

nRF5 Series: Developing with SEGGER Embedded Studio

Getting Started Guide

v1.1

Contents

| | |
|--|-----------|
| Revision history | iii |
| 1 Introduction | 4 |
| 2 Minimum requirements | 5 |
| 3 Related documentation | 6 |
| 4 Development kits, boards, and chips | 7 |
| 5 SoftDevices | 8 |
| 6 Running a first test | 9 |
| 7 Setting up your toolchain | 10 |
| 7.1 Nordic tools and downloads | 10 |
| 7.2 Setting up the nRF5 SDK | 13 |
| 7.3 Installing SEGGER tools | 13 |
| 7.4 Installing the nRF5x Command Line Tools | 14 |
| 8 Programming an application | 15 |
| 8.1 Erasing the board | 15 |
| 8.2 Importing Keil projects | 16 |
| 8.3 Compiling the application | 20 |
| 8.4 Configuring placement of the SoftDevice | 21 |
| 8.5 Programming the firmware | 24 |
| 8.6 Adding files | 24 |
| 8.6.1 Adding source files | 24 |
| 8.6.2 Including header files | 25 |
| 9 Communicating with the board | 27 |
| 9.1 Connecting via RTT | 27 |
| 9.1.1 Connecting via RTT on Windows | 27 |
| 9.1.2 Connecting via RTT on Linux | 28 |
| 9.2 Connecting via CDC-UART | 28 |
| 10 Testing the application | 30 |
| 10.1 Testing with a mobile device | 30 |
| 10.2 Testing with a computer | 31 |
| 11 Debugging | 33 |
| Glossary | 34 |
| Acronyms and abbreviations | 35 |
| Legal notices | 36 |

Revision history

| Date | Version | Description |
|--------------|---------|---|
| October 2018 | 1.1 | Added Adding files on page 24 |
| July 2018 | 1.0 | First release |

1 Introduction

This guide will help you get started with your nRF5 Series *Development Kit (DK)* and developing your application with *SEGGER Embedded Studio (SES)*.

If you have worked with any of Nordic Semiconductor's products before, you are probably familiar with the required software tools. In this case, this guide will mostly provide reference information.

If this is your first time developing software for a Nordic Semiconductor *System on Chip (SoC)*, this guide will help you set up your development toolchain and guide you through all the steps that are necessary to develop, program, test, and debug your application.

This guide describes how to work with *SES*. *SES* is a cross-platform *Integrated Development Environment (IDE)*, so you can run it on different operating systems. For use with Nordic Semiconductor devices, you can get a free license that has no limitations.

There are two Getting Started Guides that show how to work with different *IDEs*:

- [nRF5 Series: Developing with SEGGER Embedded Studio](#) (this document)
- [nRF5 Series: Developing on Windows with ARM Keil MDK](#)

Check out the [Nordic DevZone](#) for additional setup information and help.

2 Minimum requirements

Ensure that you have all the required hardware and that your computer fulfills the software requirements.

Hardware requirements

- One of the following nRF5 Series development kits:
 - nRF52840 *DK*
 - nRF52 *DK*
 - nRF51 *DK*
- Micro-USB 2.0 cable
- Personal computer (PC) or Mac
- Smartphone or tablet that supports *Bluetooth*® Low Energy
- Optional: nRF52840 Dongle or nRF51 Dongle

Software requirements

One of the following operating systems:

- Windows 7, Windows 8, or Windows 10
- macOS
- Linux

3 Related documentation

This guide is a starting point and gives essential information for getting started with software development on nRF5 Series devices. For more detailed information, check the documentation of the separate components and tools.

Development Kit User Guides

[nRF52840 Development Kit](#)

[nRF52 Development Kit](#)

[nRF51 Development Kit](#)

Dongle User Guides

[nRF52840 Dongle](#)

[nRF51 Dongle](#)

Compatibility Matrices

[nRF52840 Compatibility Matrix](#)

[nRF52832 Compatibility Matrix](#)

[nRF52810 Compatibility Matrix](#)

[nRF51 Series Compatibility Matrix](#)

nRF5 SDK documentation

[nRF5 SDK](#)

Tools User Guides

[nRF5x Command Line Tools](#)

[nRF Connect Bluetooth Low Energy](#)

4 Development kits, boards, and chips

Each nRF5 Series [DK](#) contains a board, which in turn contains a chip. Some of Nordic Semiconductor's tools target the chip, while others target the development board or the development kit. Therefore, you must be aware of the relation between the different components.

You can find all compatibility information in the [compatibility matrix](#) for each chip. The following table summarizes the information that you must be aware of for programming your development board.

| Development kit | Development board | Chip |
|-----------------------------|-------------------|---|
| nRF52840 DK | PCA10056 | nRF52840 |
| nRF52840 Dongle | PCA10059 | nRF52840 |
| nRF52 DK | PCA10040 | nRF52832 Note: This kit is also used for nRF52810 development. See nRF52810 Development Kit for more information. |
| nRF51 DK | PCA10028 | nRF51422 |
| nRF51 Dongle | PCA10032 | nRF51422 |

Table 1: Relation between development kits, boards, and chips

5 SoftDevices

A *SoftDevice* is a wireless protocol stack that complements an nRF5 Series *SoC*. Nordic Semiconductor provides them as qualified, precompiled binary files. While it is possible to build applications without using a *SoftDevice*, all nRF5 SDK example applications that use Bluetooth Low Energy or ANT™ require a *SoftDevice*.

See the [compatibility matrices](#) for detailed information about which *SoftDevice* versions are supported for each chip. The following table summarizes the usage scenarios for each *SoftDevice*.

| Protocol | Role | Chip | SoftDevice |
|------------------------------|------------------------|--|------------|
| Bluetooth Low Energy | Peripheral | <ul style="list-style-type: none"> nRF51422 nRF51822 | S110 |
| | | <ul style="list-style-type: none"> nRF52810 nRF52832 | S112 |
| | Central or Peripheral | <ul style="list-style-type: none"> nRF51422 nRF51822 | S120 |
| | Central and Peripheral | <ul style="list-style-type: none"> nRF51422 nRF51822 | S130 |
| | | <ul style="list-style-type: none"> nRF52832 | S132 |
| | | <ul style="list-style-type: none"> nRF52840 | S140 |
| ANT | | <ul style="list-style-type: none"> nRF51422 | S210 |
| | | <ul style="list-style-type: none"> nRF52832 | S212 |
| Bluetooth Low Energy and ANT | Peripheral | <ul style="list-style-type: none"> nRF51422 | S310 |
| | Central and Peripheral | <ul style="list-style-type: none"> nRF52832 | S332 |

Table 2: *SoftDevice* overview

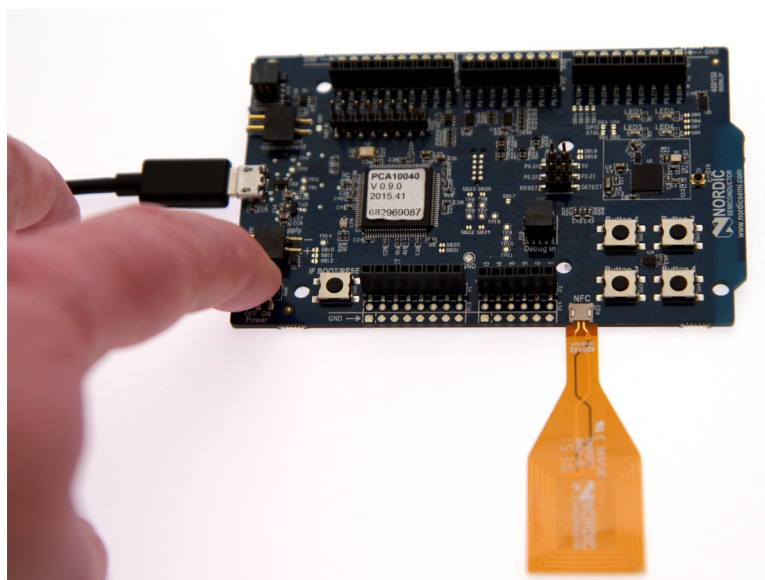
6 Running a first test

Before you start developing, program and run a simple application on your development board to ensure that the board functions as expected and the communication between your computer and your development board works.

Before you begin, download the latest compatible version of the [nRF5 SDK](#). For information about which SDK supports which IC revisions, check the [compatibility matrices](#).

The nRF5 SDK contains precompiled HEX files of the most common examples. Extract the zip file into a folder of your choosing.

1. Power up the nRF5 Series development board:
 - a) Connect one end of a micro-USB 2.0 cable to the USB connector on the board and the other end to one of your PC's USB host ports.
 - b) Slide the **power switch** to "ON".



Observe that **LED1** starts blinking.

2. Open a file explorer and confirm that the development board has appeared as a removable drive named "JLINK".
3. In the folder where you extracted the nRF5 SDK, navigate to `examples\peripheral\blinky\hex`.
4. Select the HEX file that corresponds to your development board and copy it to the JLINK drive. The development board will now restart and run the application.

If the LEDs on the board start blinking, you have successfully programmed your first application! Next, continue to set up your development toolchain and program a more advanced application.

7 Setting up your toolchain

Before you can start developing, you must install the required software. This software includes tools to connect to your development board, an *IDE* for developing your application, and the nRF5 SDK that provides libraries and example applications.

See [Nordic tools and downloads](#) on page 10 for an overview of available tools and the links to download the latest versions for your operating system.

The following tools are required:

- [nRF5 SDK](#)
- [SEGGER Embedded Studio \(SES\)](#)
- [SEGGER J-Link Software and Documentation Pack](#)

The following tools are optional:

- nRF5x Command Line Tools for [Windows](#), [Linux32](#), [Linux64](#), or [macOS](#) (all including nrfjprog)

See the following sections for installation instructions.

7.1 Nordic tools and downloads

This overview lists all available Nordic Semiconductor tools and supported *IDEs*. Not all of these tools are required. To help you pick the *IDE* and tools you want to use, see the following sections for common setup scenarios.

Development IDE

Pick one of the IDEs with a compiler supported by Nordic:

| IDE | Windows | Linux | OSX |
|---|---------|-------|-----|
| SEGGER Embedded Studio (SES) | Yes | Yes | Yes |
| MDK-ARM Keil μVision | Yes | No | No |
| GNU/GCC | Yes | Yes | Yes |
| IAR | Yes | No | No |

SES is the recommended platform. It is free for use with nRF devices.

Essential tools

You need to download these Nordic tools to develop with our devices.

| Tool | Description | Download | Documentation | Protocol |
|-----------------------------------|---|--|---|--|
| SDK (Software Development Kit) | Application examples, source files, SoftDevices | Windows/Linux | nRF5 SDK nRF5 SDK for Mesh nRF5 SDK for Thread and Zigbee | BLE/ANT Bluetooth Mesh Thread and Zigbee |
| nRF5x Command Line Tools | Collection of command line tools, like nrfjprog, mergehex | nRF5x Tools Windows32 nRF5x Tools Linux32 nRF5x Tools Linux64 nRF5x Tools OSX | nRF5x Command Line Tools | BLE/ANT |

Optional tools

These tools are not essential, but we recommend that you use them.

| Tool | Description | Download | Documentation | Protocol |
|---|---|---|--|----------|
| SoftDevice | Wireless protocol stack | Click the Download tab on: nRF52840 Product page nRF52832 Product page nRF52810 Product page nRF51822 Product page nRF51422 Product page | nRF52 Series SDS nRF51 Series SDS | BLE/ANT |
| nRF Connect for Desktop | Expandable desktop tool with several apps, including: <ul style="list-style-type: none"> • Peer device emulator • Power Profiler • Programmer • Cloud Gateway | nRF Connect for OS-X nRF Connect for Ubuntu Linux nRF Connect for Windows | nRF Connect Bluetooth Low Energy | BLE |
| nRF Connect for Mobile | Peer device emulator app for smartphones | Android v4.3 or later IOS v8 or later | | BLE |
| Nordic nRF Toolbox app | App that contains all the Nordic apps | Android v4.3 or later IOS v8 or later Windows Phone v8.1 or later | | BLE |
| nRF5x pynrfjprog | Simple Python interface for the nrfjprog DLL | nRF5x pynrfjprog | nRF5x pynrfjprog | BLE/ANT |
| ANTware II | Peer device emulator for the ANT protocol running on computers | ANTware II | ANTware II | ANT |
| nRF Sniffer | App for monitoring on-air traffic | nRF Sniffer download | nRF Sniffer | BLE |
| nRF Thread Topology Monitor | Tool for visualizing Thread mesh network topology in real time | nRF Thread Topology Monitor for Ubuntu Linux 64-bit nRF Thread Topology Monitor for Windows | nRF Thread Topology Monitor | Thread |
| Thread Border Router | Gateway for connecting Thread | Thread Border Router | Thread Border Router | Thread |

| Tool | Description | Download | Documentation | Protocol |
|------|-------------------------|----------|---------------|----------|
| | network to the Internet | | | |

See also [Nordic mobile apps](#) for a list of available Bluetooth Low Energy and Mesh mobile apps for iOS, Android and Windows Phones.

7.2 Setting up the nRF5 SDK

The nRF5 SDK does not require installation. You only need to download and extract the files.

If you followed the instructions in [Running a first test](#) on page 9, you already downloaded and extracted the nRF5 SDK files and are all set up.

To set up your SDK environment:

1. Download the [nRF5 SDK](#) zip file.

If you have an nRF52 device, select the latest version. For nRF51 devices, select the latest version with support for nRF51 (currently, v12.3.0). For information about which SDK supports which IC revisions, check the [compatibility matrices](#).

2. Extract the zip file to the directory that you want to use to work with the SDK.

This folder will be referred to as *SDK_dir* in the following documentation.

Note: Compilers tend to run into problems with long path names. Therefore, place the folder as close to the root level of your file system as possible (for example, at `C:/Nordic/SDK`). Also, avoid using spaces in the file path and folder name.

7.3 Installing SEGGER tools

Download and install the most recent releases of [SES](#) and the J-Link Software and Documentation Pack.

1. Download the software packages for your operating system from [SEGGER downloads](#).

You need the following packages:

- Embedded Studio for ARM (version 3.30 or later)
- J-Link Software and Documentation Pack (version 6.10g or later)

2. Install both packages.

3. Obtain and activate your free license for [SES](#):

- a) Open [SES](#).

[SES](#) will automatically load a test project.

- b) Click **Build > Build and Debug**.

A window asking for a license will pop up. [SES](#) is free of charge for use with Nordic Semiconductor devices, but you still need to request and activate a license.

- c) Select **Activate Your Free License** and fill in your information to request a license.

Note: In [SES](#) versions before 3.34, this option was called **Get a Free License**.

The license is sent to you in an email.

- d) After you receive your license key, enter it to activate the license.

7.4 Installing the nRF5x Command Line Tools

The nRF5x Command Line Tools are used for developing, programming, and debugging of Nordic Semiconductor's nRF5x SoCs (System on Chip).

When installing on macOS or Linux, the SEGGER software must be installed in its default location, or the shared library must be placed so that `dlopen()` can find it. The default location is:

- On macOS: `/Applications/SEGGER/JLink`
- On Linux: `/opt/SEGGER/JLink`

The SEGGER software can be installed by downloading and running the installer from [SEGGER Software](#).

When installing the nRF5x Command Line Tools on Windows, SEGGER software is automatically installed in addition to the tools.

Complete the following steps to install the nRF5x Command Line Tools:

1. Download the software for your operating system:

- Windows: [nRF5x-Command-Line-Tools for Win32](#)
- Linux 32-bit: [nRF5x-Command-Line-Tools-Linux-i386](#)
- Linux 64-bit: [nRF5x-Command-Line-Tools-Linux-x86_64](#)
- macOS: [nRF5x-Command-Line-Tools for Os X](#)

2. Install the software by running the installer and following the given instructions, or by extracting the `.tar` archive anywhere on your file system and adding the extracted folders to your path.

3. Enter the following command in a command line to make sure that `nrfjprog` is installed correctly:

```
nrfjprog --version
```

If you get an error message that the command cannot be found, `nrfjprog` must be manually added to the PATH. On Windows:

- Go to the Windows **Advanced system settings** and click **Environment Variables**.
- Select the Path variable and click **Edit**.
- Add the following text at the end of the variable value: `;C:\Program Files (x86)\Nordic Semiconductor\nrf5x\bin`
Make sure that you add a semicolon (;) between entries in the PATH values: `path1;path2`
- Click **OK** twice.

On Linux, assuming that you have extracted the `.tar` archive into `/opt/nrfjprog`:

- Add the following command to the configuration file for your command line, for example, to `~/.bashrc`:

```
export PATH=$PATH:/opt/nrfjprog
```

Open a new command prompt and repeat the command. It should now succeed.

8 Programming an application

After setting up the required toolchain, you are ready to compile your application and program (or "flash") it to your development board.

Starting with v14.1.0, the nRF5 SDK supplies SEGGER Embedded Studio projects. If you are using an older version of the nRF5 SDK (for example, nRF5 SDK v12.3.0, which supports nRF51 Series devices), you must import and convert the Keil μ Vision projects.

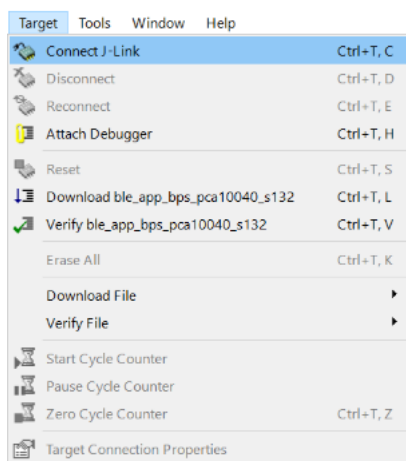
There is a series of video tutorials that show how to get started with SEGGER Embedded Studio in combination with the nRF5 SDK. Check them out here: [Getting started with SEGGER Embedded Studio and the nRF5 SDK](#)

8.1 Erasing the board

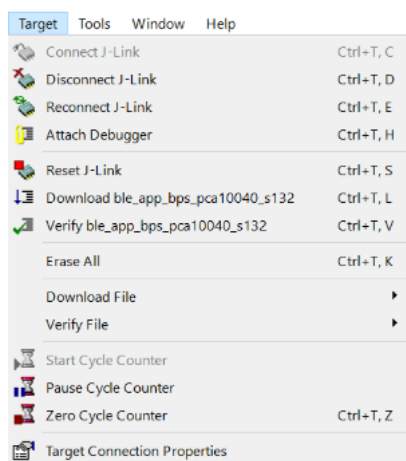
Before you program an example to the development board, you should erase the contents of the board.

There are different ways to erase the board. You can, for example, use [SES](#) or the command line tool nrfjprog (part of the nRF5x Command Line Tools).

- To erase the contents of the board with [SES](#), complete the following steps:
 - a) Select **Target > Connect J-Link**.



- b) After the connection is established, select **Target > Erase All**.



- To erase the contents of the board with nrfjprog, enter the following command:

- For nRF51 devices: `nrfjprog --family nRF51 --eraseall`
- For nRF52 devices: `nrfjprog --family nRF52 --eraseall`

8.2 Importing Keil projects

If you are using a version of the SDK that supports [SES](#), thus nRF5 SDK v14.1.0 or later, you should open one of the supplied [SES](#) projects and skip this step. If you are using an older version of the SDK, for example, nRF5 SDK v12.3.0 for nRF51 Series devices, you must create your own [SES](#) projects based on the supplied Keil projects.

Before you begin, install the CMSIS-CORE Support Package. To do so, open [SES](#), go to **Tools > Package Manager**, and select and install **CMSIS-CORE Support Package**.

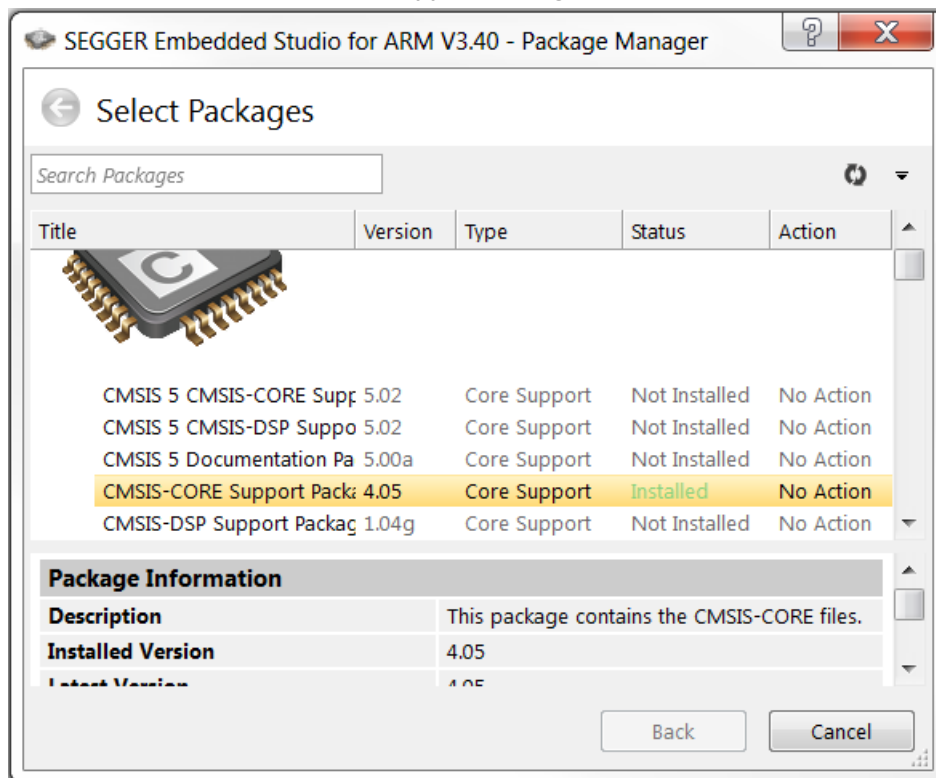
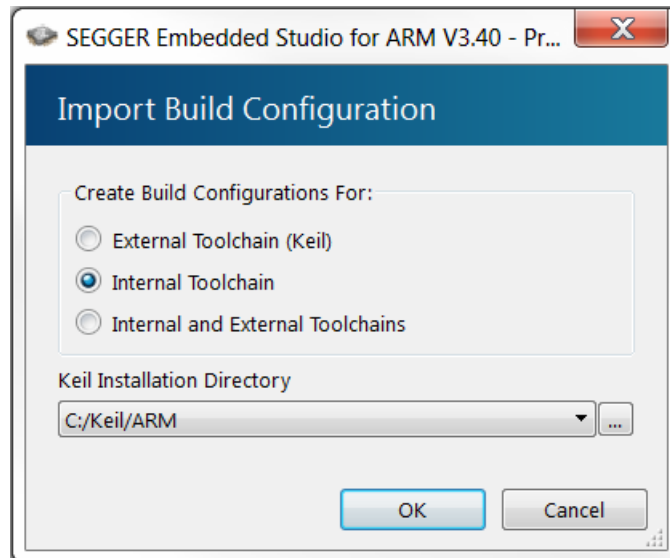


Figure 1: SES Package Manager

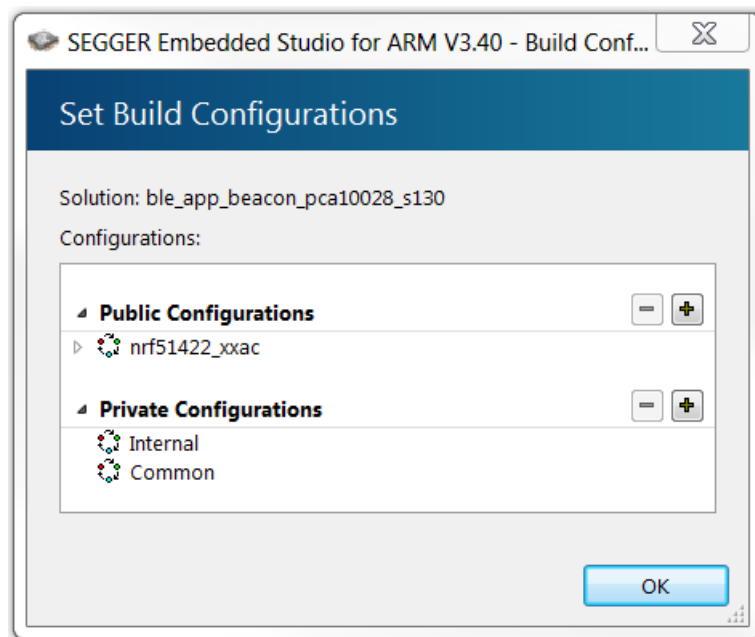
Complete the following steps to import a Keil project into [SES](#):

1. Select **File > Import Project > Import Keil MDK Project**.
In [SES](#) versions before 3.40, this option was located under **File > Import IAR EWARM / Keil MDK Project**.
2. Navigate to the folder that contains the nRF5 SDK and select the example that you want to import.
Use the *.uvprojx file that is located in the arm5_no_packs folder.
3. In the **Import Build Configuration** window that appears, select **Internal Toolchain**.



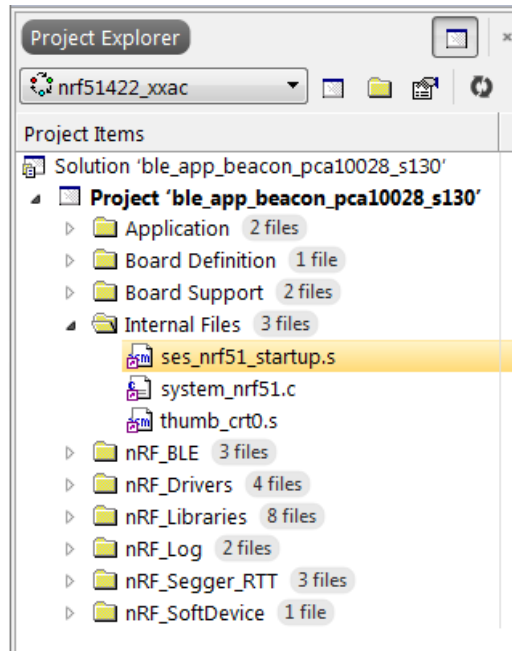
4. If the project that you imported contains a *Target* for flashing the *SoftDevice*, delete this *Target* in *SES*:
 - a) Select **Project > Build Configurations**.
 - b) Under **Public Configurations**, select the *Target* (for example, **flash_s130_nrf51_2.0.1_softdevice**).
 - c) Click the - symbol to remove the *Target*.

The build configuration should now look similar to this:



5. Add the nRF5 MDK files that are required for startup and system setup.
In Keil μ Vision, these files are provided by the nRF_DeviceFamilyPack. Since this pack is not available for *SES*, you must add them manually to the *SES* project.
 - a) Download [ses_nrf51_startup.s](#) and [ses_nrf52_startup.s](#) and save them in a new folder *SDK_dir/components/toolchain/embedded_studio*.
 - b) In the Project Explorer, navigate to **Internal Files** and remove the *Cortex_M_Startup.s* file.
 - c) Right-click **Internal Files** and select **Add Existing File**.
 - d) Browse to the *SDK_dir/components/toolchain/* folder and select the .c file that corresponds to your device (for example, *system_nrf52.c*).
 - e) Right-click **Internal Files** and select **Add Existing File** again. This time, select the startup file that you downloaded, for example, *SDK_dir/components/toolchain/embedded_studio/ses_nrf51_startup.s*.

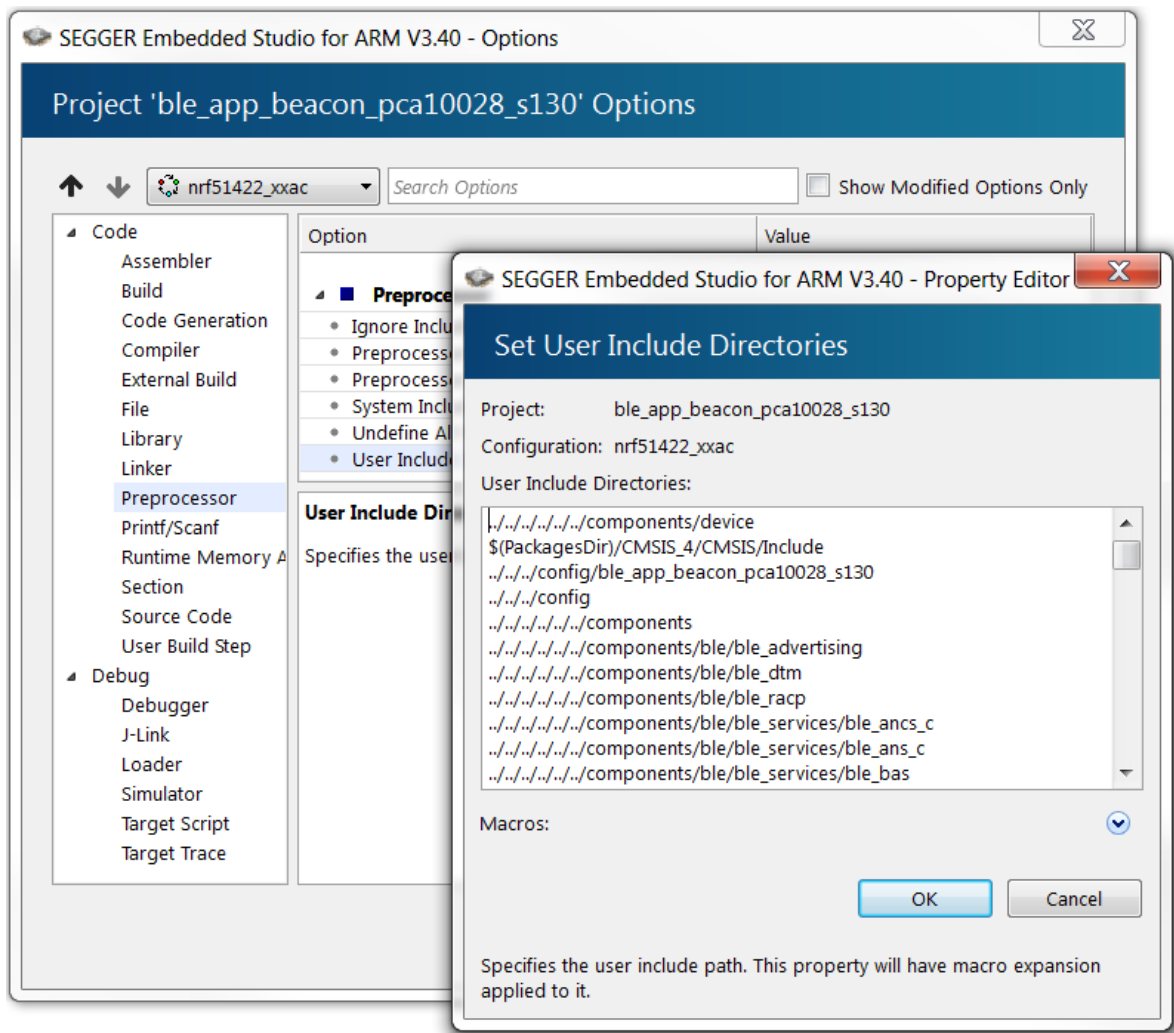
The Project Explorer should now look similar to this:



6. Include the device header files in the project:

- a) Right-click your project and select **Edit Options**.
- b) Select **Preprocessor**.
- c) Add the following path to the User Include Directories field: `../../../../../../../../components/device`

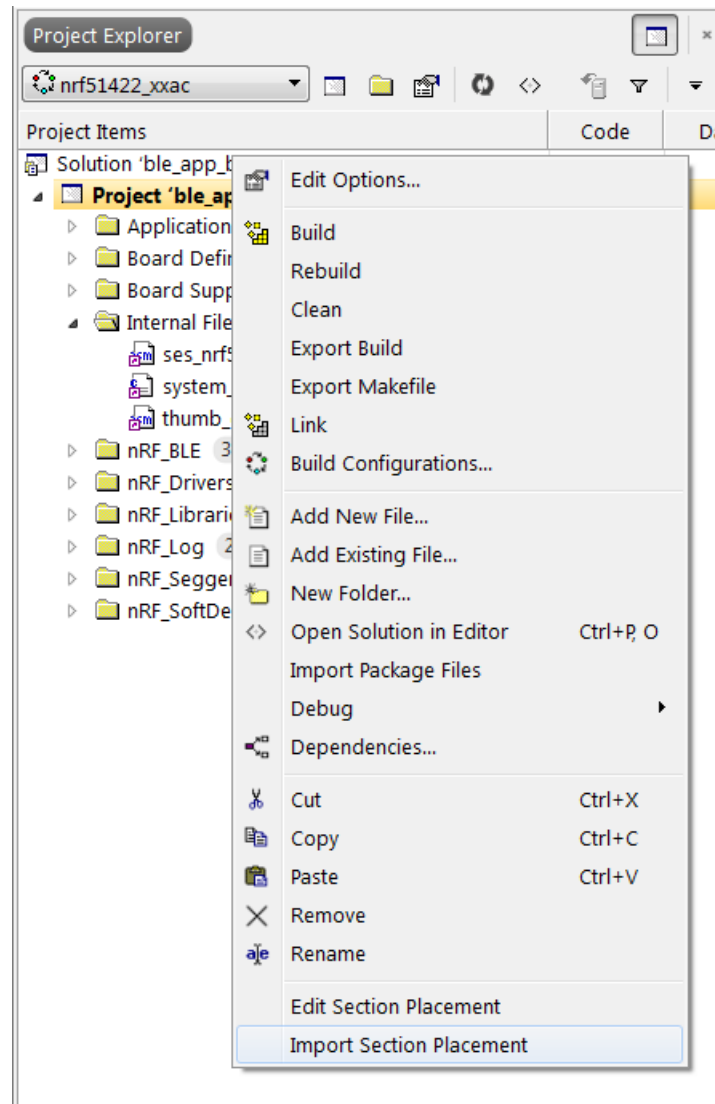
The user include directories should now look similar to this:



7. If your project uses modules that require section variables (for example, the Peer Manager, Flash Data Storage, or Flash Storage), define where in the flash information from these modules should be stored.
 - a) Download the `flash_placement.xml` and place it in your project directory.

Files are provided for the following versions of the nRF5 SDK:

 - For nRF5 SDK v12.x.x: [flash_placement.xml](#)
 - For nRF5 SDK v13.x.x and nRF5 SDK v14.0.0: [flash_placement.xml](#)
 - b) Right-click on your project in the Project Explorer and select **Import Section Placement**.



c) Confirm that you want to use the section placement for the current build configuration.

8.3 Compiling the application

You can compile the application from a [SES](#) project provided by the nRF5 SDK v14.2.0 or later, or from the project that you created based on an older SDK example.

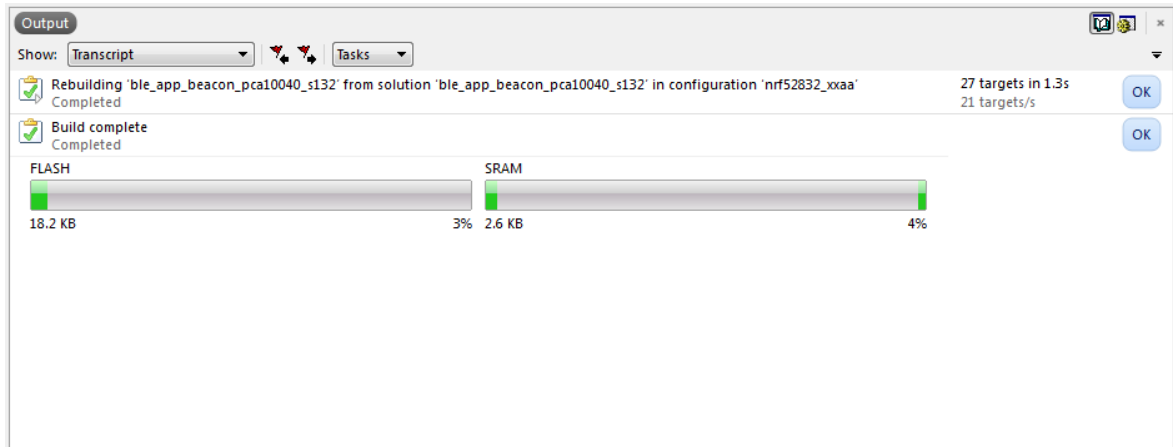
1. Open your project in [SES](#).

In nRF5 SDK v14.2.0 or later, [SES](#) projects are located in the `ses` subfolder of the example folder, for example, `SDK_dir/examples/ble_peripheral/ble_app_uart/pca10040/s132/ses`.

2. Select **Build > Build project_target**.

Alternatively, press `F7`. Make sure that there are no build errors.

The output should look similar to this:



8.4 Configuring placement of the SoftDevice

If your application uses Bluetooth or ANT, you must program a *SoftDevice* in addition to the application.

Note:

If your application does not use a *SoftDevice*, you can skip this step.

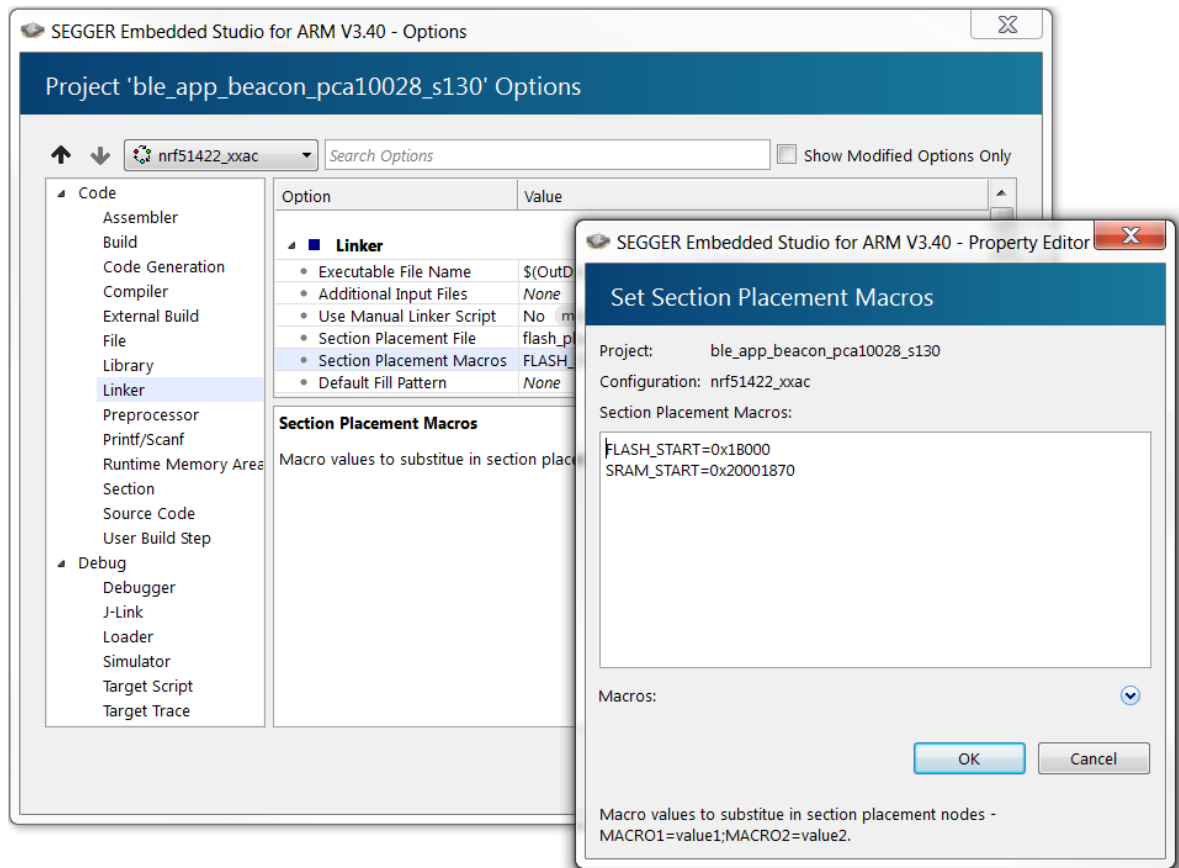
If you are using a *SES* project from nRF5 SDK v14.2.0 or later, the placement of the SoftDevice is already configured correctly and you can skip this step.

If your application requires a *SoftDevice*, the flash and SRAM position where the compiled binary will be placed must be configured as follows:

1. In *SES*, right-click your project and select **Edit Options**.
2. Select **Linker**.
3. In the Section Placement Macros field, add values for FLASH_START and SRAM_START.

To find the correct values, check the Keil project that you imported (in Keil μ Vision, select **Options for Target > Target**), or program the firmware with approximate values, run it, and check the log output in the debug terminal for the recommended values.

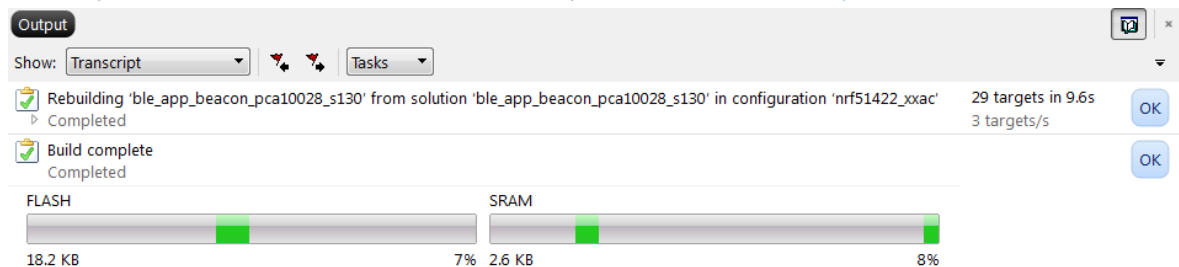
For example, when running the ble_app_uart example application from nRF5 SDK v12.3.0 on PCA10028 with SoftDevice S130 v2.0.1, specify the section placement macros as shown:



4. Select **Build > Rebuild project_target** to rebuild the project.

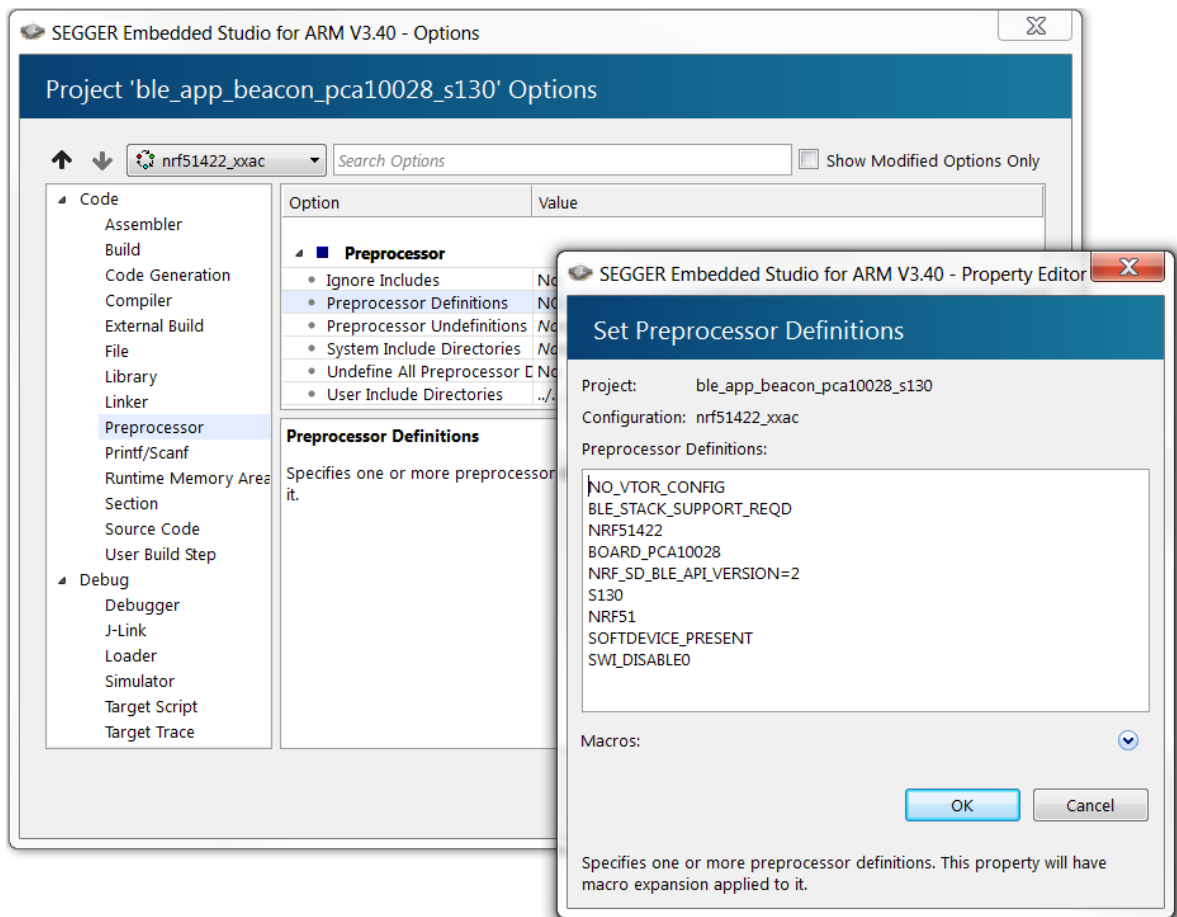
Alternatively, press **ALT + F7**.

The output should now look similar to this, with space reserved for the *SoftDevice*:

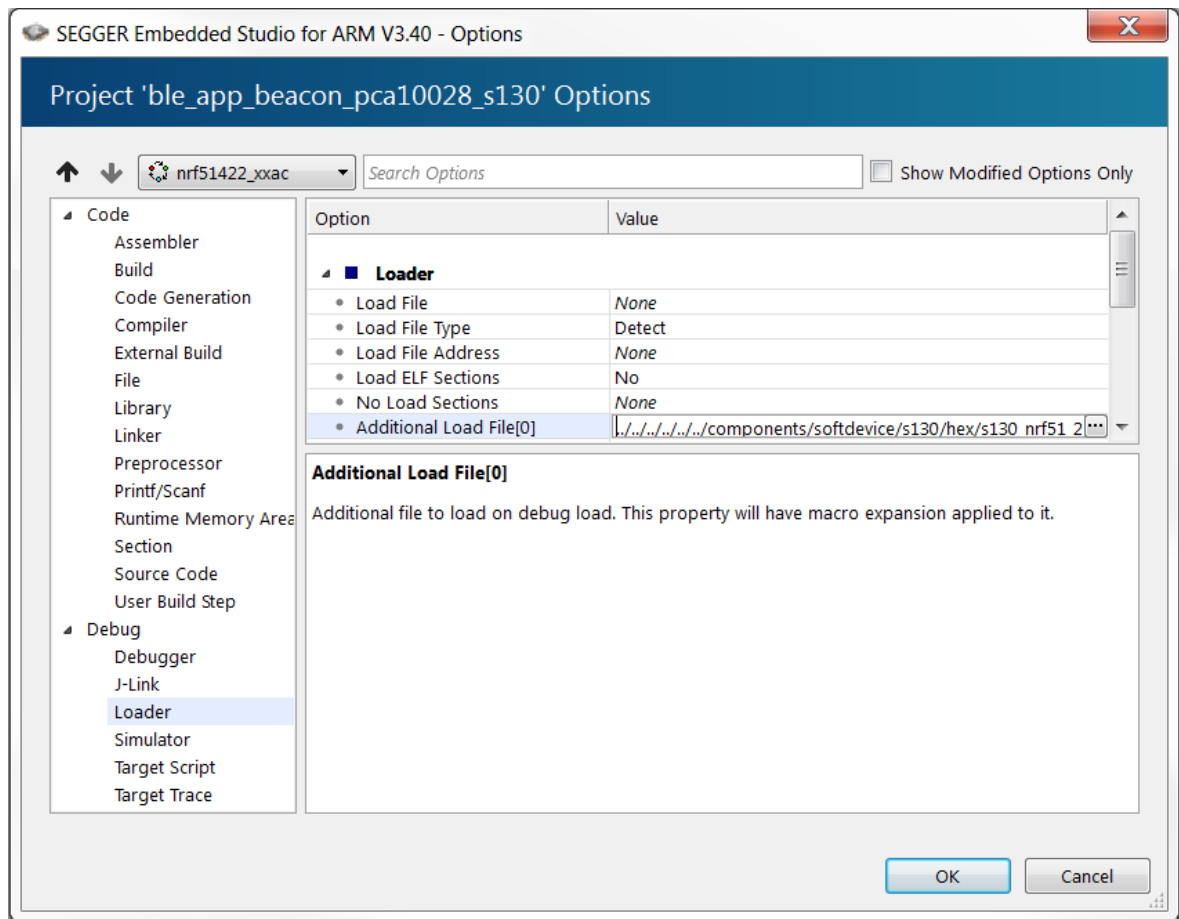


5. Right-click your project and select **Edit Options**.
6. Select **Preprocessor**.
7. Add the definition `NO_VTOR_CONFIG` to the Preprocessor Definitions.

This definition tells *SES* to expect a *SoftDevice* to be present that will forward exceptions and interrupts to the application.



8. In the Debug section of the project options, select **Loader**.
9. Add the absolute path to the *SoftDevice* to the Additional Load File[0] field, for example, `../../../../../../../../components/softdevice/s130/hex/s130_nrf51_2.0.1_softdevice.hex`.



8.5 Programming the firmware

After compiling the application, you can program it to the development board. If you configured a [SoftDevice](#) to be used, it is programmed together with the application.

1. Connect the development board to your computer.
2. Select **Debug > Build and Run**.

Alternatively, press `Ctrl + F5`.

8.6 Adding files

After compiling and programming an unmodified example to ensure that your toolchain is set up correctly, modify your project by adding files and libraries.

8.6.1 Adding source files

All source files that are part of the application you are developing must be added to the project.

You can add existing files or create files in the project directory.

- To add an existing file, right-click your project or any subfolder in the **Project Explorer** and select **Add Existing File**. Browse to the file that you want to import and open it.

The original file is not copied into the project, but it is included from its original location. That means that any modifications that you do will apply to all projects that use this file.

- To create a file, right-click your project or any subfolder in the **Project Explorer** and select **Add New File**.

By default, the file is created in the project directory.

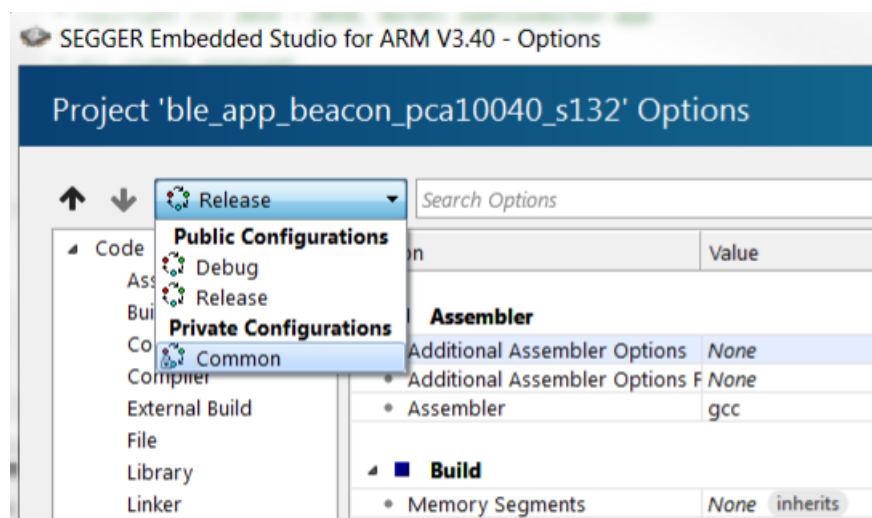
8.6.2 Including header files

Required header files must be linked to your project by adding their path to the user include directories.

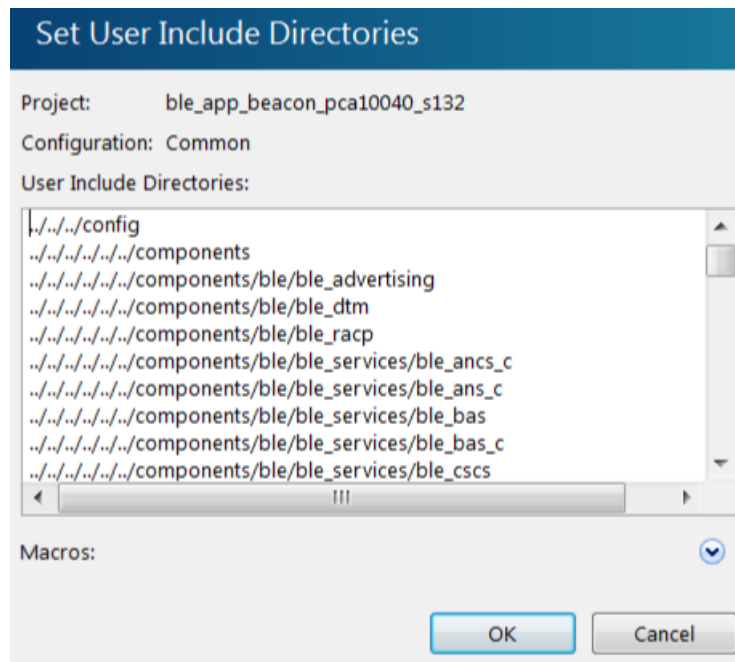
Header files contain function declarations and macro definitions. You can request the use of header files by adding a `#include` preprocessing directive in your source files.

Header files are not linked to the project through the Project Explorer. To include a header file so that [SES](#) can find it, you must add its path to the list of directories in which [SES](#) looks for header files:

- In the **Project Explorer**, right-click your project and select **Edit Options**.
- In the **Project Options** window, select the **Common** configuration (sorted under **Private Configurations**).



- Select **Preprocessor**.
- Double-click **User Include Directories** and add the path to the folder that contains the header file.
You can specify an absolute path or a path that is relative to the project directory. Using a relative path is preferable if you might want to move or copy your project to, for example, a new SDK version in the future.



5. Click **OK**.

9 Communicating with the board

Unless you programmed a very simple application, you probably want to connect to the board from your computer to display logging information or send input. You can use *Real Time Transfer (RTT)* or *Universal Asynchronous Receiver/Transmitter (UART)* for communicating with the board.

SEGGER Real Time Transfer (RTT) is a proprietary technology for bidirectional communication that supports J-Link devices and ARM-based microcontrollers. The advantage of using *RTT* is that it is very efficient and does not require any other peripheral than the J-Link debugging interface. *RTT* is directly supported in *SES*.

Connecting via *UART* is quick and power-efficient, but it requires dedicated use of the *UART* peripheral for logging. The nRF5 *DKs* and the nRF51 Dongle include a *UART* to USB CDC ACM bridge, which is needed to connect to the *UART*. Alternatively, you can use an external *UART* to USB bridge. We use the term CDC-UART to refer to *UART* communication through the *UART* to USB CDC ACM bridge, to distinguish it from communication through the Nordic UART Service (NUS) over Bluetooth Low Energy.

9.1 Connecting via RTT

SES directly supports *RTT*. You can see *RTT* output in the Debug Terminal during debugging sessions.

Alternatively, you can use J-Link RTT to view *RTT* output. See the following sections for instructions for Windows and Linux.

9.1.1 Connecting via RTT on Windows

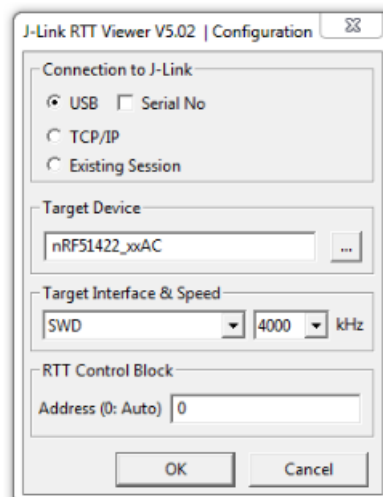
To communicate via *RTT*, connect your development board via USB and run the J-Link RTT Viewer.

Note: *SES* natively supports *RTT*. If enabled, the monitor shows up when you start debugging. Alternatively, you can use SEGGER's J-Link RTT Viewer as described below.

The J-Link RTT Viewer is installed as part of the nRF5x Command Line Tools.

To run the J-Link RTT Viewer on Windows, complete the following steps:

1. Select the correct target device.
The target device is represented by the ID of your development board.
2. Select **SWD** as the target interface.



9.1.2 Connecting via RTT on Linux

To communicate via [RTT](#), connect your development board via USB and use SEGGER's J-Link RTT to establish a connection.

Note: [SES](#) natively supports [RTT](#). If enabled, the monitor shows up when you start debugging. Alternatively, you can use SEGGER's J-Link RTT Viewer as described below.

SEGGER's J-Link RTT is part of the J-Link Software and Documentation Pack, which is available from [SEGGER downloads](#).

To use J-Link RTT on Linux, complete the following steps:

1. Enter `JLinkExe -if SWD` to set up the connection:

```
you@yourcomputer:~$ JLinkExe -if SWD
SEGGER J-Link Commander V5.10u (Compiled Mar 17 2016 19:06:22)
DLL version V5.10u, compiled Mar 17 2016 19:06:19

Connecting to J-Link via USB...O.K.
Firmware: J-Link OB-SAM3U128-V2-NordicSemi compiled Mar 15 2016 18:03:17
Hardware version: V1.00
VTref = 3.300V

Type "connect" to establish a target connection, '?' for help
J-Link>
```

2. Enter `connect` at the prompt to establish the connection.
JLinkExe will ask for additional information. You can accept the default values.
3. From another terminal, start **JLinkRTTClient**.

RTT output is visible in the terminal that runs **JLinkRTTClient**.

9.2 Connecting via CDC-UART

To connect via CDC-UART, start a terminal emulator and connect to the used COM port.

There is a wide variety of terminal emulators that you can use, for example, minicom or screen (both terminal-based, available for Linux), Termit (GUI-based, Windows only), or PuTTY (GUI-based, available for multiple operating systems).

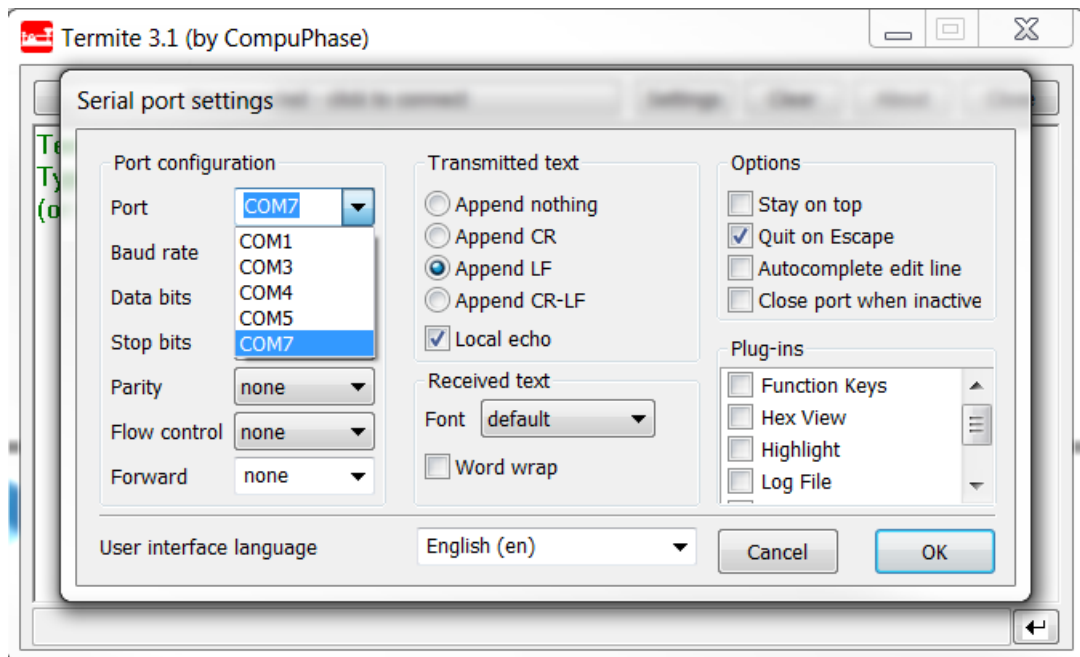
When configuring the connection, use the following [UART](#) settings:

- Baud rate: 115.200
- 8 data bits
- 1 stop bit
- No parity
- HW flow control: RTS/CTS

The following instructions show how to configure Termit correctly. Other terminal emulators can be set up in a similar way.

1. Download and install the latest version of [Termit](#).
2. Connect the development board to your computer.
3. Open Termit and click **Settings**.

Depending on what devices you have connected to your computer, you might have several choices, as shown in the following figure:



4. Select the correct COM port to connect to the board.

To find the correct port, follow these steps:

- a) Go to the start menu in Windows and type `devmgmt.msc` to open the Device Manager.
- b) Scroll down and expand **Ports (COM & LPT)**.
- c) Find the port named **JLink CDC UART Port** and note down the number in parentheses.
- d) If you have more than one J-Link UART port, unplug the one that you want to use, plug it back in, and observe which one appeared last.

5. Configure the baud rate and the flow control. Use the default values for the rest of the settings (8 data bits, 1 stop bit, no parity).

By default, the SDK uses a baud rate of 115200 and RTS/CTS flow control.

6. Make sure that **Append LF** is selected.

This option appends a newline character to any text that is sent.

7. Configure the terminal to send an RTS (Ready To Send) signal to the development board:

- a) Go to **Settings > Plug Ins**.
- b) Enable **Status LEDs** and click **OK**.
- c) Click on the dark green rectangle above RTS to set this signal high.

The text `Start...` is displayed in Terminate.

Alternatively, you can turn off hardware flow control in your application.

10 Testing the application

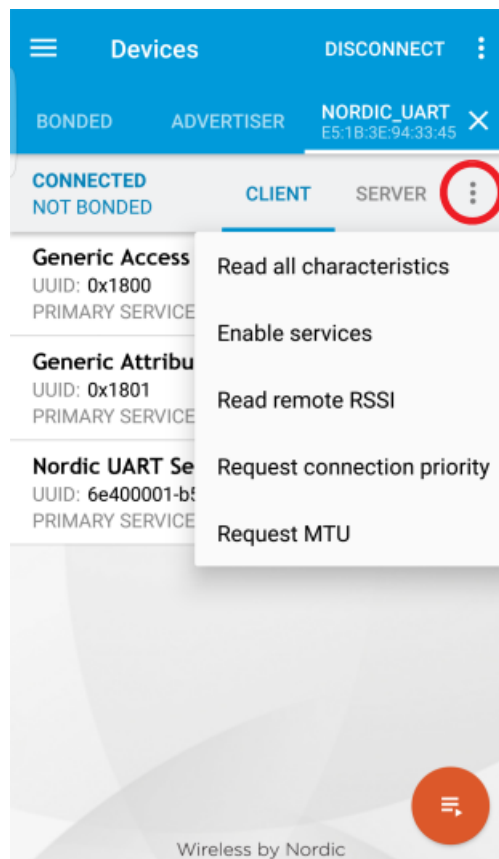
The next step after compiling and programming your application is to test it. Nordic Semiconductor provides its own testing tool, nRF Connect, which is available both for mobile and for desktop.

10.1 Testing with a mobile device

If you have a mobile device that supports Bluetooth Low Energy, download the nRF Connect app from Google Play or App Store to test your application.

The following procedure assumes that you have programmed the ble_app_uart example from the nRF5 SDK. Steps for testing other examples are similar. See the testing instructions for each example in the [nRF5 SDK documentation](#) for more information.

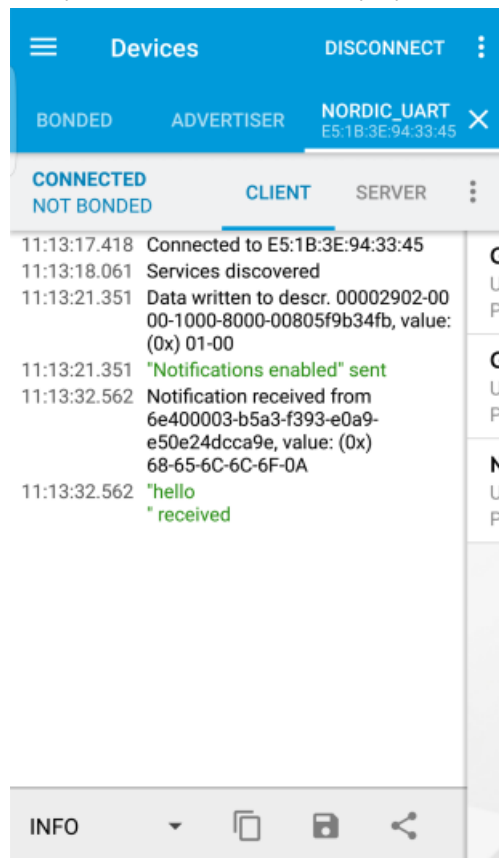
1. Download and install nRF Connect from Google Play or App Store.
2. Open nRF Connect.
3. Make sure that the [DK](#) is running the ble_app_uart example.
LED1 should be blinking every 2 seconds, indicating that it is advertising.
4. Tap **Scan**.
5. Find the device and tap **Connect**.
The default device name for the ble_app_uart example is "Nordic_UART".
6. When connected, tap the options button below the device name and select **Enable services**.



This example communicates over Bluetooth Low Energy using the Nordic UART Service (NUS).

7. Tap the options button and select **Show log**.

8. In a [terminal connected via CDC-UART](#), enter `hello` and send it to the [DK](#).
The text is sent through the [DK](#) to your device, which will display it in the nRF Connect log:



10.2 Testing with a computer

If you have an nRF51 Dongle or a second [DK](#), you can test your application with nRF Connect for Desktop. nRF Connect for Desktop is available for Windows, Linux, and macOS.

Note: This method requires an nRF5 [DK](#) or dongle to be connected to your computer.

The following procedure assumes that you have programmed the `ble_app_uart` example from the nRF5 SDK. Steps for testing other examples are similar. See the testing instructions for each example in the [nRF5 SDK documentation](#) for more information.

1. Download and install [nRF Connect for Desktop](#).
2. Connect the dongle or the second [DK](#) to a USB port of your computer.
3. [Connect to the board that runs the ble_app_uart example via CDC-UART](#).
4. Open nRF Connect for Desktop and add the Bluetooth Low Energy app.
5. Launch the Bluetooth Low Energy app.
6. Select the serial port for the dongle or the [DK](#) that is connected to your computer (not the board that runs the `ble_app_uart` example).

If the device has not been used with the nRF Connect Bluetooth Low Energy app before, you may be asked to update the J-Link firmware and connectivity firmware for the device. You need to have the correct connectivity firmware on the nRF SoC to continue. When the nRF SoC has been programmed with the correct firmware, the nRF Connect Bluetooth Low Energy app proceeds to connect to it over USB. When the connection is established, the device appears in the main view.

7. Click **Start scan**.

8. Find the device and click **Connect**.

The default device name for the ble_app_uart example is "Nordic_UART".

9. Select the **UART** RX characteristic value.

10. Write 30 31 32 33 34 35 36 37 38 39 (the hexadecimal value for the string "0123456789") and click **write**.

The text "0123456789" is displayed in the terminal that is connected to the board via **UART**.

11. Enter any text, for example, Hello, in the terminal.

In nRF Connect, the **UART** TX characteristic value changes to the corresponding ASCII value. For example, for Hello, the value is 48 65 6C 6C 6F.

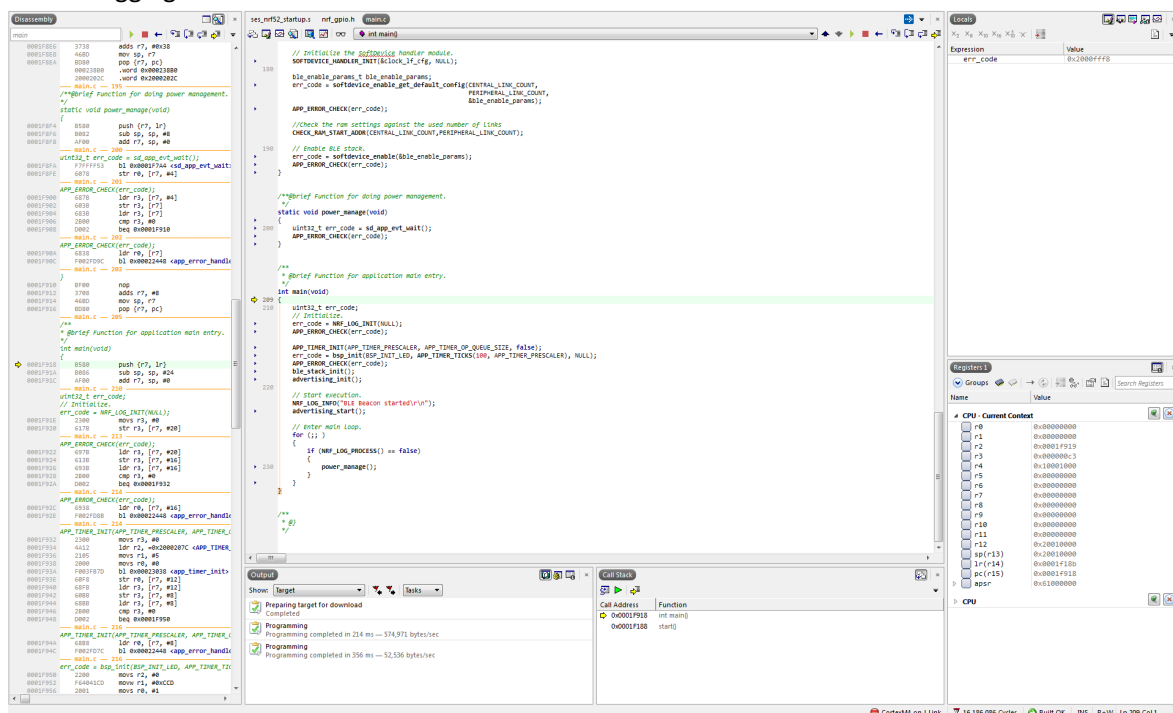
11 Debugging

To actually see what is happening on the development board while the application is running, you must set up a J-Link debugging session. [SES](#) has an integrated debugger that you can use to step through your application.

1. Open your project in [SES](#).
2. Select **Debug > Go**.

Alternatively, press **F5**.

The debugging interface looks like this:



By default, the application will break in main. You can set additional break points, single-step through the application, read registers, and so on.

This video tutorial shows you how to use [SES](#) for debugging:

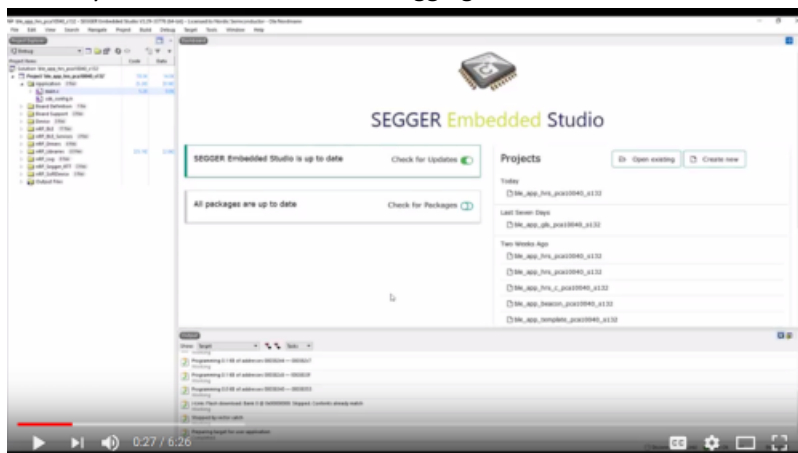


Figure 2: YouTube Tutorial

Glossary

Development Kit (DK)

A development platform used for application development.

GNU Compiler Collection (GCC)

A compiler system that supports various programming languages, maintained by the GNU Project.

Integrated Development Environment (IDE)

A software application that provides facilities for software development.

Real Time Transfer (RTT)

A proprietary technology for bidirectional communication that supports J-Link devices and ARM-based microcontrollers, developed by SEGGER Microcontroller.

SEGGER Embedded Studio (SES)

A cross-platform [IDE](#) for embedded C/C++ programming with support for Nordic Semiconductor devices, produced by SEGGER Microcontroller.

SoftDevice

A wireless protocol stack that complements the nRF5 Series SoCs. Nordic Semiconductor provides these stacks as qualified, precompiled binary files.

System on Chip (SoC)

A microchip that integrates all the necessary electronic circuits and components of a computer or other electronic systems on a single integrated circuit.

Target

The goal of an operation, for example, programming a specific image on a device, compiling a specific set of files, or removing previously generated files.

Universal Asynchronous Receiver/Transmitter (UART)

A hardware device for asynchronous serial communication between devices.

Acronyms and abbreviations

These acronyms and abbreviations are used in this document.

DK

Development Kit

GCC

GNU Compiler Collection

IDE

Integrated Development Environment

RTT

SEGGER Real Time Transfer

SES

SEGGER Embedded Studio

SoC

System on Chip

UART

Universal Asynchronous Receiver/Transmitter

Legal notices

By using this documentation you agree to our terms and conditions of use. Nordic Semiconductor may change these terms and conditions at any time without notice.

Liability disclaimer

Nordic Semiconductor ASA reserves the right to make changes without further notice to the product to improve reliability, function or design. Nordic Semiconductor ASA does not assume any liability arising out of the application or use of any product or circuits described herein.

All information contained in this document represents information on the product at the time of publication. Nordic Semiconductor ASA reserves the right to make corrections, enhancements, and other changes to this document without notice. While Nordic Semiconductor ASA has used reasonable care in preparing the information included in this document, it may contain technical or other inaccuracies, omissions and typographical errors. Nordic Semiconductor ASA assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.

Life support applications

Nordic Semiconductor products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury.

Nordic Semiconductor ASA customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nordic Semiconductor ASA for any damages resulting from such improper use or sale.

RoHS and REACH statement

Nordic Semiconductor products meet the requirements of *Directive 2011/65/EU of the European Parliament and of the Council* on the Restriction of Hazardous Substances (RoHS 2) and the requirements of the *REACH* regulation (EC 1907/2006) on Registration, Evaluation, Authorization and Restriction of Chemicals.

The SVHC (Substances of Very High Concern) candidate list is continually being updated. Complete hazardous substance reports, material composition reports and latest version of Nordic's REACH statement can be found on our website www.nordicsemi.com.

Trademarks

All trademarks, service marks, trade names, product names and logos appearing in this documentation are the property of their respective owners.

Copyright notice

© 2018 Nordic Semiconductor ASA. All rights are reserved. Reproduction in whole or in part is prohibited without the prior written permission of the copyright holder.

