Radiocrafts

Embedded Wireless Solutions

SDK Quick Start

RIIM – Radiocrafts Industrial IP Mesh
Rev. 1.70

Table of Figures

Terms and Abbreviations

Abbreviation	Description
ADC	Analog-to-Digital Converter
API	Application Programming Interface
CoAP	Constrained Application Protocol
DTLS	Datagram Transport Layer Security
GPIO	General Purpose Input/Output
I^{2}C	Inter-Integrated Circuit
ICI	Intelligent C-Programmable I/O
Image	A compiled (generated) binary file ready for upload to a RIIM module. This can be a ICI application made by the user or a Platform file made by Radiocrafts
MAC	Media Access Control
OSI	Open Systems Interconnection
PAN	Personal Area Network
PHY	Physical Layer of the OSI model
RF	Radio Frequency
RIIM	Radiocrafts Industrial IP Mesh
SDK	Software Development Kit
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver Transmitter
UDP	User Datagram Protocol
VSCode	Microsoft Visual Studio Code
WSN	Wireless Sensor Network

1 Intro

The purpose of this document is to guide the reader though the installation and setup of the RIIM SDK and through the process of compiling and downloading a pre-made ICI application in the SDK.

2 Introduction to RIIM

The RIIM network consists of these key elements

- The RIIM SDK
  - Software development kit with ICI application frameworks and tools for creating and uploading end ICI applications to the RC1882-IPM
- The RC1882-IPM module
  - The RC1882-IPM module can be configured as Border Router node, Mesh Router node or Leaf node.
    - As a Border Router it acts as the base of the mesh network. It can connect to an external network via ethernet or custom user ICI application on other interfaces such as UART
    - As a Mesh Router, it will be able to transport packets in the RIIM mesh network
    - As a Leaf, it is not able to transport packets to other nodes except its parent. This mode uses the least amount of energy.
  - All node configurations require an ICI application for RF and interface configuration and the user application. The same RIIM Software Development Kit (SDK) is used to create the ICI application for for all node configurations.

Below is an illustration of the different elements and the documentation available

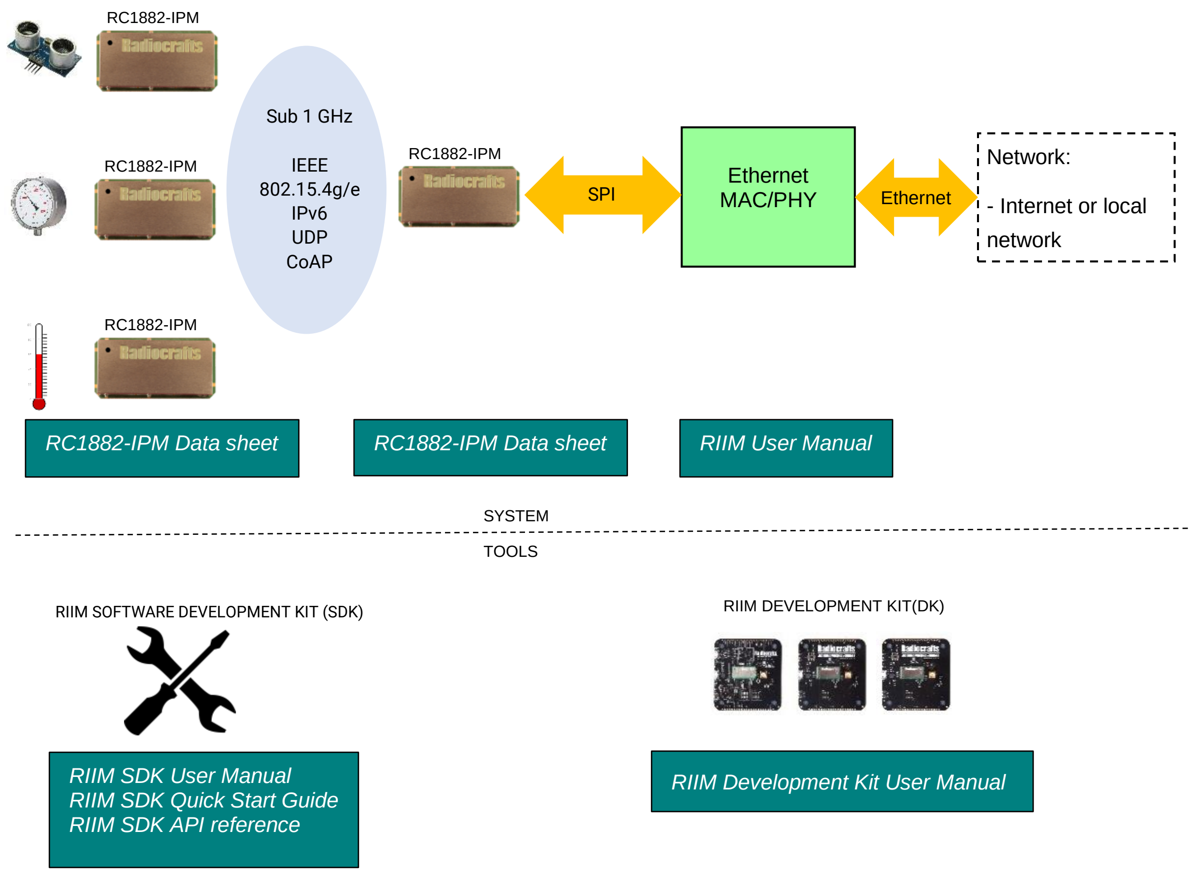


Figure 1. RIIM network – system and documentation overview

3 Working with the SDK

The SDK runs on both Linux and Windows, and setup of both systems are shown in the following two chapters.

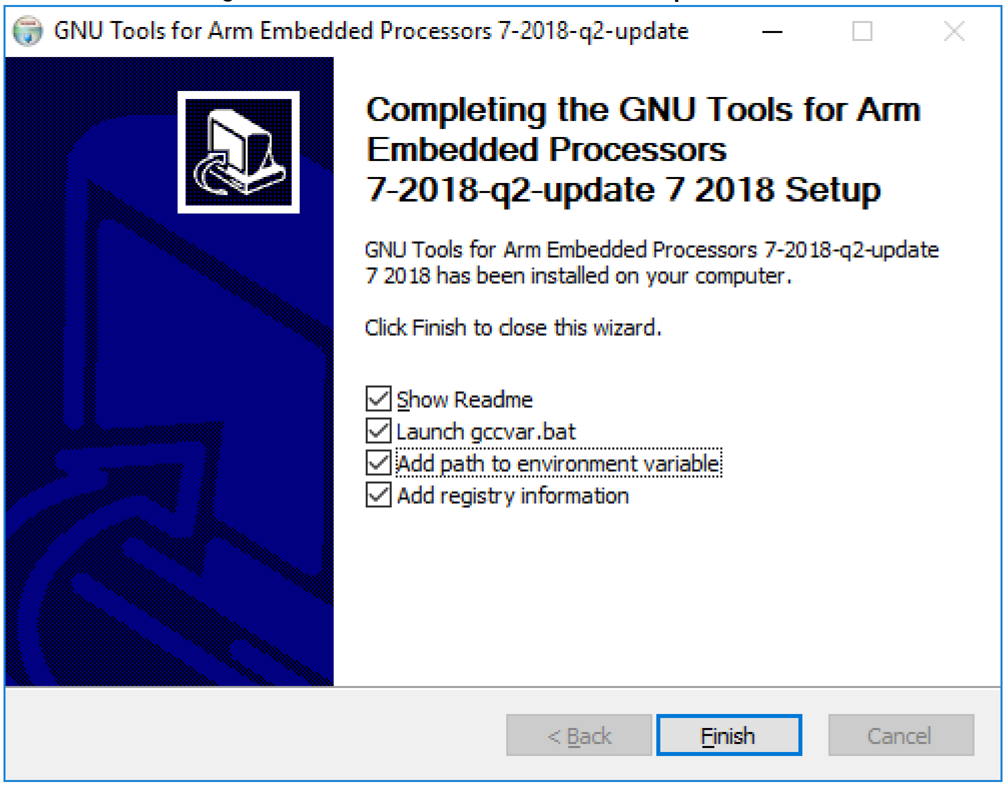
3.1. Windows

To use the SDK on windows, you will need GNU Embedded Toolchain for ARM, Windows version of the MAKE tool, and of course the SDK files.

3.1.1. Installing GNU Embedded Toolchain for ARM

The toolchain can be downloaded from the official ARM web page at https://developer.arm.com/open-source/gnu-toolchain/gnu-rm/downloads .
Radiocrafts recommends that you use the 10.3 based variant[1] - scroll down the ARM web page to find it.

If you are running windows 7 or later, most want to use the sha-2 version (Signed installer for windows 7 and later). Installation is straight forward, but ensure to check **"Add path to environment variable"** as shown here:



Completing the GNU Tools for Arm Embedded Processors 7-2018-q2-update 7 2018 Setup

GNU Tools for Arm Embedded Processors 7-2018-q2-update 7 2018 has been installed on your computer.

Click Finish to close this wizard.

☑ Show Readme
☑ Launch gccvar.bat
☑ Add path to environment variable
☑ Add registry information

[1] Other versions will probably also work, but this is the one tested and supported by Radiocrafts

3.1.2. Installing the MAKE tool

The MAKE tool can be downloaded from sourceforge.net here:
http://gnuwin32.sourceforge.net/packages/make.htm
Most users will need the package described on that page as «Complete package, except sources»

3.1.3. Installing the SDK

After downloading the SDK from the Radiocrafts web site, unpack the zip file to your desired folder. No real installation procedure is required.

3.1.4. Installing a terminal program

A terminal program is used to connect to the module via the UART. In windows, this connection will show up as a COM-port. See chapter **9.3 How do I find my UART ports on Windows** for details on finding the ports.
There are a lot of different terminal programs to choose from, for instance Microsoft Terminal (https://github.com/microsoft/terminal) or Tera Term (https://ttssh2.osdn.jp/).

3.2. Linux

To use the SDK on Linux (Ubuntu), you can install GNU Embedded Toolchain for ARM and unpack the SDK files. See the commands below.

```
$ sudo apt-get install gcc-arm-none-eabi
$ unzip RIIM_SDK.zip
```

Or you can download a specific version from at https://developer.arm.com/open-source/gnu-toolchain/gnu-rm/downloads and follow the installation instructions

3.2.1. Installing a terminal program

A simple terminal program for linux is **microcom**. You can install it by typing

```
$ sudo apt-get install microcom
```

3.3. SDK Directory Structure

The SDK directory contains the following:

- docs/
    - The documentation
- Framework/
    - The SDK internals. **Do not modify the content unless you know what you are doing.**
- Framework/Linker/
    - Files used during linking.
- Framework/Makefile
    - Used by the build process.
- Framework/Tools/
    - Containing the command line tools and scripts. These are described in detail later.
- Framework/User_API/
    - Containing the API header files
- ICI_Applications/

- Your code as well as example ICI applications. **This is where most users will work.**

3.4. Example structure and naming conventions

The examples contained in the **ICI_Applications** use a common structure and naming convention. The structure is like this:

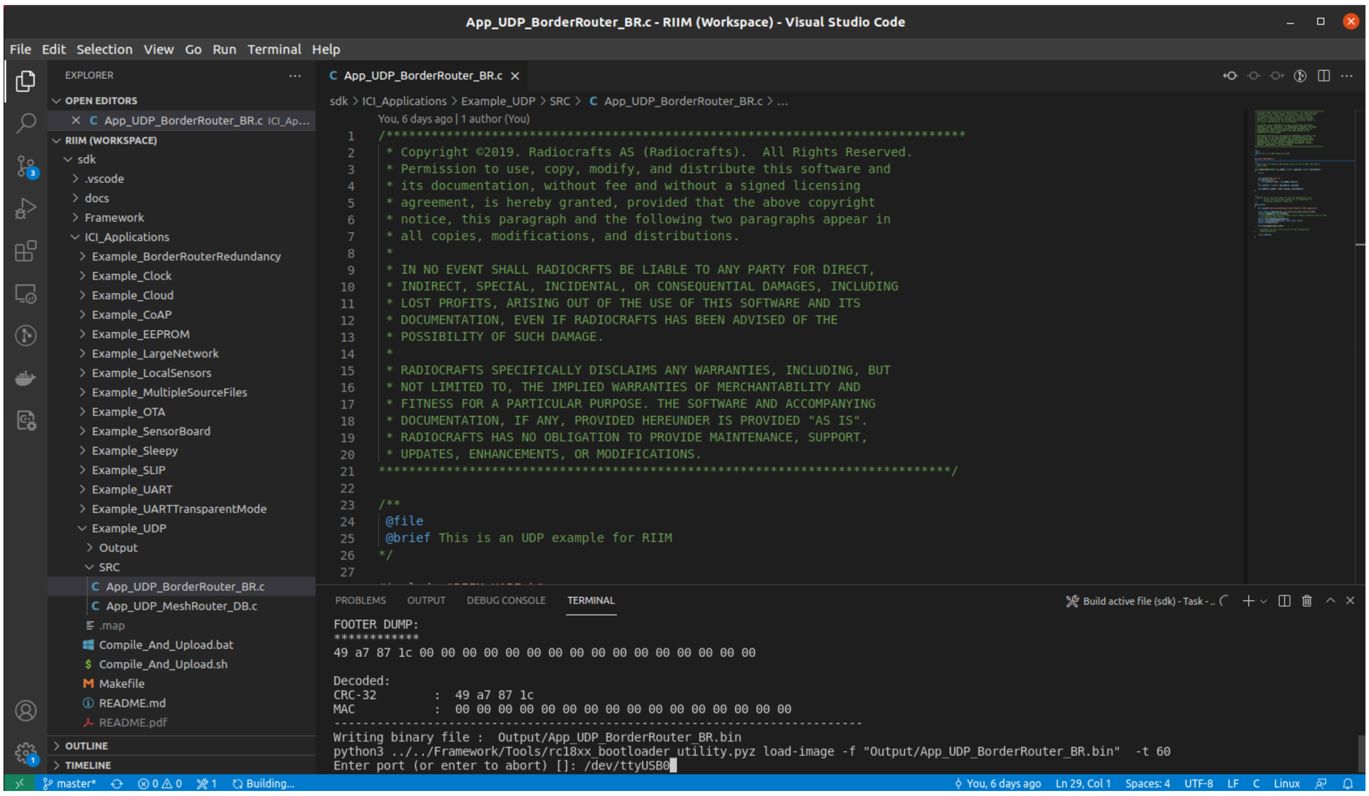
```
My_App
├── Compile_And_Upload.bat          Automatic build script for Windows
├── Compile_And_Upload.sh           Automatic build script for Linux
├── Makefile                        Makefile for use with make
├── Output                          Generated images to be transferred to module
│   ├── App_MyApp_Leaf_SB.bin        Image for the platform running on the SB
│   ├── App_MyApp_BorderRouter_BR.bin Image for the platform running on the BR
│   └── App_MyApp_MeshRouter_DB.bin  Image for the platform running on the DB
├── README.md                       Readme file in markdown format
├── README.pdf                      The same readme file in PDF format
└── SRC                             The source files to be built into images
    ├── App_MyApp_Leaf_SB.c         Source file for the platform running on SB
    ├── App_MyApp_BorderRouter_BR.c Source file for the platform running on BR
    └── App_MyApp_MeshRouter_DB.c   Source file for the platform running on DB
```

The naming convention of the source and image files is as follows:
`App_`*`Description`*`_`*`Platform`*`_`*`Board`*`.c`

- `Description`
  - `Describes the ICI application, such as: ADC, Accelerometer, Cloud, LargeNetwork`
- `Platform`
  - `Describes the RIIM platform it is made for. RIIM has 3 platform types: BorderRouter, MeshRouter and Leaf`
- `Board`
  - `Describes what board the image is supposed to run on. These corresponds to the 3 types of boards contained in the Development Kit from Radiocrafts. They can be:`
    - `DB Development Board`
    - `SB Sensor board`
    - `BR Border Router`

4 Building and uploading the example ICI application using VSCode

The SDK provides a workspace file and setup for compilation and uploading ICI applications in Microsoft Visual Studio Code. To open the SDK in VSCode, open the file **RIIM.code-workspace** present at the top level of the SDK. To build an ICI application, simply select the file you want to compile in the leftmost EXPORER window and press **SHIFT-CTRL-B**. This builds whatever file you have selected and is open in the editor.
When building is finished, you are prompted «Enter port (or enter to abort) []:» in the bottom TERMINAL window. Here you enter the port your RIIM module is connected to, for instance **COM3** or **/dev/ttyUSB0**. Or just press **ENTER** to exit without uploading anything.


App_UDP_BorderRouter_BR.c - RIIM (Workspace) - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
RIIM (WORKSPACE)
sdk
.vscode
docs
Framework
ICI_Applications
Example_BorderRouterRedundancy
Example_Clock
Example_Cloud
Example_CoAP
Example_EEPROM
Example_LargeNetwork
Example_LocalSensors
Example_MultipleSourceFiles
Example_OTA
Example_SensorBoard
Example_Sleepy
Example_SLIP
Example_UART
Example_UARTTransparentMode
Example_UDP
Output
SRC
App_UDP_BorderRouter_BR.c
App_UDP_MeshRouter_DB.c
.map
Compile_And_Upload.bat
Compile_And_Upload.sh
Makefile
README.md
README.pdf
OUTLINE
TIMELINE
sdk > ICI_Applications > Example_UDP > SRC > App_UDP_BorderRouter_BR.c > ...
You, 6 days ago | 1 author (You)
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Build active file (sdk) - Task - ..
master* 0 0 1 Building...
You, 6 days ago Ln 29, Col 1 Spaces: 4 UTF-8 LF C Linux

```
/****************************************************************************
 * Copyright ©2019. Radiocrafts AS (Radiocrafts).  All Rights Reserved.
 * Permission to use, copy, modify, and distribute this software and
 * its documentation, without fee and without a signed licensing
 * agreement, is hereby granted, provided that the above copyright
 * notice, this paragraph and the following two paragraphs appear in
 * all copies, modifications, and distributions.
 *
 * IN NO EVENT SHALL RADIOCRFTS BE LIABLE TO ANY PARTY FOR DIRECT,
 * INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING
 * LOST PROFITS, ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS
 * DOCUMENTATION, EVEN IF RADIOCRAFTS HAS BEEN ADVISED OF THE
 * POSSIBILITY OF SUCH DAMAGE.
 *
 * RADIOCRAFTS SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT
 * NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND
 * FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE AND ACCOMPANYING
 * DOCUMENTATION, IF ANY, PROVIDED HEREUNDER IS PROVIDED "AS IS".
 * RADIOCRAFTS HAS NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT,
 * UPDATES, ENHANCEMENTS, OR MODIFICATIONS.
***************************************************************************/

/**
 @file
 @brief This is an UDP example for RIIM
*/

```

```
FOOTER DUMP:
************
49 a7 87 1c 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Decoded:
CRC-32        :  49 a7 87 1c
MAC           :  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
--------------------------------------------------------------------------
Writing binary file :  Output/App_UDP_BorderRouter_BR.bin
python3 ../../Framework/Tools/rc18xx_bootloader_utility.pyz load-image -f "Output/App_UDP_BorderRouter_BR.bin"  -t 60
Enter port (or enter to abort) []: /dev/ttyUSB0
```

Figure 2 - Using VSCode for compilation and upload

5 Building and uploading the example ICI application using scripts

To build and upload your first ICI application, go to the folder **ICI_Applications/My_App** and execute **Compile_And_Upload.bat** (Windows) or **Compile_And_Upload.sh** (Linux). Execution can also be done by double-clicking on the file from the file explorer, or by execution from the command line.

This will take the source files provided in the SDK and create image files that are ready to be uploaded to the modules that are mounted on the boards in the Development Kit.

After the compilation and image generation is finished, you'll end up with this:

```
Output/App_MyApp_Leaf_SB.bin is generated. Do you want to upload it to the module
/ board now?
Enter your SERIAL port name e.g. /dev/ttyUSB0 and hit ENTER
If you only hit ENTER, you will skip uploading

Enter serial port name:
```

Here you will have to type in the name of the serial port / COM-port that your board is connected to. See chapter **11.3 "How do I find my UART port on Windows"** on details on how to determine which COM port to use. Remember also to type in the whole COM port name, not just the number. For instance : "COM23", **not** just "23"

The filenames themselves describes what board the image is to be uploaded to. See chapter 3.4 for a description on the naming convention used. In the example above: App_MyApp_Leaf_SB means that this image should be uploaded to the Sensor Board.

```
Enter serial port name: /dev/ttyUSB0

……………

Loading Image

Waiting for Bootloader to initiate transfer...

Start transfer:

End transfer

file upload successful

Waiting for bootloader status..

Bootloader Status: Success




----------------------------------------------

ALL DONE!

----------------------------------------------

Press enter to continue
```

You need to repeat this compile and upload process for each of the three boards in the development kit. Pay attention so that the right ICI application is uploaded on the right board.

When finished, you need to reset the board(s) you just uploaded the image to. The reset button is located on the side of the board. The modules on your boards are now programmed with the ICI applications that are needed to connect the boards and start your RIIM network.

Descriptions of the examples are in the README-files in the Example-directories.

6 Compile the ICI application manually using MAKE

Important: This requires that you have the MAKE tool installed correctly on your machine. See chapter 3.1.2 for description on how to install this.

Makefiles are provided with all examples and the My_App ICI application. To compile your program, first enter the directory at the command line. Then type in **make** and specify source file name to be compiled like in this example:

```
make SOURCE_FILE=SRC/App_MyApp_Leaf_SB.c
```

This will compile the file **App_MyApp_Leaf_SB.c** located in the **SRC** directory. You can change this to whatever you like. When compiling, an output similar to this will appear in the command line window:

```
~/RIIM_SDK/ICI_Applications/My_App$ make SOURCE_FILE=SRC/App_MyApp_Leaf_SB.c
Starting creation of RIIM user application
RIIM user application created. Output file is Output/App_MyApp_Leaf_SB.hex

   text    data     bss     dec     hex  filename
     64       4       0      68      44  Output/App_MyApp_Leaf_SB.elf
../../Framework/Tools/rc188x_image_generator -t app -p 2 -hw 2 -f Output/App_MyApp_Leaf_SB.hex
RC188x Image Generator
Reading hex file :  Output/App_MyApp_Leaf_SB.hex
Creating image
Creating header
No keyfile specified. No encryption is performed
---------------------------------------------------------------------
HEADER DUMP:
************
00 02 00 02 00 00 02 00 00 00 23 00 00 00 44 00 dd 35 00 00 37 bc e9 71 3e aa 30 00 00 00 00 00

Decoded:
Platform_ID   :  00 02
Hardware_ID   :  00 02
Hardware_Rev  :  00 00
Image_Type    :  02
Version       :  00 00 00
Start_Page    :  23
Image_Length  :  00 00 00 44
Option        :  00
Nonce         :  dd 35 00 00 37 bc e9 71 3e aa 30
Reserved      :  00 00 00 00 00
---------------------------------------------------------------------
---------------------------------------------------------------------
FOOTER DUMP:
************
41 5b 3a 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Decoded:
CRC-32        :  41 5b 3a 01
MAC           :  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
---------------------------------------------------------------------
Writing binary file :  Output/App_MyApp_Leaf_SB.bin
```

App_MyApp_LEAF_SB.bin is the compiled image and is now located in the Output directory and is ready to be uploaded to the module.

7 Upload the image to the module manually using MAKE

The image you just created can now be uploaded to the module via UART. If you are using a Radiocrafts development board, a port called **COM*** (windows) or **/dev/ttyUSB*** (Linux) will be available for upload. See chapter 11.3 “How do I find my UART port on Windows” on details on how to determine which COM to use.

Uploading of the image is done using the **make** tool with the argument “uploadImage”. If your port is not /dev/tyyUSB0, you need to specify port when uploading (for example: **make uploadImage PORT=COM3**)

Important: You will need to reset the board *after* starting the tool to enable the tool to connect to the bootloader and upload the image. Reset should be done when the messages “Cannot connect….” start arriving.

```
~/dev/RIIM_SDK/ICI_Applications/My_App$ make uploadImage
OUTPUT_FILENAME=Output/App_MyApp_Leaf_SB PORT=/dev/ttyUSB0
../../Framework/Tools/rc188x_bootloader_utility load-image -f
Output/App_MyApp_Leaf_SB.bin -p /dev/ttyUSB0 -t 60
Cannot connect to Bootloader. Failed attempt  1.0  of  60.0 .
Cannot connect to Bootloader. Failed attempt  2.0  of  60.0 .
Cannot connect to Bootloader. Failed attempt  3.0  of  60.0 .
MODULE INFORMATION
EUI64: 00124B001CBCAAF7
Hardware ID: 0x0002 (Unknown)
Hardware Rev: 0x0002
Platform ID: 0x0002 (Unknown)
Platform Version: v35.9.0
App Version: v255.255.255
Bootloader Version: v3.0.0
Bootloader Variant: 0x02 (IP Mesh)
Lock State: 0xFF (Unlocked)

Loading Image
Waiting for Bootloader to initiate transfer...
Start transfer:
End transfer
file upload successful
Waiting for bootloader status..
Bootloader Status: Success
~/dev/RIIM_SDK/ICI_Applications/My_App$
```

And finally, start the ICI application by resetting the board you uploaded your image to.

8 Observing the output

The ICI application simply starts the module as a node and prints out a greeting + configuration. If you open the UART (**COM*** / **ttyUSB***) in a terminal program using a baudrate of 115200 and reset the board, you should see something like this *message* when the board is startet (reset):

```
~/RIIM_SDK/ICI_Applications/My_App$ microcom -s 115200 -p /dev/ttyUSB0 connected to /dev/ttyUSB0
Escape character: Ctrl-\
Type the escape character followed by c to get to the menu or q to quit
Starting RIIM Node
# RIIM node configuration:
# - Application Version : 0xffffffff
# - Hardware ID : 0x0000
# - Hardware Version : 0x0000
# - Platform ID : 0x0002
# - Platform Version : 0x00000923
# - Platform Description : RIIM_NODETYPE_MeshRouter
# - MAC Address : 0xf7aabc1c004b1200
# - PAN ID: 0x9812
# - Node ID: 43767
# - Link-layer address: aaf7
# - Local IPv6 address: fe80:0000:0000:0000:0000:00ff:fe00:aaf7:
# - Global IPv6 address: 4c46:0101:0100:0000:0000:0000:0000:0200:
# - Reset code: 0x00000001
```

9 Platform versions

The platforms come in several versions and combinations. If you want frequency hopping, choose the platform whose file ending is “TSCH”. If you want single channel, choose the platform whose file ending is “CSMA”. The difference is described more in detail in the RIIM User Manual.

NB! Note that when opening an development kit or sample modules these can have a platform image from SDK version earlier than the SDK downloaded from Radiocrafts.com. So we recommend to always update the platform on your kit when downloading an SDK.

10 Upgrading or changing the Platform image automatically

Changing and upgrading the Platform Image is also possible through automated scripts. The scripts for doing so are located in the folder **Framework/Platform/** . These work just as the automatic scripts for ICI applications described earlier in chapter 4.

The automatic script first prepare the “single channel” variant of the platform then asks you to enter the com port number, if you press “enter” to skip, then it will prepares the “TSCH- Frequency Hopping” variant of the same platform and ask you for the com port number. However, in the “Output” folder you will find .bin files ready made for each variant, which you can directly upload to a board by using the “Bootloader Utility”.

The scripts can be run from the command line or double-clicked on from the file explorer. They are called **Upload_BorderRouter_Platform_[TYPE]**, **Upload_MeshRouter_Platform_[TYPE]** and **Upload_Leaf_Platform_[TYPE]**.**[TYPE]** refers to the network type and can be **SingleChannel** or **TSCH**.
Executing the script will prompt for the name of the serial port, just like for ICI applications:

```
~/ RIIM_SDK/Framework/Platform$ ./Upload_Leaf_Platform_SingleChannel.sh

----------------------------------------------------------------------
Output/RIIM_Platform_Leaf_0x020000-SingleChannel.bin is generated. Do you want to
upload it to the module / board now?
Enter your SERIAL port name e.g. /dev/ttyUSB0 and hit ENTER
If you only hit ENTER, you will skip uploading

Enter serial port name:
```

When finished, you need to reset the board(s) you just uploaded the image to.

11 Upgrading or changing the Platform image manually using MAKE

Uploading a new platform image manually is done in the same way as for the user ICI application images. For instance, you could do it like this to upload the BorderRouter platform :

```
~/RIIM_SDK/ICI_Applications/My_App$ make uploadImage
OUTPUT_FILENAME=../../Framework/Platform/Output/RIIM_Platform_BorderRouter_0x01010
0 PORT=/dev/ttyUSB0
```

You will need to change the PORT to your port (COMxx on Windows, or /dev/ttyUSBx on Linux)

When finished, you need to reset the board(s) you just uploaded the image to.
Factory programmed development kits come with this configuration:

- The sensor board (SB) uses RIIM_Platform_Leaf_0xXXXXXX
- The development board (DB) uses RIIM_Platform_MeshRouter_0xXXXXXX
- The border router (BR) uses RIIM_Platform_BorderRouter_0xXXXXXX

,Where XXXXXX is the platform version number. To find the available platform versions, look into the Framework/Platform/Output directory.

12 The way forward

Now you are ready to start testing. A nice place to start can be to edit the file **App_MyApp_Leaf_SB.c** and see that your changes is reflected in the executing code.

Be sure to go to www.radiocrafts.com for the latest SDK, documentation and knowledge.

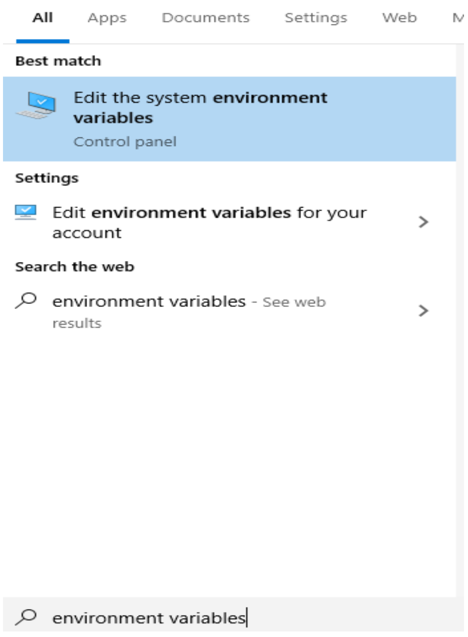
13 Troubleshooting

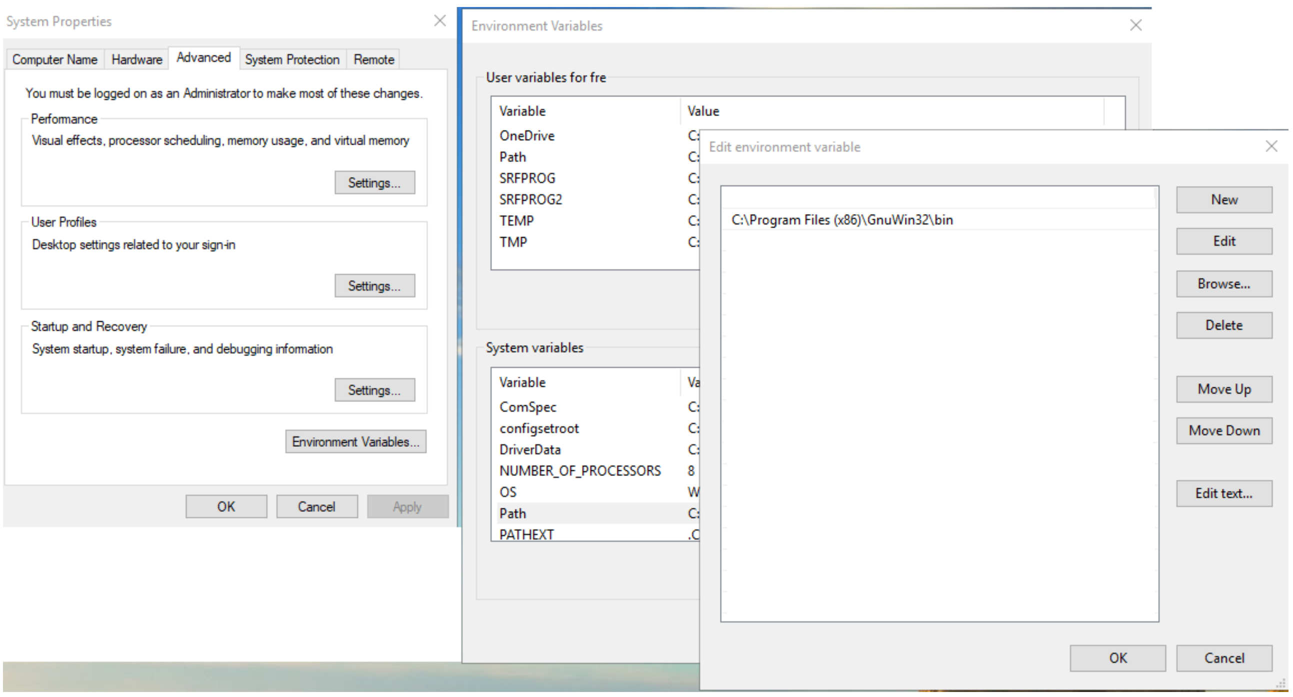
*13.1. I'm not able to use **make** on Windows*

The **make** tool may not write its location to the system PATH variable.

To fix this:

1. Go to «System Propterties». You can search for «environment variables» in the Start menu.
2. Select the «Advanced» tab
3. Click «Environment Variables» near the bottom of the window
4. Select Path from the upper list (User variables). Do **not** use the «System variables»
5. Select Edit
6. In the new window, select «New»
7. Fill in the path (Probably C:\Program Files (x86)\GnuWin32\bin) and click «OK»


All
Apps
Documents
Settings
Web
Best match
Edit the system environment variables
Control panel
Settings
Edit environment variables for your account
Search the web
environment variables - See web results
environment variables


System Properties

Computer Name | Hardware | Advanced | System Protection | Remote

You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...

User Profiles
Desktop settings related to your sign-in
Settings...

Startup and Recovery
System startup, system failure, and debugging information
Settings...

Environment Variables...

OK | Cancel | Apply

Environment Variables

User variables for fre

Variable	Value
OneDrive	C:
Path	C:
SRFPROG	C:
SRFPROG2	C:
TEMP	C:
TMP	C:

System variables

Variable	Va
ComSpec	C:
configsetroot	C:
DriverData	C:
NUMBER_OF_PROCESSORS	8
OS	W
Path	C:
PATHEXT	.C

Edit environment variable

C:\Program Files (x86)\GnuWin32\bin

New | Edit | Browse... | Delete | Move Up | Move Down | Edit text...

OK | Cancel

13.2. My *make* behaves strange

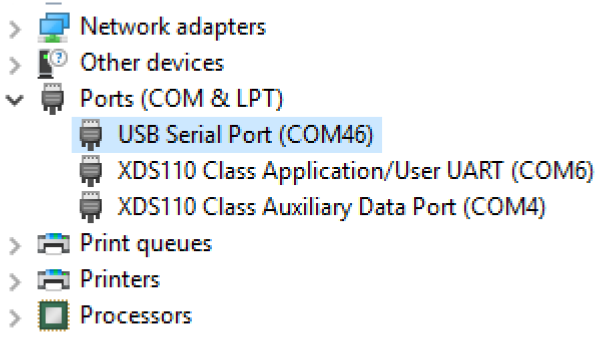
There may be other installations of **make** on your computer

To fix this:

- Remove the other installations of **make**
- Move Up/Down the path entries in the “Edit environment variable” window

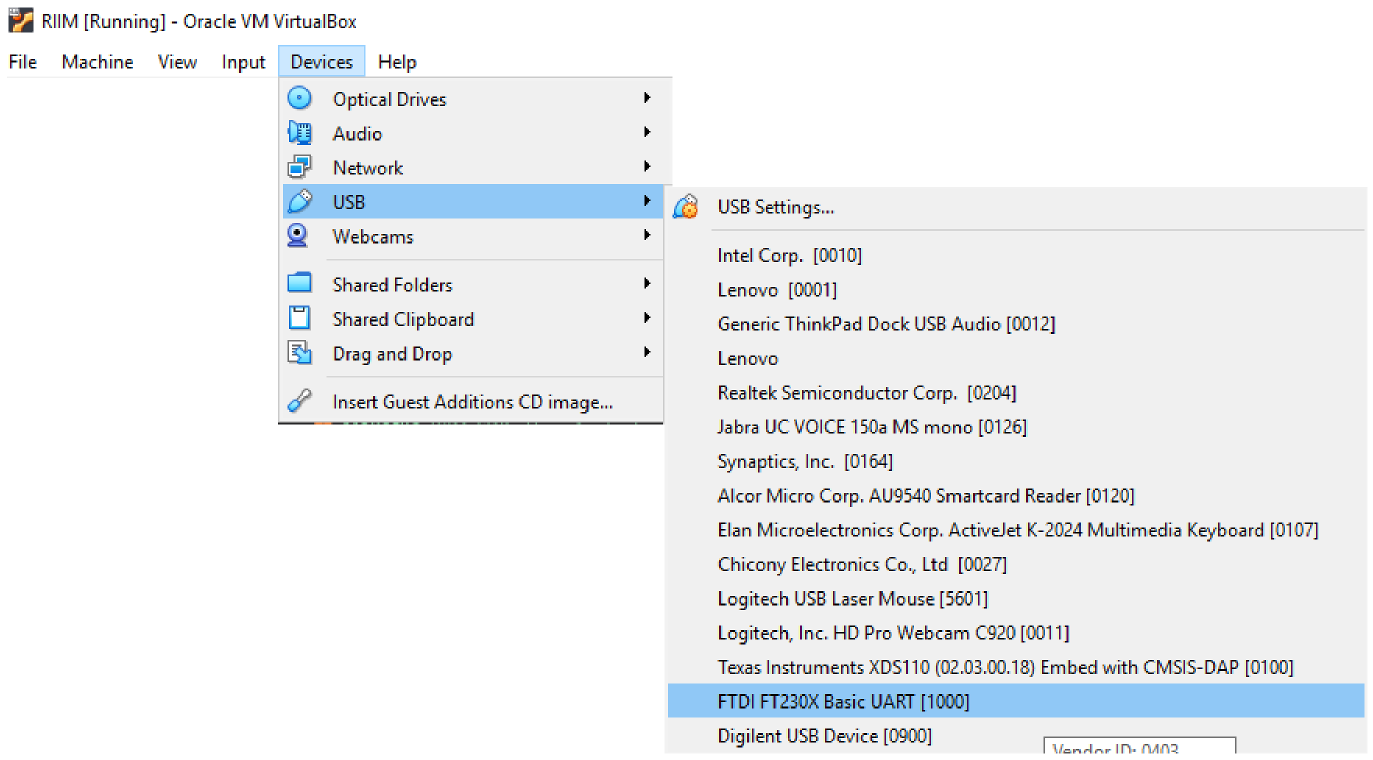
13.3. How do I find my UART Port on Windows?

When you connect a development board to a windows PC, Windows will install driver for it. After the driver has installed, you will have a new COM port listen in device manager like this:


- Network adapters
- Other devices
- Ports (COM & LPT)
  - USB Serial Port (COM46)
  - XDS110 Class Application/User UART (COM6)
  - XDS110 Class Auxiliary Data Port (COM4)
- Print queues
- Printers
- Processors

13.4. How do I connect the USB UART to my Linux Virtual Machine

Connect it using Devices->USB->FTDI FT230X Basic UART like this:


RIIM [Running] - Oracle VM VirtualBox
File
Machine
View
Input
Devices
Help
Optical Drives
Audio
Network
USB
Webcams
Shared Folders
Shared Clipboard
Drag and Drop
Insert Guest Additions CD image...
USB Settings...
Intel Corp. [0010]
Lenovo [0001]
Generic ThinkPad Dock USB Audio [0012]
Lenovo
Realtek Semiconductor Corp. [0204]
Jabra UC VOICE 150a MS mono [0126]
Synaptics, Inc. [0164]
Alcor Micro Corp. AU9540 Smartcard Reader [0120]
Elan Microelectronics Corp. ActiveJet K-2024 Multimedia Keyboard [0107]
Chicony Electronics Co., Ltd [0027]
Logitech USB Laser Mouse [5601]
Logitech, Inc. HD Pro Webcam C920 [0011]
Texas Instruments XDS110 (02.03.00.18) Embed with CMSIS-DAP [0100]
FTDI FT230X Basic UART [1000]
Digilent USB Device [0900]

13.5. *I get a permission denied on my ttyUSB port*

You need to be part of the “dialout”-group in Linux. You can add yourself using this command:
sudo adduser YOUR_USERNAME dialout
You need to log out and in again for the changes to take effect.

13.6. *How do I find my UART Port on Linux*

On Linux, you will have a device called /dev/ttyUSB* . The first board you plug in will get ttyUSB0, the next will have ttyUSB1 and so on. No driver installation is necessary.

Radiocrafts
Embedded Wireless Solutions

Document Revision History

Document Revision	Changes
1.00	Advance Information
1.10	Updated examples
1.20	Updated links, added troubleshooting for installing **make** and GCC, changed node names
1.30	Updated directory and file structures, better explanation of using the SDK tools, added automatic build and upload, description of platform image upload
1.40	Fixed spelling, updated tools and platforms
1.50	Updated with CSMA and TSCH platform variants
1.60	Updated platform upgrade script description
1.70	Added chapter for VSCode, changed recommended compiler

Disclaimer

Trademarks

Life Support Policy

Radiocrafts Webpage

For more info go to our web page : https://radiocrafts.com/
There you can find Knowledge base and Document Library that includes Application notes, Whitepapers, Declaration of Conformity, User Manuals, Data Sheet and more.

Contact Radiocrafts

Sales requests: https://radiocrafts.com/contact/