coap://[fd02::212:4b00:60d:a46b]/temperature v:1 t:0 tkl:0 c:1 id:19162 Temperature (C): 28.5</pre>
/led/0 (only applicable for IoT-U10 and RE-MOTE)	POST	<pre>root@valleyisland-64:~# coap-client -m post -e led=toggle coap://[fd02::212:4b00:60d:a46b]/led/0 v:1 t:0 tkl:0 c:2 id:46220 LED0 Toggle: ON</pre>

## 8.2 Lightweight Machine to Machine (LWM2M)

Lightweight Machine to Machine (LWM2M) is a protocol from the Open Mobile Alliance for M2M or IoT device management. It is a new, still on-going effort to create a new technical standard for remote management of machine-to-machine devices, service enablement, and application management. For more detailed information about how this protocol works, obtain specification documents from the URL below:

<http://openmobilealliance.hs-sites.com/lightweight-m2m-specification-from-oma>

To manage 6LoWPAN nodes using this protocol, the nodes should support the LWM2M protocol. The nodes act as LWM2M clients and communicate with the LWM2M server running on the target system. Commands and queries can be sent to the nodes through the LWM2M server's command prompt interface. The following sections describe the required steps to configure the LWM2M clients and manage them through the LWM2M server.



## 8.2.1 Configuring and Running LWM2M Bootstrap Server

The LWM2M (Wakaama) Bootstrap Server provides a **Bootstrap Interface** for provisioning the essential information into the LWM2M clients to enable the LWM2M clients to register themselves with one or more LWM2M servers.

The LWM2M Bootstrap Server's executable binary (**bootstrap\_server**) and its configuration file (**bootstrap\_server.ini**) can be found under directory `/opt/6lowpan/sbin/`.

User needs to open a terminal to operate the Bootstrap and LWM2M. Assuming the user has opened a new terminal at the target system.

1. Navigate the terminal to the directories. # `cd /opt/6lowpan/sbin`
2. User needs to change the all the prefix from **aaaa** to **fd02** in `bootstrap_server.ini` using the terminal. Open the `bootstrap_server.ini`.  
# `vi bootstrap_server.ini`

```
uri=coap://[aaaa::1]:5683 to uri=coap://[fd02::1]:5683
uri=coaps://[aaaa::1]:5684 to uri=coaps://[fd02::1]:5684
```

3. To start the LWM2M Bootstrap Server, perform the following:  
# `./bootstrap_server -f bootstrap_server.ini`  
Messages are displayed in the console when provisioning happens between the LWM2M clients and the LWM2M Bootstrap server. The LWM2M clients are ready to connect to LWM2M server after successfully configured by the LWM2M Bootstrap Server.

Here is an example of **bootstrap\_server.ini**:

```
# Standard non-encrypted LWM2M server

[Server]

id=1
uri=coap://[fd02::1]:5683
bootstrap=no
lifetime=300
security=NoSec

# DTLS encrypted LWM2M server with Pre-Shared key
# using identity 'OurIdentity' and password 'OurSecret'.
# The identity and password are specified in hexadecimal form.

[Server]

id=2
uri=coaps://[fd02::1]:5684
bootstrap=no
lifetime=300
```



```

security=PSK
public=4f75724964656e74697479
secret=4F7572536563726574

# For any client, we delete all server accounts and
# provision all the server info.

[Endpoint]

Delete=/0
Delete=/1
Server=1

```

The settings above provision the LWM2M clients to connect to the first entry of the LWM2M server listed in the configuration file, which is a non-DTLS encrypted LWM2M server with URI (coap://[fd02::1]:5683). This is the ipv6 address of the target system.

## 8.2.2 Running and Using LWM2M Server to Access the LWM2M Clients

1. Start the LWM2M Server in other terminal

```

# cd /opt/6lowpan/sbin
# ./lwm2mserver

```

Information as shown below is displayed on screen automatically when one or more LWM2M client has successfully registered itself to the LWM2M server:

```

> 95 bytes received from [fd02::212:4b00:60d:61fe]:5683
40 02 15 96 B2 72 64 4D 10 65 70 3D 5A 6F 6C 65
@....rdM.ep=Zole
72 74 69 61 20 52 45 2D 4D 2D 34 42 30 30 30 36 rtia RE-M-
4B0006
30 44 36 32 33 39 FF 3C 30 2F 31 3E 2C 3C 33 2F
0D61fe.<0/1>,<3/
30 3E 2C 3C 33 33 30 33 2F 30 3E 2C 3C 33 32 30
0>,<3303/0>,<320
30 2F 30 3E 2C 3C 33 33 31 31 2F 30 3E 2C 3C 33
0/0>,<3311/0>,<3
33 31 31 2F 31 3E 2C 3C 33 33 31 31 2F 32 3E 311/1>,<3311/2>

```

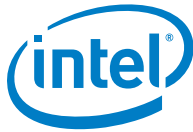
New client #0 registered.

Client #0:

```

name: "Zolertia RE-M-4B00060D61fe"
binding: "Not specified"
lifetime: 86400 sec

```



```
objects: /0/1, /3/0, /3200/0, /3303/0, /3311/0, /3311/1,
/3311/2,
```

```
> 80 bytes received from [fd02::212:4b00:60d:a46b]:5683
40 02 DE 1B B2 72 64 4D 0A 65 70 3D 49 6F 54 2D
@....rdM.ep=IoT-
55 31 30 2D 34 42 30 30 30 36 30 44 41 34 35 44 U10-
4B00060DA46B
FF 3C 30 2F 31 3E 2C 3C 33 2F 30 3E 2C 3C 33 33
.<0/1>,<3/0>,<33
30 33 2F 30 3E 2C 3C 33 32 30 30 2F 30 3E 2C 3C
03/0>,<3200/0>,<
33 33 31 31 2F 30 3E 2C 3C 33 33 31 31 2F 31 3E
3311/0>,<3311/1>
```

New client #1 registered.

Client #1:

```
name: "IoT-U10-4B00060DA46B"
binding: "Not specified"
lifetime: 86400 sec
objects: /0/1, /3/0, /3200/0, /3303/0, /3311/0, /3311/1,
```

Depending on the type of services or functions supported by the LWM2M clients, different object information may be shown.

2. Enter **"help"** command in the LWM2M server's command prompt to get a list of commands supported by the server as shown below:

```
> help
help      Type 'help [COMMAND]' for more details on a command.
list      List registered clients.
read      Read from a client.
disc      Discover resources of a client.
write     Write to a client.
time      Write time-related attributes to a client.
attr      Write value-related attributes to a client.
clear     Clear attributes of a client.
exec      Execute a client resource.
del       Delete a client Object instance.
create    create an Object instance.
observe   Observe from a client.
cancel    Cancel an observe.
q         Quit the server.
```



3. Now, enter “**list**” command to get a list of LWM2M clients registered to the server.

```
> list
```

```
Client #0:
```

```
    name: "Zolertia RE-M-4B00060D61fe"
```

```
    binding: "Not specified"
```

```
    lifetime: 86400 sec
```

```
    objects: /0/1, /3/0, /3200/0, /3303/0, /3311/0, /3311/1,
             /3311/2,
```

```
Client #1:
```

```
    name: "IoT-U10-4B00060DA46B"
```

```
    binding: "Not specified"
```

```
    lifetime: 86400 sec
```

```
    objects: /0/1, /3/0, /3200/0, /3303/0, /3311/0, /3311/1,
```

From the example above, there are only two LWM2M clients registered to the server and the clients' names are “**Zolertia RE-M-4B00060D61FE**” and “**IoT-U10-4B00060DA46B**”. The ID of the client is **0** and **1** respectively. There are seven IPSO objects reported by the LWM2M client for Zolertia RE-MOTE and only six for IoT-U10.

IPSO objects are presented with **/Object ID/ Object Instance**.

The following objects are supported by IoT-U10 and Zolertia RE-MOTE.

**Table 5. Objects Supported by IoT-U10**

/0/1	LWM2M Security	Instance 1	
/3/0	Device	Instance 0	
/3200/0	Digital Input	Instance 0	e.g., switch/button
/3303/0	Temperature Sensor	Instance 0	
/3311/0	Light Control	Instance 0	e.g., LED/light bulb
/3311/1	Light Control	Instance 1	e.g., LED/light bulb
/3311/2 (Only supported for Zolertia RE-MOTE)	Light Control	Instance 2	e.g., LED/light bulb

**Object** is a collection of **Resources**. A **Resource** is an atomic piece of information or interface that can be accessed or manipulated (for example, Read, Written or Executed) by user. Objects/Resources can be accessed with simple URI **/Object ID/ Object Instance/ Resource ID**, as in the following table.



Table 6. Objects/Resources

URI	Object ID	Instance ID	Resource ID
/3200/0/5500	Digital Input	Instance 0	Digital Input State
/3303/0/5700	Temperature Sensor	Instance 0	Sensor Value
/3311/0/5850	Light Control	Instance 0	On/Off

For more information about the definition of Object and Resource, refer to the following URL:

<http://www.openmobilealliance.org/wp/OMNA/LwM2M/LwM2MRegistry.html>

Below are some of the basic examples on how the LWM2M client is accessed through the URIs as shown above.

1. **To read the state of input button** of the LWM2M client (Client ID 0), enter the following command in LWM2M (Wakaama) server's command prompt:

```
> read 0 /3200/0/5500
OK
> 13 bytes received from [fd02::212:4b00:60d:61fe]:5683
64 45 78 39 39 78 C6 EF C2 06 05 FF 30 dEx99x.....0
```

```
Client #0 /3200/0/5500 : 2.05 (COAP_205_CONTENT)
1 bytes received of type text/plain:
30 0
```

2. **To get the reading of temperature sensor** of the LWM2M client:

```
> read 0 /3303/0/5700
OK
> 17 bytes received from [fd02::212:4b00:60d:61fe]:5683
64 45 78 3A 3A 78 6F F1 C2 06 05 FF 32 39 2E 32
dEx::xo.....29.2
38 8
```

```
Client #0 /3303/0/5700 : 2.05 (COAP_205_CONTENT)
5 bytes received of type text/plain:
32 39 2E 32 38 29.28
```

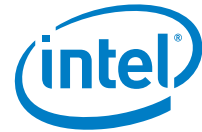
3. **To turn on the first LED** of the LWM2M client:

```
> write 0 /3311/0/5850 1
OK
> 8 bytes received from [fd02::212:4b00:60d:61fe]:5683
64 44 78 3D 3D 78 FE F1 dDx==x..
```

```
Client #0 /3311/0/5850 : 2.04 (COAP_204_CHANGED)
```

4. **To turn off the first LED** of the LWM2M client:

```
> write 0 /3311/0/5850 0
```



```
OK
> 8 bytes received from [fd02::212:4b00:60d:61fe]:5683
64 44 78 41 41 78 C2 F2 dDxAx..
```

```
Client #0 /3311/0/5850 : 2.04 (COAP_204_CHANGED)
```

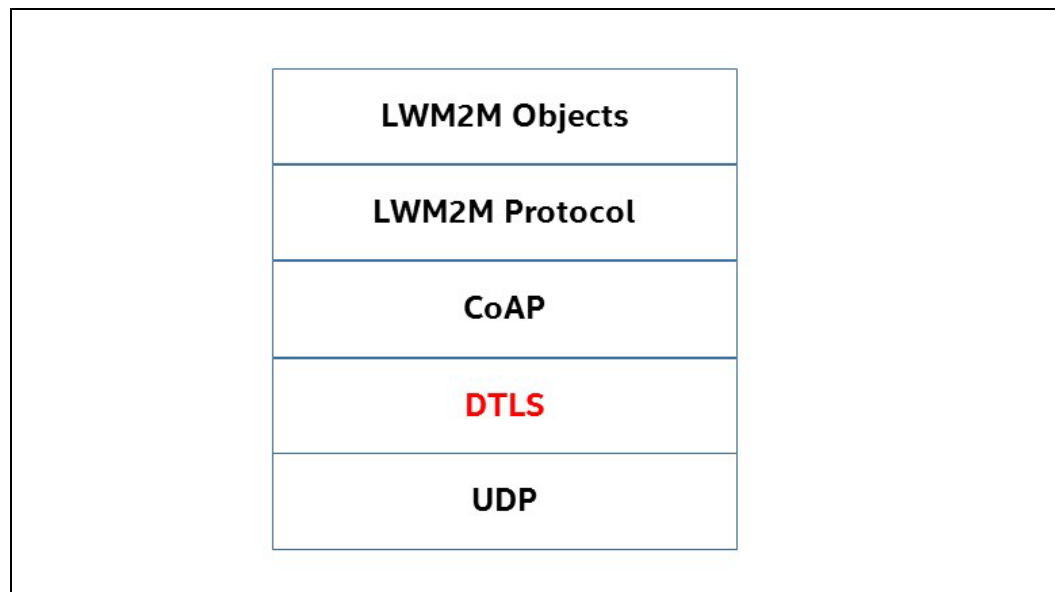
For more advance operations on LWM2M object's resources, refer to the LWM2M specification document.

**Note:** All data received from LWM2M client are displayed in ASCII format.

### 8.2.3 LWM2M Network with Datagram Transport Layer Security (DTLS) Encryption

**Datagram Transport Layer Security (DTLS)** is a protocol for securing network transmission for datagram-oriented transport such as UDP. It provides security for data interchanged between the LWM2M server and LWM2M client. Refer to Figure 5 below.

**Figure 5. LWM2M Protocol Stack**



DTLS can be enabled easily by changing the configuration file of LWM2M (Wakaama) Bootstrap Server. Just change the settings under **Endpoint** in **bootstrap\_server.ini** as below:

```
[Endpoint]
Delete=/0
Delete=/1
Server=2
```



The settings above provision the LWM2M clients to connect to LWM2M server with id = Refer to [Section 8.2.1](#) the 2nd entry of the server in **bootstrap\_server.ini** are configured using the DTLS Pre-Shared Key security.

Start the LWM2M Bootstrap Server:

```
# cd /opt/6lowpan/sbin
# ./bootstrap_server -f bootstrap_server.ini
```

Then, start the LWM2M (Wakaama) server with DTLS:

```
# cd /opt/6lowpan/sbin
# ./lwm2mserver-dtls
```

The LWM2M clients should connect to the LWM2M server with secure encryptions and available for access through the LWM2M server as described [Section 8.2.2](#).

**Note:** For more information about enhance the 6LoWPAN network security, refer to 6LoWPAN IoT Security Guide.

## 8.3 IoT WebDemo Server

This section describes the use of the 6LoWPAN IoT WebDemo Server. This is a sample application for the user to be able to access 6LoWPAN nodes.

There are known security risks when using this tool under certain system configurations. The user owns the responsibility of ensuring that system security is not compromised by operating this tool.

Before beginning:

- Target PC running 6LoWPAN IoT Software.
- PC is connected in LAN/WLAN configuration.
- Extranet connection is optional.

### 8.3.1 Start the 6LoWPAN IoT WebDemo Server

1. Open a new terminal while running the NBR.
2. Go to this directory.  

```
# cd /opt/6lowpan/sbin/examples/sparrow
```
3. Start the 6LoWPAN Demo Server (wsdemoserver.py) to allow the local client and remote client to access the 6LoWPAN network through HTTP protocol with IPv4 address.  

```
# python wsdemoserver.py
```





The above command starts the 6LoWPAN Demo Server as a background process. Now the local client and remote client are able to access the HTTP server through port 8000.

For Zolertia RE-MOTE running sub-GHz, start the web demo with default channel 0 as below:

```
# python wsdemoserver.py -c 0
```

4. After the 6LoWPAN Demo Server has started, you can browse to the IP address of the target system using any modern browser (with support for HTML5 and JavaScript).



- The default port is 8000, so the link to the index page is (assuming target system IP address is 192.168.0.100) the following:

<http://192.168.0.100:8000/index.html>

The 6LoWPAN Demo page contains a device list where all the 6LoWPAN nodes can be seen and all the basic function can be demonstrated by clicking the buttons. The device list is shown in the screenshot below.

Figure 6. Device List for all 6LoWPAN Nodes

The screenshot shows a web browser window with the URL 192.168.0.55:8000/index.html. The page title is "Sparrow IoT Application". Below the title, there are tabs for "6LoWPAN Devices", "Network Topology", "RF and Serial Radio", and "About". The "6LoWPAN Devices" tab is active, showing a table of discovered devices. The table has columns for IPv6 Address, Last Seen, Rank, Type, Actions, and Msg. There are four rows of data, each representing a different 6LoWPAN node. The "Actions" column contains buttons for "Ping node", "Led 1 Toggle", "Led 2 Toggle", and "Read Temp". The "Msg" column shows the last received message for each node.

IPv6 Address	Last Seen	Rank	Type	Actions	Msg
fd02:212:4b00:60d:a46b (via server)	27	2	IoT-U10	Ping node, Led 1 Toggle, Led 2 Toggle, Read Temp	Temperature: 34.69 C
fd02:212:4b00:60d:a46b (via server)	28	3	IoT-U10	Ping node, Led 1 Toggle, Led 2 Toggle, Read Temp	Temperature: 32.06 C
fd02:212:4b00:60d:631b (via server)	27	2	Zolertia RE-Mote revision A	Ping node, Led 1 Toggle, Led 2 Toggle, Read Temp	Temperature: 30.71 C
fd02:212:4b00:60d:7db2 (via server)	29	2	Sparrow Dual-Op	Ping node	

Last Message: DBG: SENT: 8/16/17 10:02:21:2:4b00:60d:a46b

In the screenshot above, you can see there are two 6LoWPAN nodes discovered by the NBR. Users can click the **"Update list"** button to refresh the device list if the 6LoWPAN nodes are not listed on the page. Each of the devices is listed with IPv6 address together with their type.

CC2538EM node is identified as Sparrow Dual-Op and IoT-U10 node is identified as IoT-U10 and Zolertia RE-MOTE as Zoul RE-MOTE.

**CC2538EM board does not have any LED or sensors. Any features related to temperature sensors and LED controls are disabled. These hardware features are only available on IoT-U10 and Zolertia RE-MOTE.**

### 8.3.2 Pinging a 6LoWPAN Node

Nodes that connect to the 6LoWPAN network are listed in the device table. Each of the nodes has a ping button so that it is possible to ping the nodes from the web application.

Click the **"Ping node"** button to ping a node. The ping result is shown in a pop-up window.

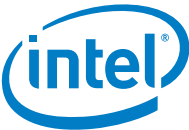
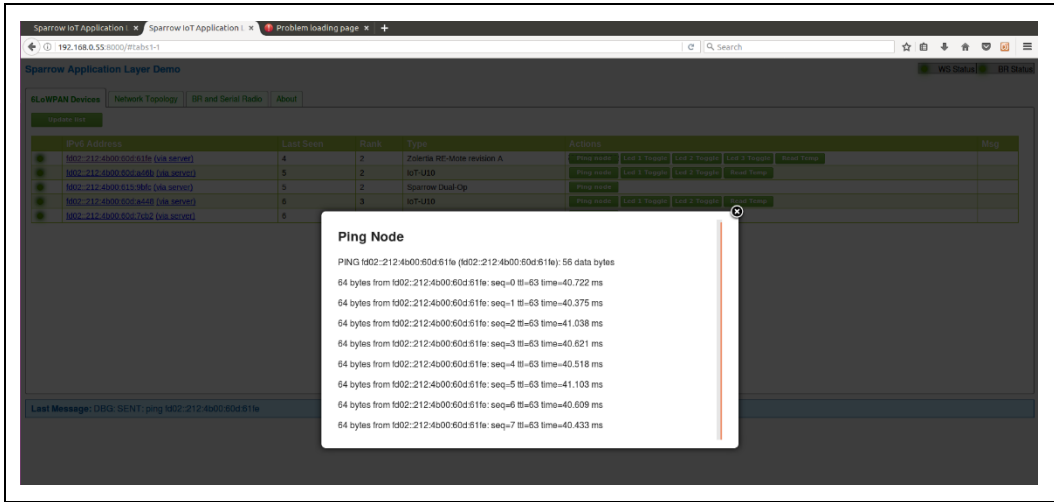


Figure 7. Example of Pinging Node using IoT WebDemo Server



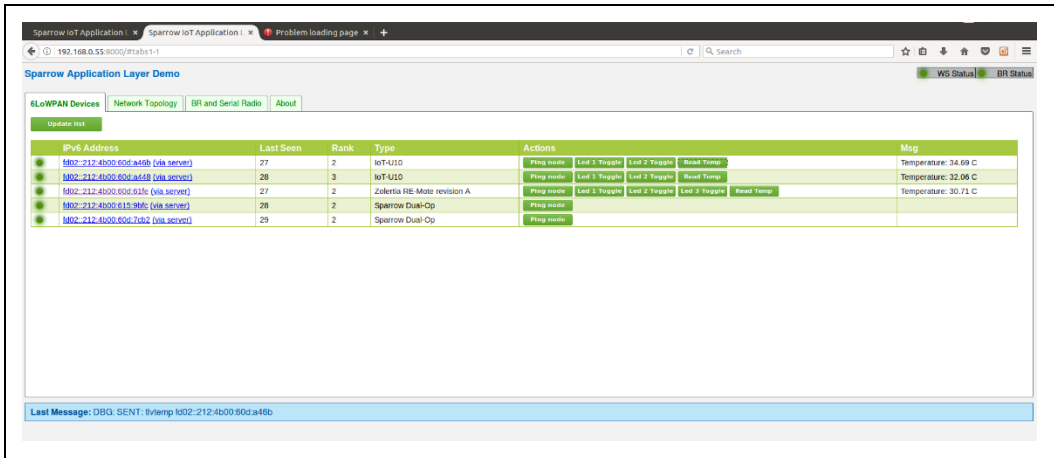
### 8.3.3 Toggle LEDs

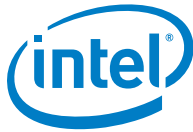
If the node is a Yanzi IoT-U10 radio device, there are two Toggle LED buttons on the same row as the IPv6 address of the node. If the node is a Zolertia RE-MOTE, there are three buttons to Toggle LED. Clicking these buttons toggles the corresponding LED on the IoT-U10 node and RE-MOTE.

### 8.3.4 Reading Temperature

There is a button for reading the temperature of the device. If clicked, the request is sent to the 6LoWPAN web server, which then sends a request for reading the temperature variable. The temperature is displayed at the end of the same row.

Figure 8. Example of Temperature Reading





### 8.3.5 HTTP Server

For IoT-U10 and Zolertia RE-MOTE's node, there is also a tiny web server that shows some information on the node. If your OS supports IPv6 and you are directly connected to the same network, you can either browse to [http://\[IPv6-address-of-node\]/](http://[IPv6-address-of-node]/).

The screenshot below shows that we can browse to the IPv6 address of the nodes on 6LoWPAN network with web browser. A simple webpage with some links is shown on the browser. Users can click the links displayed on the web page to retrieve some basic network information about the node.

**Note:** Make sure the node firmware is compiled with `MAKE_WITH_WEBSERVER=1` to enable HTTP support for nodes.

Figure 9. IPv6 Address of the Nodes on 6LoWPAN Network



### 8.3.6 Network Topology View

The screenshot shown below is the network topology view that shows two 6LoWPAN nodes that are connected to the NBR. All these nodes support the network statistics functional instance. This instance makes it possible to read out routing layer data such as Rank (position in the network) and Parent (the default route of the node) and some other information. This information is used to create a network topology graph.

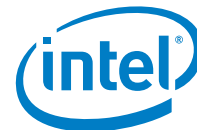
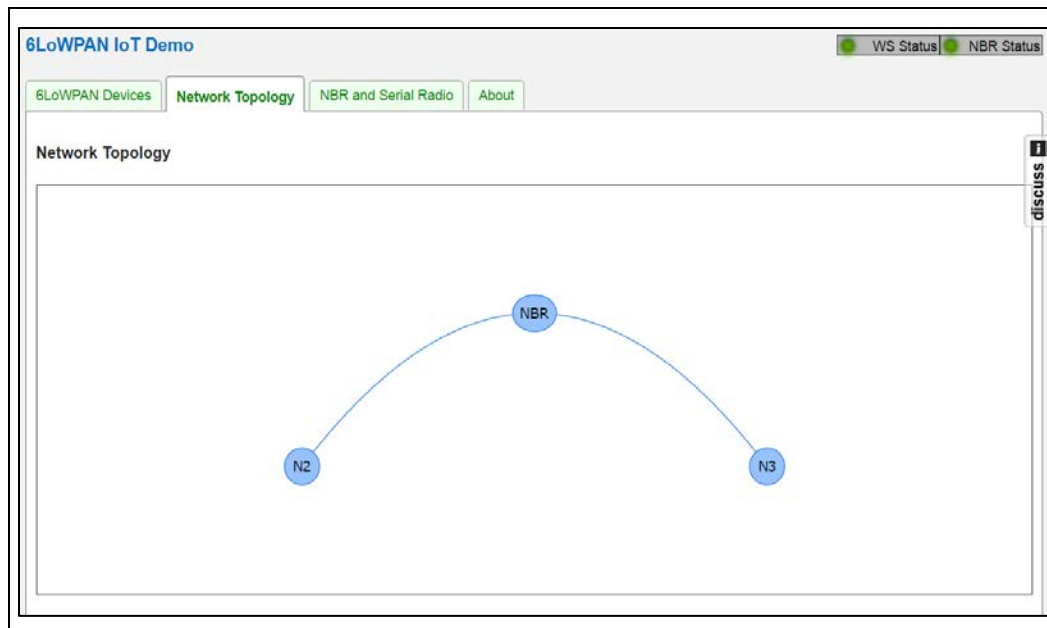


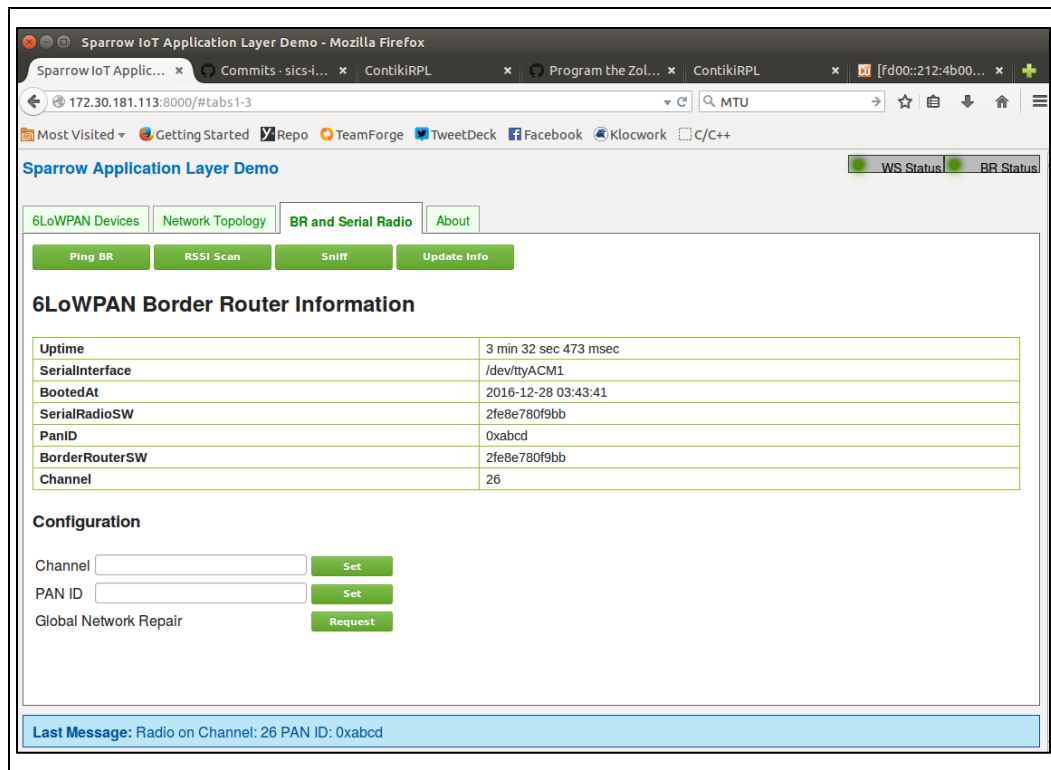
Figure 10. Network Topology View



### 8.3.7 Configuration Page

The configuration page of the 6LoWPAN webdemo server page shows some information about the network border router (**NBR**) and serial radio. It also allows configuration of channel, PAN ID, and link layer security key as shown in the screenshot below.

Figure 11. Configuration Page of NBR

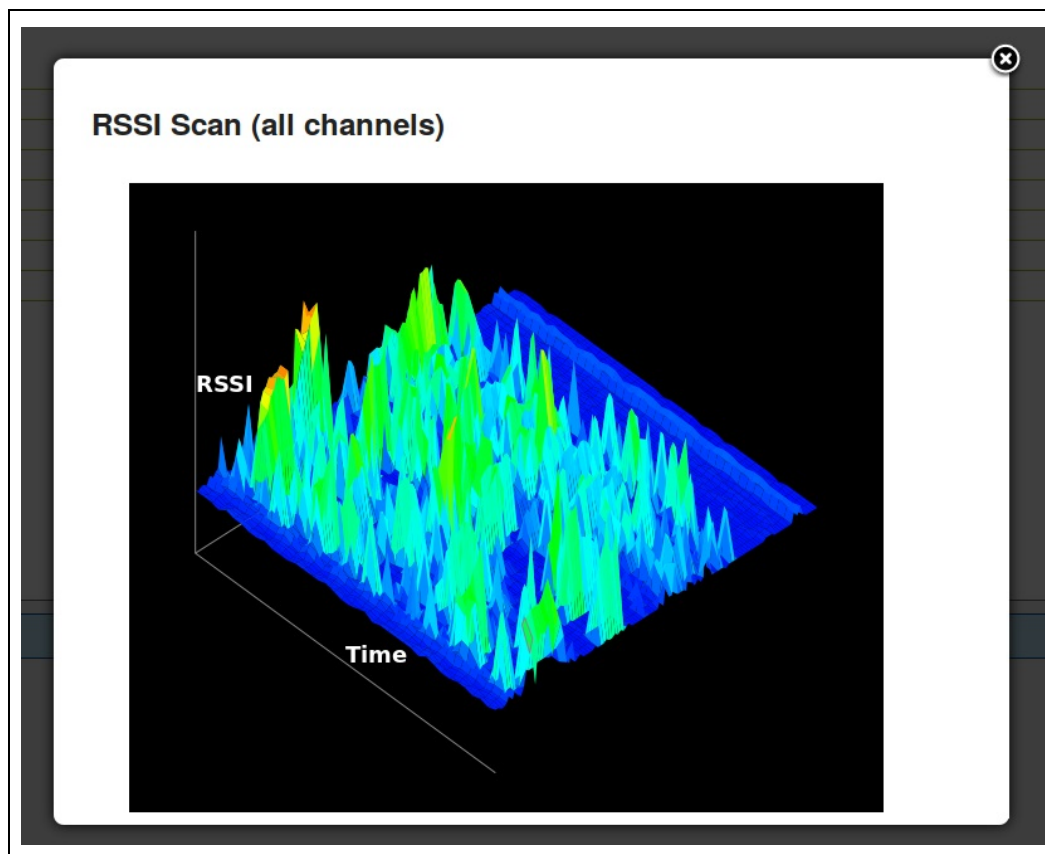


### 8.3.8 RSSI Scanner

The RSSI Scanner demo is started by clicking **"RSSI Scan"** on the **"NBR and Serial Radio"** webpage. The NBR then reconfigures the serial radio to act as a RSSI Scanner and it pops up a window within the webpage that show the RSSI scanning results in a 3D surface plot. The graph is updated as long as the sniffer mode is running. While the scanner is running, the NBR will not manage the 6LoWPAN network.

To exit the scanner mode, click outside the window or at the closing button on the upper right corner.

Figure 12. RSSI Scanner

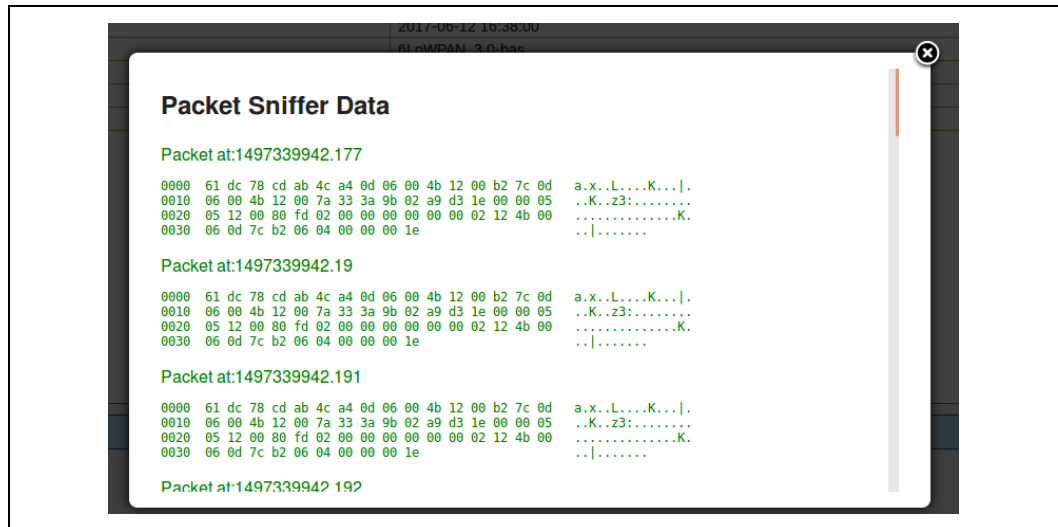


### 8.3.9 802.15.4 Sniffer

The sniffer demo is started by clicking the "Sniffer" button on the "NBR and Serial Radio" webpage. It sniffs a set of packets for a short time and displays the packets in hexadecimal in a pop-up window. While the sniffer is running, the NBR will not manage the 6LoWPAN network.



Figure 13. 802.15.4 Sniffer



To exit the sniffer mode, click outside the window or on the closing button on the upper right corner.





## 9.0 Over-The-Air (OTA) Firmware Upgrade Tool

This section provides information on Over-the-Air (OTA) firmware upgrade tool. This tool allows flashing firmware wirelessly into the hardware devices.

### 9.1 Using the 6LoWPAN Over-the-Air Firmware Upgrade Tool

Make sure the hardware device already flashed with the rescue image generated as described in **Section 6 Building Firmware**.

1. User needs to generate a .jar firmware binary to be able to use this firmware upgrade tool. Refer to **Section 6** for details.
2. Open a new terminal. Transfer the .jar file using scp command from user's Linux development machine to Bay Trail board's **/opt/6lowpan/sbin**.

```
# scp ilab@<your ip address>:<file source> <file destination>
```

Example:

```
# scp ilab@172.30.181.113:~/6LoWPAN2.0-development
/repo/sparrow/examples/felicia/iot-u10/felicia-firmware.jar
/opt/6lowpan/sbin/tools/sparrow
```

**Note:** NBR application must be started in order to upgrade firmware. Refer to Section 7.1

3. Locate the OTA upgrade tool.

```
# cd /opt/6lowpan/sbin/tools/sparrow
```

4. Updating the firmware using OTA upgrade tool.

- To update the serial-radio: Execute the tlupgrade.py
 

```
# python tlupgrade.py -a 127.0.0.1 -i felicia-firmware.jar
```
- b. To update the node: Execute the tlupgrade.py
 

```
# python tlupgrade.py -a <node address> -i felicia-firmware.jar
```

Example:

```
# python tlupgrade.py -a fd02:212:4b00:60d:a46c -i felicia-firmware.jar
```

5. To check status for Serial-Radio and Node:

- Serial-Radio:
 

```
# python tlupgrade.py -s -a 127.0.0.1
```
- Node:
 

```
# python tlupgrade.py -s -a <node address>
```

Example:

```
# python tlupgrade.py -s -a fd02:212:4b00:60d:a46c
```

## Appendix A

### A.1 UART Connection

CC2538DK (acts as serial radio) supports a UART connection with NBR (Intel Atom Processor E3800/ Intel Atom Processor E3900 Series). Refer to the schematic for each of the products for more details.

Figure 14. UART Connection Block Diagram for Baytrail-I

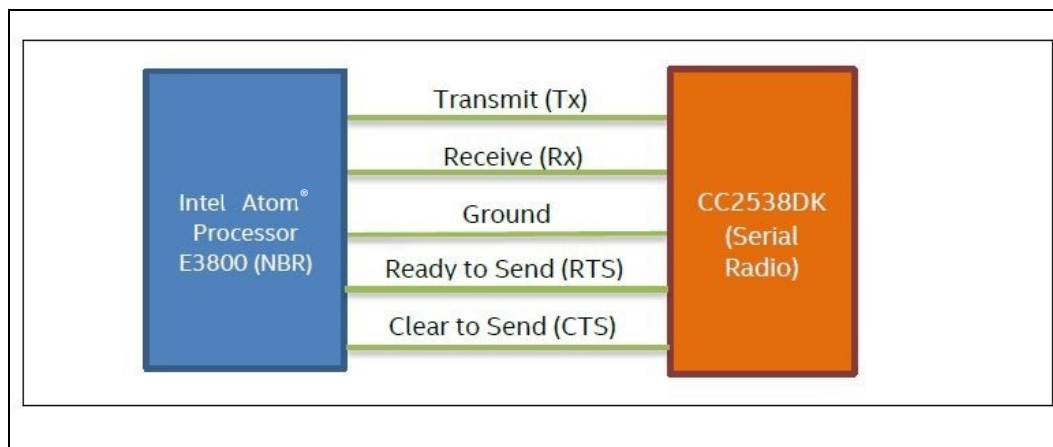
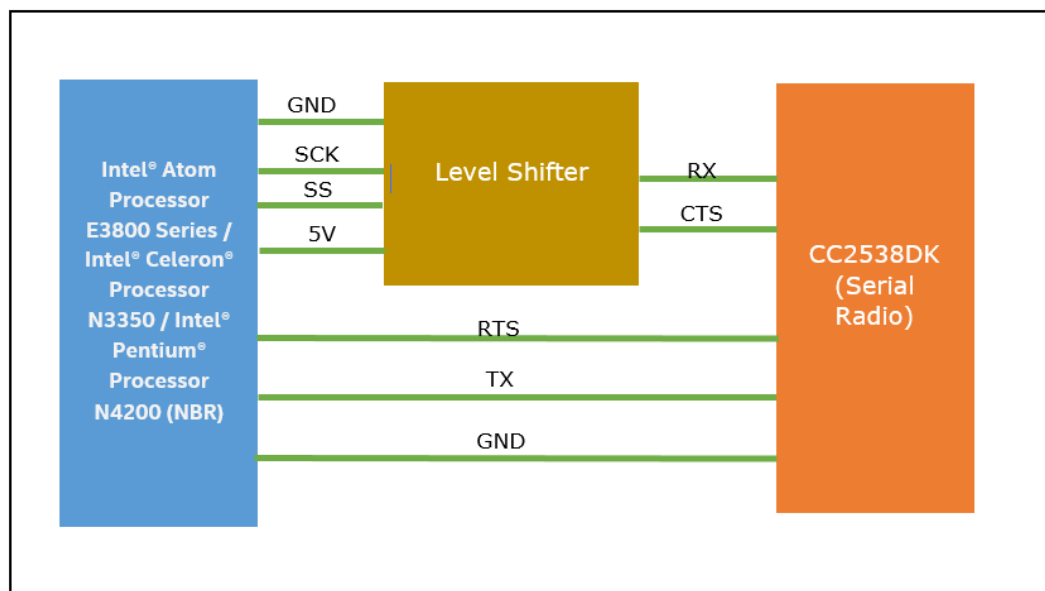


Figure 15. UART Connection Block Diagram for Apollo Lake -I



§