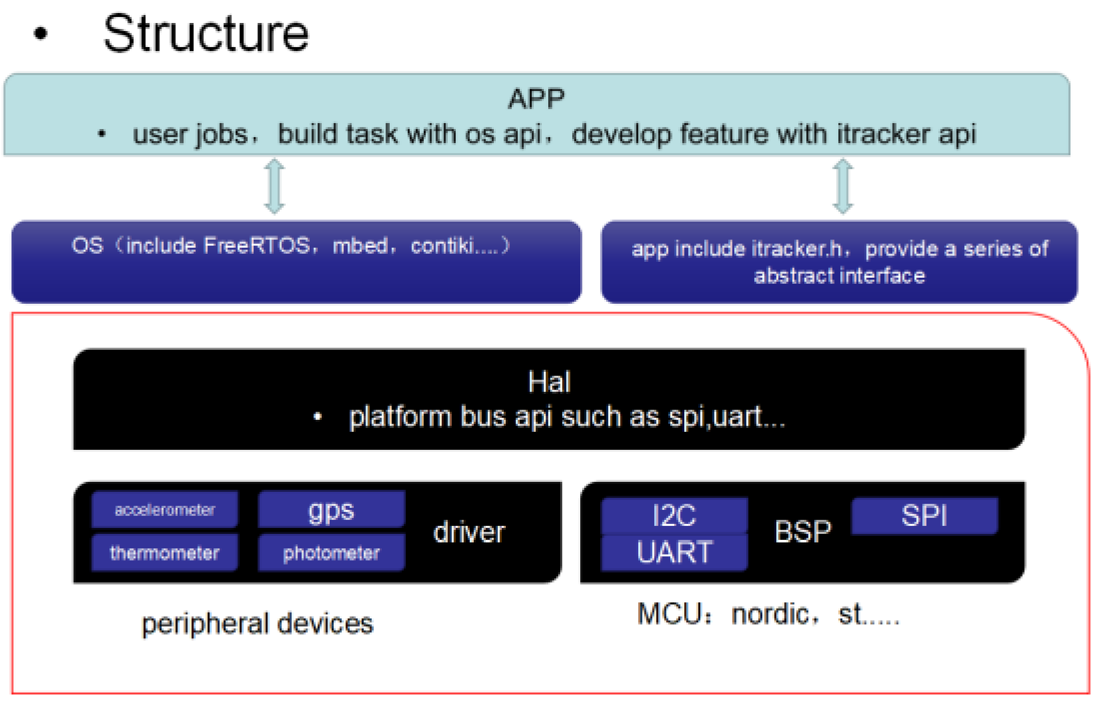
RUI_Platform_Firmware_GCC

1.Overview

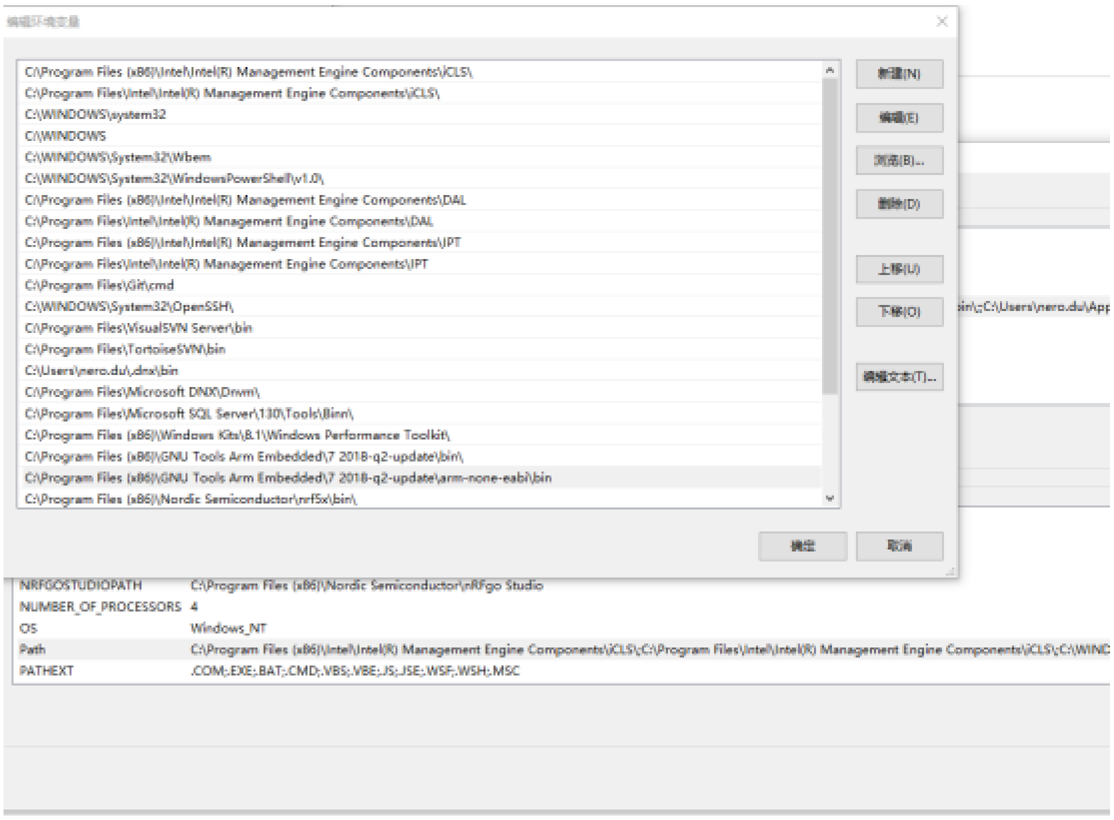
RUI is an open platform running with OS(freertos, mbed...) based on nordic low power chip, including source code and compiler with gcc, designed by Rakwirelss. Besides, RUI not only provides corresponding code suit for Rakwireless hardware board, but also supports user-defined board, which is the most attractive feature. A sensor drivers libraries is supplied and user just chooses the sensor they need and changes the hardware pin parameter in header file. Now RUI just includes nRF52832 and nRF52840, more chips will be added gradually. Also RUI supports gcc compile in windows, linux, OS X, based on NRF52 SDK.


Structure
APP
user jobs，build task with os api，develop feature with itracker api
OS（include FreeRTOS，mbed，contiki....）
app include itracker.h，provide a series of abstract interface
Hal
platform bus api such as spi,uart...
accelerometer
gps
thermometer
photometer
driver
I2C
UART
BSP
SPI
peripheral devices
MCU：nordic，st.....

2.Install Toolchain

2.1 Windows

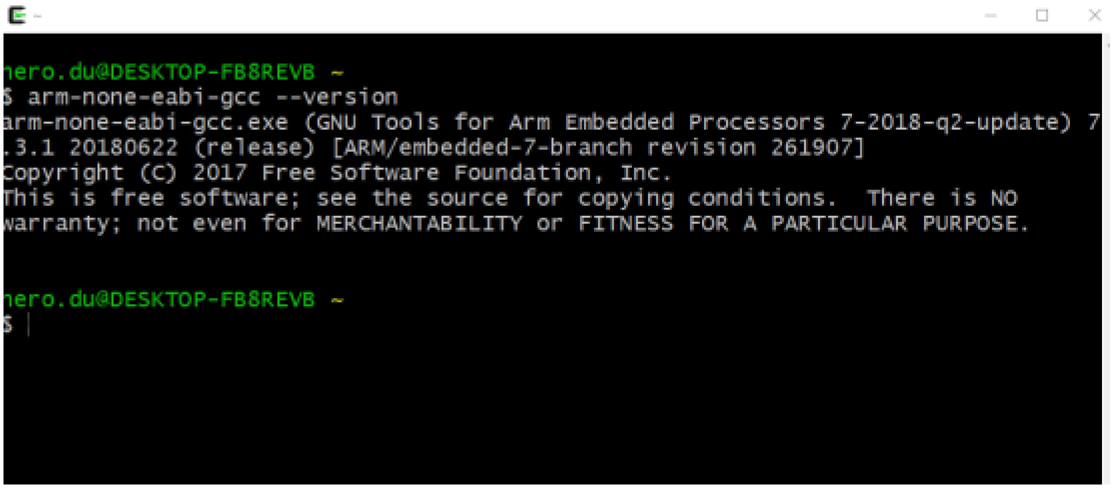
If compile in windows, it is necessary to install gcc compiler environment. We recommend to use Cygwin64 Terminal because of the shell command compatibility with linux (download website: https://cygwin.com/install.html). Besides, the toolchain which nordic recommends with is GNU ARM Eclipse Windows Build Tools (download website: http://gnuarmeclipse.github.io/windows-build-tools/). After installation, remember to add the path of your toolchain to your OS PATH environment variable (path to install directory/GNU Tools ARM Embedded/4.9 2015q3/bin/). Adding the path makes it possible to run the toolchain executables from any directory using the terminal.



```
C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\iCLS\
C:\Program Files\Intel\Intel(R) Management Engine Components\iCLS\
C:\WINDOWS\system32
C:\WINDOWS
C:\WINDOWS\System32\Wbem
C:\WINDOWS\System32\WindowsPowerShell\v1.0\
C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\DAL
C:\Program Files\Intel\Intel(R) Management Engine Components\DAL
C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\IPT
C:\Program Files\Intel\Intel(R) Management Engine Components\IPT
C:\Program Files\Git\cmd
C:\WINDOWS\System32\OpenSSH\
C:\Program Files\VisualSVN Server\bin
C:\Program Files\TortoiseSVN\bin
C:\Users\nero.du\.dnx\bin
C:\Program Files\Microsoft DNX\Dnvm\
C:\Program Files\Microsoft SQL Server\130\Tools\Binn\
C:\Program Files (x86)\Windows Kits\8.1\Windows Performance Toolkit\
C:\Program Files (x86)\GNU Tools Arm Embedded\7 2018-q2-update\bin\
C:\Program Files (x86)\GNU Tools Arm Embedded\7 2018-q2-update\arm-none-eabi\bin
C:\Program Files (x86)\Nordic Semiconductor\nrf5x\bin\
```

```
bin\;C:\Users\nero.du\App
NRFGOSTUDIOPATH        C:\Program Files (x86)\Nordic Semiconductor\nRFgo Studio
NUMBER_OF_PROCESSORS   4
OS                     Windows_NT
Path                   C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\iCLS\;C:\Program Files\Intel\Intel(R) Management Engine Components\iCLS\;C:\WINC
PATHEXT                .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
```

To verify that path is set correctly, type the following in your terminal: arm-none-eabi-gcc --version. This will return the version of the C compiler if the executable is found in your path.



```
nero.du@DESKTOP-FB8REVB ~
$ arm-none-eabi-gcc --version
arm-none-eabi-gcc.exe (GNU Tools for Arm Embedded Processors 7-2018-q2-update) 7
.3.1 20180622 (release) [ARM/embedded-7-branch revision 261907]
Copyright (C) 2017 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.


nero.du@DESKTOP-FB8REVB ~
$ |
```

To build an example in the SDK you first need to set the toolchain path in makefile.windows or makefile.posix depending on platform you are using. The makefile.posix, should be edited if your are working on either Linux or OS X. These files are located in: SDK/components/toolchain/gcc/Makefile.posix(or

Makefile.windows). Open the file in a text editor, and make sure that the GNU_INSTALL_ROOT variable is pointing to your Gnu tools for ARM embedded Processors install directory like below(modify the file just for your OS):

```
GNU_INSTALL_ROOT := C:/Program Files (x86)/GNU Tools Arm Embedded/7 2018-q2-update/bin/
GNU_VERSION := 7.3.1
GNU_PREFIX := arm-none-eabi
```

2.2 Linux and OS X

Linux and OS X already have the necessary shell commands, but GNU make may not be a part of the standard distro. Call "make -v" from the terminal to check whether it is installed or not. GNU make would need to be installed if it's not recognized as a command. GNU make is bundled with Xcode tools if working on OS X. On Linux it may be different ways to obtain GNU make depending on your distro, if not installed already. On Ubuntu you can get by entering this command: sudo apt-get install build-essential check install.

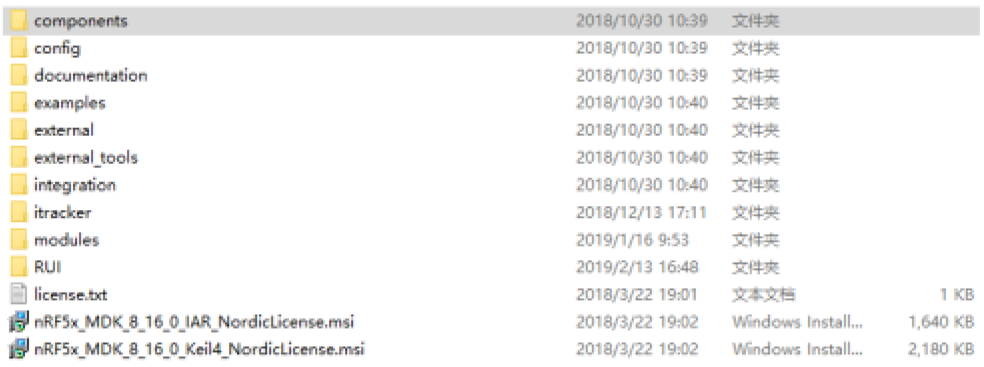
3. Download SDK

RUI supports nRF52840 (sdk version:15.2.0) and nRF52830 (sdk version:15.0.0) for now, so choose the sdk according to your chip. You can download SDK from the way below:

http://developer.nordicsemi.com/nRF5_SDK/nRF5_SDK_v15.x.x/

4. Compile

OK, it is ready to compile!
1. Put RUI to the root directory of sdk.

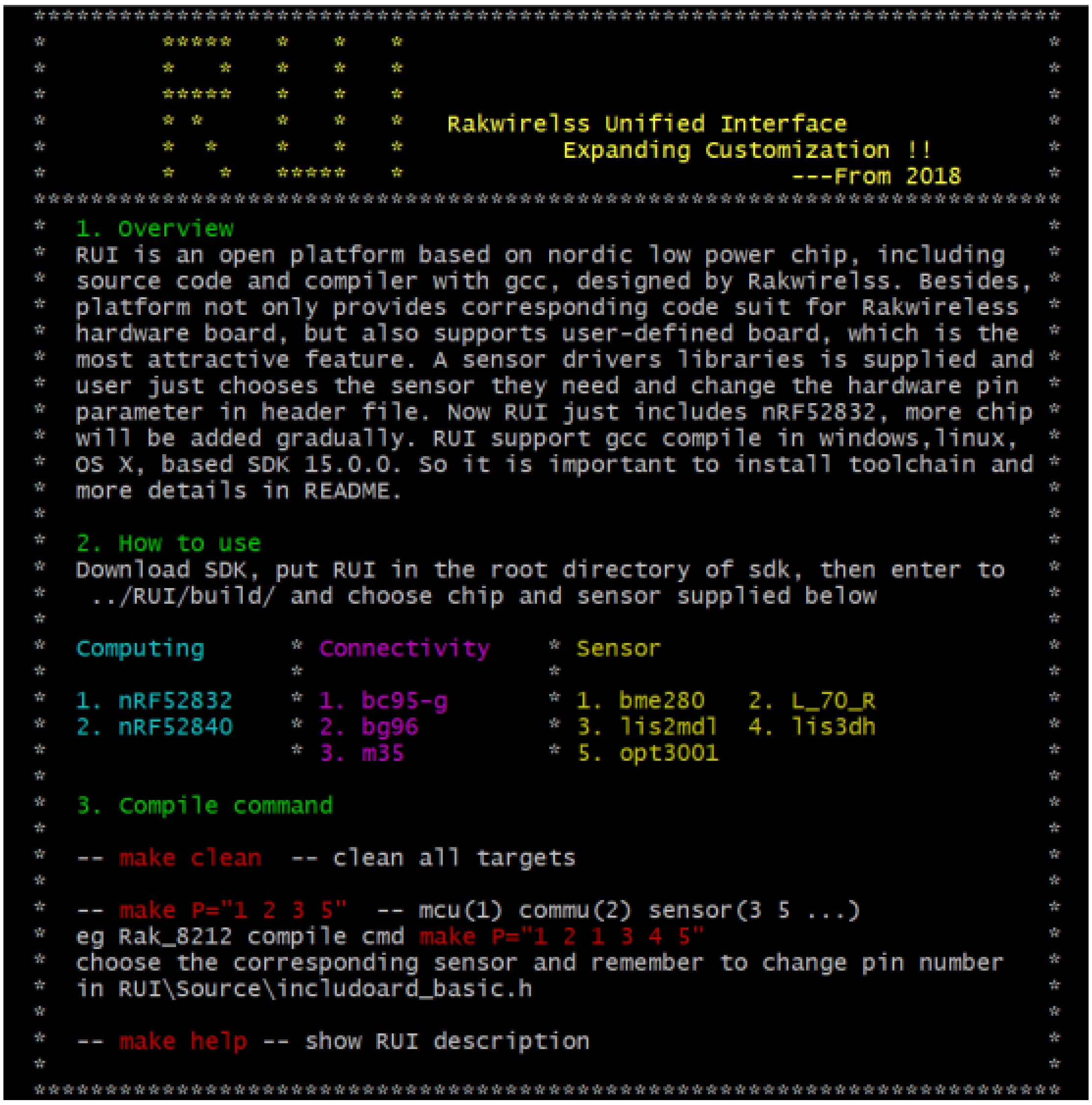


components	2018/10/30 10:39	文件夹	
config	2018/10/30 10:39	文件夹	
documentation	2018/10/30 10:39	文件夹	
examples	2018/10/30 10:40	文件夹	
external	2018/10/30 10:40	文件夹	
external_tools	2018/10/30 10:40	文件夹	
integration	2018/10/30 10:40	文件夹	
itracker	2018/12/13 17:11	文件夹	
modules	2019/1/16 9:53	文件夹	
RUI	2019/2/13 16:48	文件夹	
license.txt	2018/3/22 19:01	文本文档	1 KB
nRF5x_MDK_8_16_0_IAR_NordicLicense.msi	2018/3/22 19:02	Windows Install...	1,640 KB
nRF5x_MDK_8_16_0_Keil4_NordicLicense.msi	2018/3/22 19:02	Windows Install...	2,180 KB

2. Enter to ../RUI/build/ and execute: make help. The details of RUI will show and you can choose to compile. According to the architecture, RUI contains three parts:

- ✧ Computing, mcu including nRF52832 and nRF52840 now
- ✧ Connectivity, NB-IOT moudle like bg96, m35...
- ✧ Sensor, peripheral like accelerometer, gps, thermometer...

With the purpose of decoupling, RUI has a more expansibility and not just limited in the hardware we supply. User can design their own board and run RUI on it just change the hardware parameter like pin number in head file.



```
************************************************************************
*        *****    *    *    *                                          *
*        *   *    *    *    *                                          *
*        *****    *    *    *                                          *
*        * *      *    *    *   Rakwirelss Unified Interface           *
*        *  *     *    *    *           Expanding Customization !!     *
*        *   *    *****     *                           ---From 2018   *
************************************************************************
*  1. Overview                                                         *
*  RUI is an open platform based on nordic low power chip, including   *
*  source code and compiler with gcc, designed by Rakwirelss. Besides, *
*  platform not only provides corresponding code suit for Rakwireless  *
*  hardware board, but also supports user-defined board, which is the  *
*  most attractive feature. A sensor drivers libraries is supplied and *
*  user just chooses the sensor they need and change the hardware pin  *
*  parameter in header file. Now RUI just includes nRF52832, more chip *
*  will be added gradually. RUI support gcc compile in windows,linux,  *
*  OS X, based SDK 15.0.0. So it is important to install toolchain and *
*  more details in README.                                             *
*                                                                      *
*  2. How to use                                                       *
*  Download SDK, put RUI in the root directory of sdk, then enter to   *
*   ../RUI/build/ and choose chip and sensor supplied below            *
*                                                                      *
*  Computing        * Connectivity     * Sensor                        *
*                   *                  *                               *
*  1. nRF52832      * 1. bc95-g        * 1. bme280    2. L_70_R        *
*  2. nRF52840      * 2. bg96          * 3. lis2mdl   4. lis3dh        *
*                   * 3. m35           * 5. opt3001                    *
*                                                                      *
*  3. Compile command                                                  *
*                                                                      *
*  -- make clean  -- clean all targets                                 *
*                                                                      *
*  -- make P="1 2 3 5"  -- mcu(1) commu(2) sensor(3 5 ...)             *
*  eg Rak_8212 compile cmd make P="1 2 1 3 4 5"                        *
*  choose the corresponding sensor and remember to change pin number   *
*  in RUI\Source\includoard_basic.h                                    *
*                                                                      *
*  -- make help -- show RUI description                                *
*                                                                      *
************************************************************************
```

3. Use command make P="1 2 1 3 4 5" to compile. It is consist of three parts too. For example, we compile itracker_8212 which includes:

nRF52832+bg96+bme280+lis2mdl+lis3dh+opt3001

And the command should be like:

$$\text{make P="1}\quad 2\quad \underbrace{1\quad 3\quad 4\quad 5\text{"}}_{\text{sensor}}$$

mcu + moudle +

You can just choose what you want on your board because of the complete decoupling.

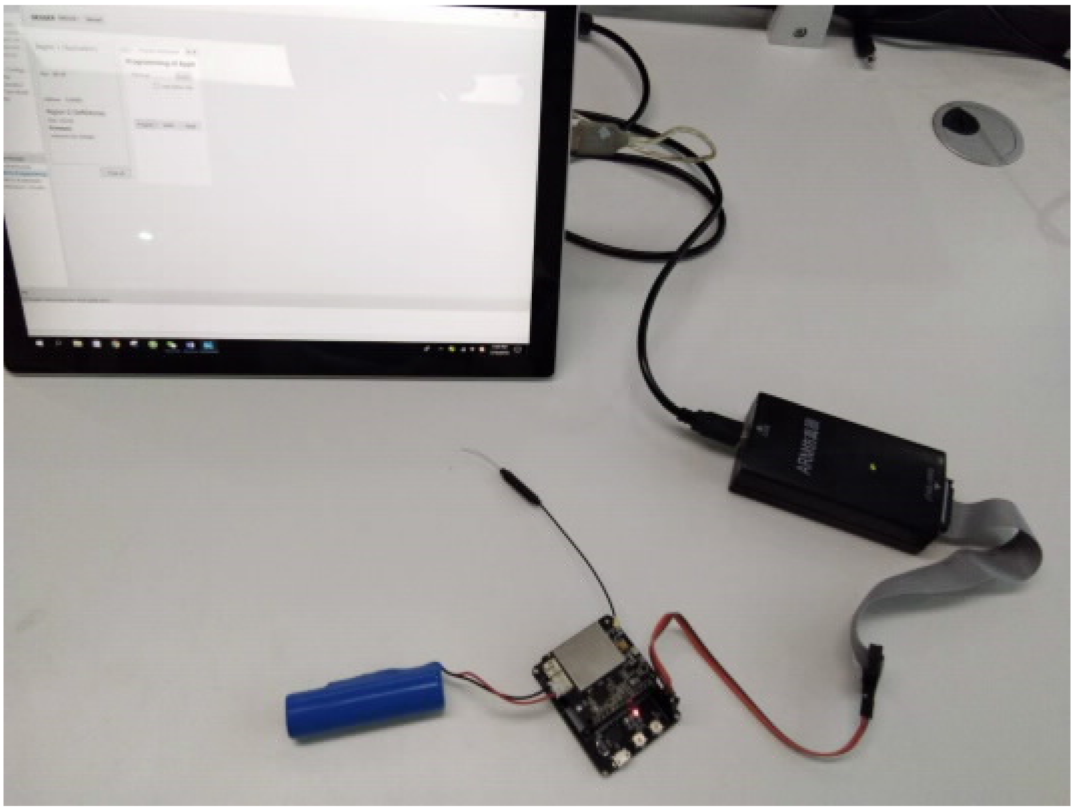
5.Directory

The directory of RUI is like below:

- ✧ Source: source code include app, driver, service, hal, external, include.....
- ✧ build: gcc compile script
- ✧ tools: relevant nordic tools to debug and download

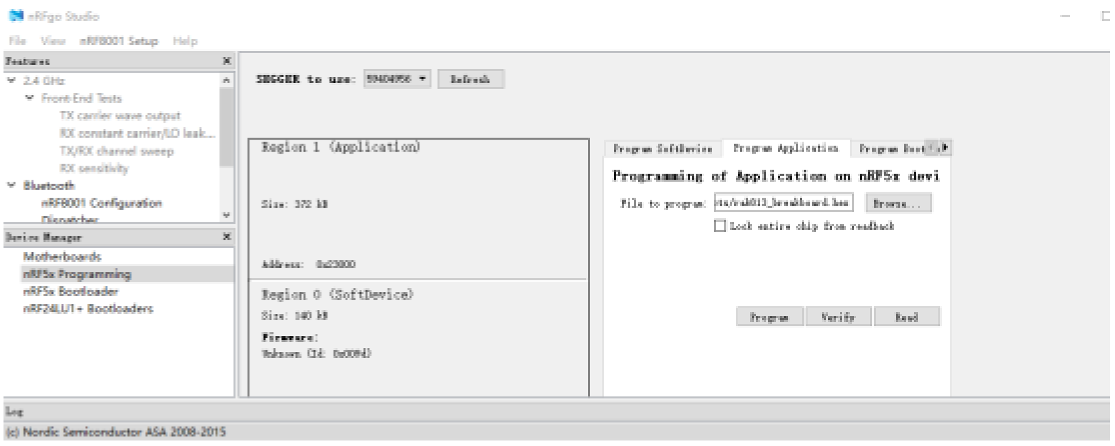
6.Burn

After compiling, app hex will be created in SDK/RUI/build/_build/nrf52_xxaa.hex. We use nordic tools to download the file, but remember to install the relevant tools for the incompatible of nRF52832 and nRF52840. Because nordic will use nRF connect to manage all ble products later. The burn tool is J-link and moudle should supply power like below:



- ✧ nRF52832

Install nrfgostudio_win-64_1.21.2_installer.msi in SDK/RUI/tools/ and it like this:

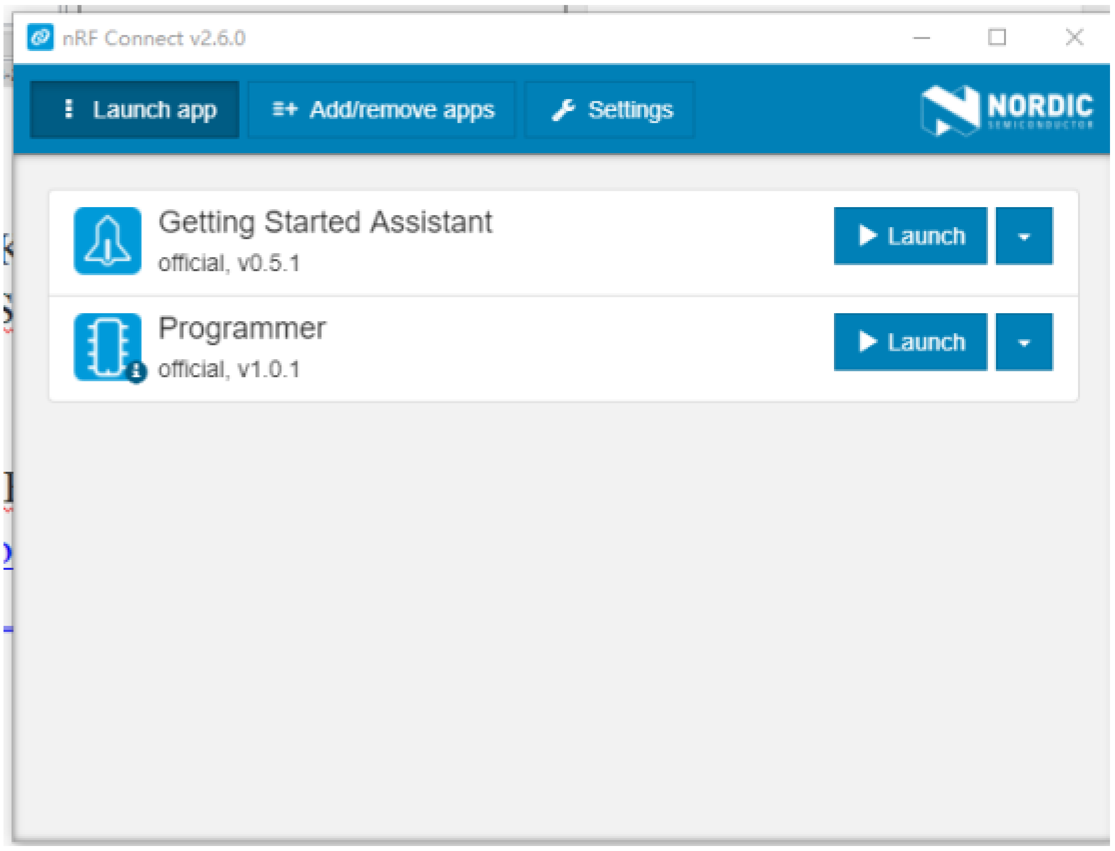


First burn SDK hex(SDK/RUI/build/hex/s132_nrf52_6.0.0_softdevice.hex) to board with Program SoftDevice. Second burn app hex with Program Application.

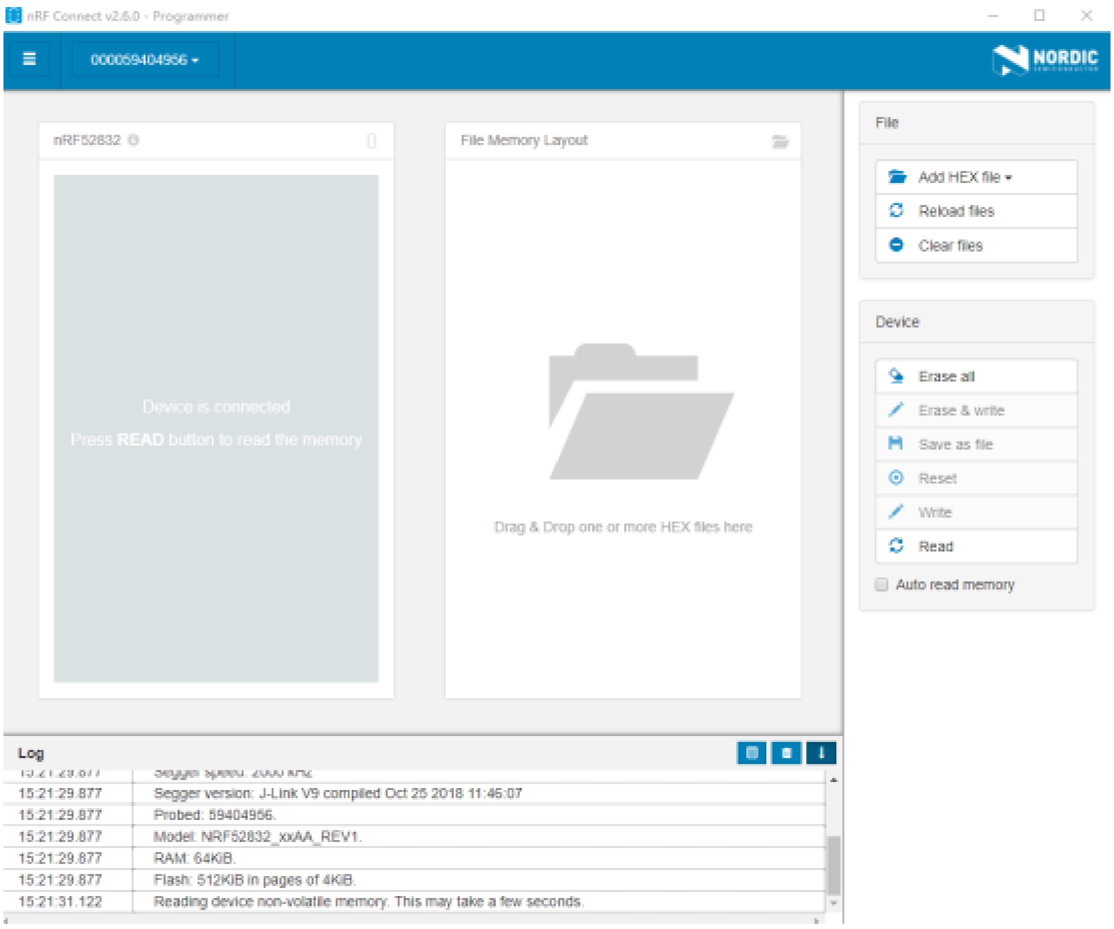
✧ nRF52840

Download nRF Connect for desktop and nRF5x Command Line Tools from https://www.nordicsemi.com/DocLib/Content/User_Guides/nrf52840_dk/latest/UG/common/nordic_tools#nordic_tools

Choose launch app and launch Programmer like below.



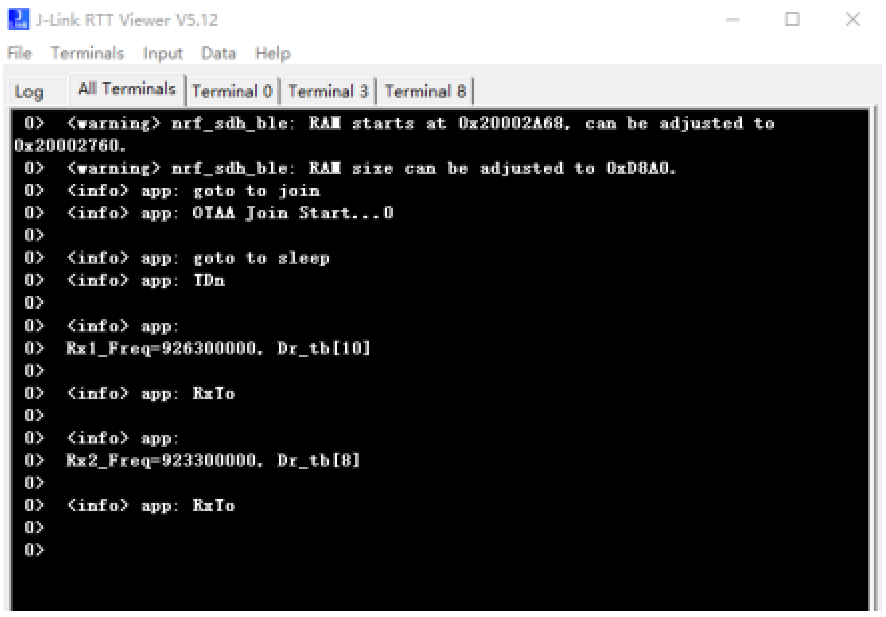
Click Select device, erase all first. Then Add HEX file and choose sdk hex and app hex, Write, done!



```
15:21:29.877   Segger speed: 2000 kHz
15:21:29.877   Segger version: J-Link V9 compiled Oct 25 2018 11:46:07
15:21:29.877   Probed: 59404956.
15:21:29.877   Model: NRF52832_xxAA_REV1.
15:21:29.877   RAM: 64KiB.
15:21:29.877   Flash: 512KiB in pages of 4KiB.
15:21:31.122   Reading device non-volatile memory. This may take a few seconds.
```

7.Debug

When install the tools, Segger J-link driver is setup meanwhile. Log will be show with Jlink RTT viewer in All Terminals.



```
 0> <warning> nrf_sdh_ble: RAM starts at 0x20002A68, can be adjusted to
0x20002760.
 0> <warning> nrf_sdh_ble: RAM size can be adjusted to 0xD8A0.
 0> <info> app: goto to join
 0> <info> app: OTAA Join Start...0
 0>
 0> <info> app: goto to sleep
 0> <info> app: TDn
 0>
 0> <info> app:
 0> Rx1_Freq=926300000, Dr_tb[10]
 0>
 0> <info> app: RxTo
 0>
 0> <info> app:
 0> Rx2_Freq=923300000, Dr_tb[8]
 0>
 0> <info> app: RxTo
 0>
 0>
```

8. Power

RUI has added deep sleep mode which mcu enters system off mode and peripheral like gsm, gps turns off except acc. Because acc will wake up mcu and reset all. Remember to turn off debug way like log (NRF_LOG_ENABLE in sdk_config.h), which will prevent mcu to enter system off mode. The way to wake up moudle is to waggle moudle for accelerated interrupt. The consumption in deep sleep mode is near to 0.96mA~1.5mA according to the specific moudle.