nRF51822 Development Kit

nRF51822

User Guide v1.0

Copyright © 2012 Nordic Semiconductor ASA. All rights reserved.
Reproduction in whole or in part is prohibited without the prior written permission of the copyright holder.

Contents

1 Introduction

The nRF51822 *Bluetooth*® low energy/2.4 GHz Proprietary Development Kit (DK) provides a complete solution for development, testing, and evaluating the nRF51822 device. The nRF51822 is part of the nRF51 series which offers a range of ultra-low power, System on Chip (SoC) solutions for your 2.4 GHz wireless products.

1.1 Minimum requirements

- nRFgo Starter Kit
- nRFgo Studio v1.13 or later
- Computer with a minimum of 2 USB ports; 4 are preferable
- Windows XP or Windows 7

1.2 External resources

- Keil MDK-ARM Lite v4.54 or later https://www.keil.com/demo/eval/arm.htm
- J-Link Software v4.52b or later

1.3 Writing conventions

This User Guide follows a set of typographic rules that makes the document consistent and easy to read. The following writing conventions are used:

- Commands are written in `Lucida Console`.
- Pin names are written in **`Consolas`.**
- File names and User Interface components are written in **bold**.
- Internal cross references are italicized and written in ***semi-bold.***

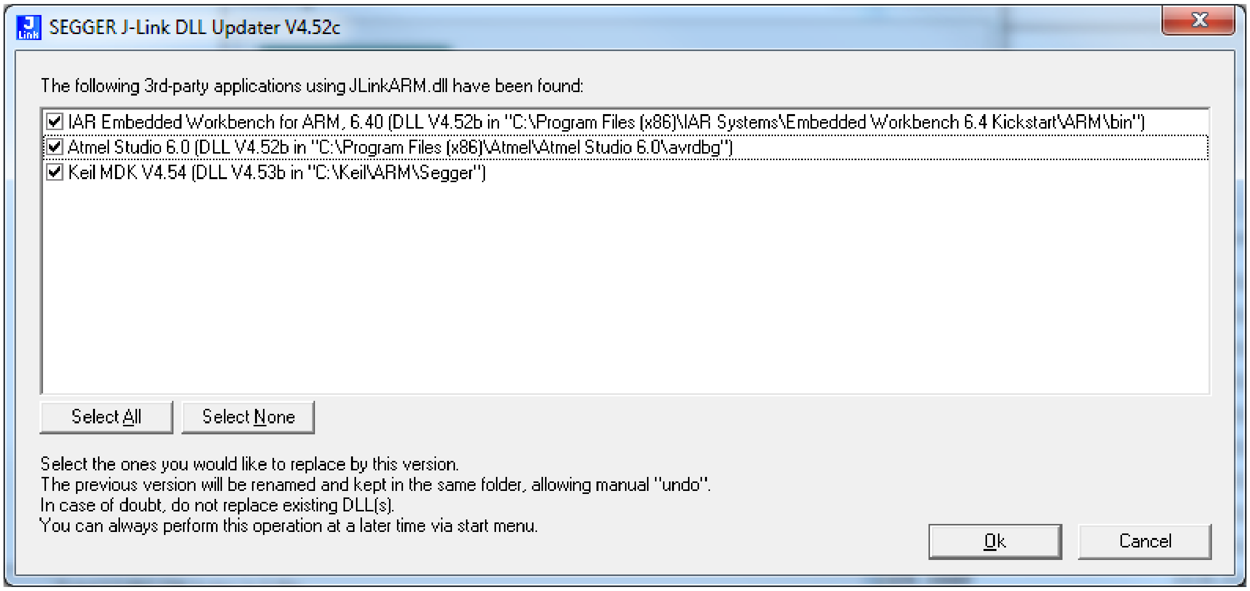
1.4 Development kit release notes

Date	Version	Description
September 2012	1.0	**Known issues** • PCA10000, PCA10004 and, PCA10005 v1.0: • The antenna matching network and layout on these boards is suitable for applications using TX output power 0 dBm or less. These boards are not suitable for applications using +4 dBm TX output power.

2 Quick start

Register, download, and install

1. Download and install Keil MDK-ARM Lite from https://www.keil.com/demo/eval/arm.htm to your hard drive. If you have Keil MDK-ARM Lite version 4.54 or later already installed, go to step 2.
2. Download and run the J-Link Software (version 4.52b or later) and documentation pack for Windows from http://www.segger.com/jlink-software.html. The serial number from your SEGGER J-Link hardware is needed.
3. During installation you will be prompted to select the IDE that should be updated with the latest SEGGER DLLs. Check the box for **Keil MDK** and any other IDEs you want to use with SEGGER.



SEGGER J-Link DLL Updater V4.52c

The following 3rd-party applications using JLinkARM.dll have been found:

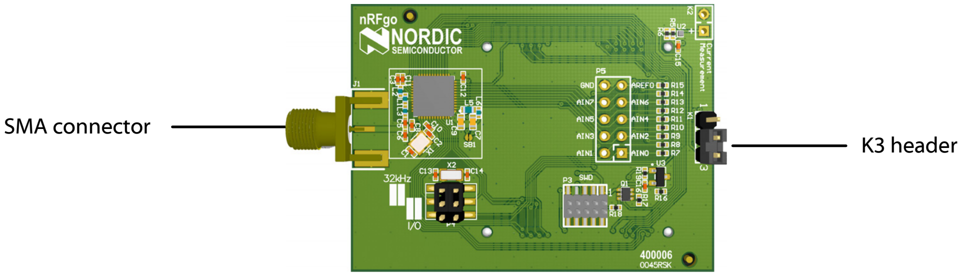
☑ IAR Embedded Workbench for ARM, 6.40 (DLL V4.52b in "C:\Program Files (x86)\IAR Systems\Embedded Workbench 6.4 Kickstart\ARM\bin")
☑ Atmel Studio 6.0 (DLL V4.52b in "C:\Program Files (x86)\Atmel\Atmel Studio 6.0\avrdbg")
☑ Keil MDK V4.54 (DLL V4.53b in "C:\Keil\ARM\Segger")

Select the ones you would like to replace by this version.
The previous version will be renamed and kept in the same folder, allowing manual "undo".
In case of doubt, do not replace existing DLL(s).
You can always perform this operation at a later time via start menu.

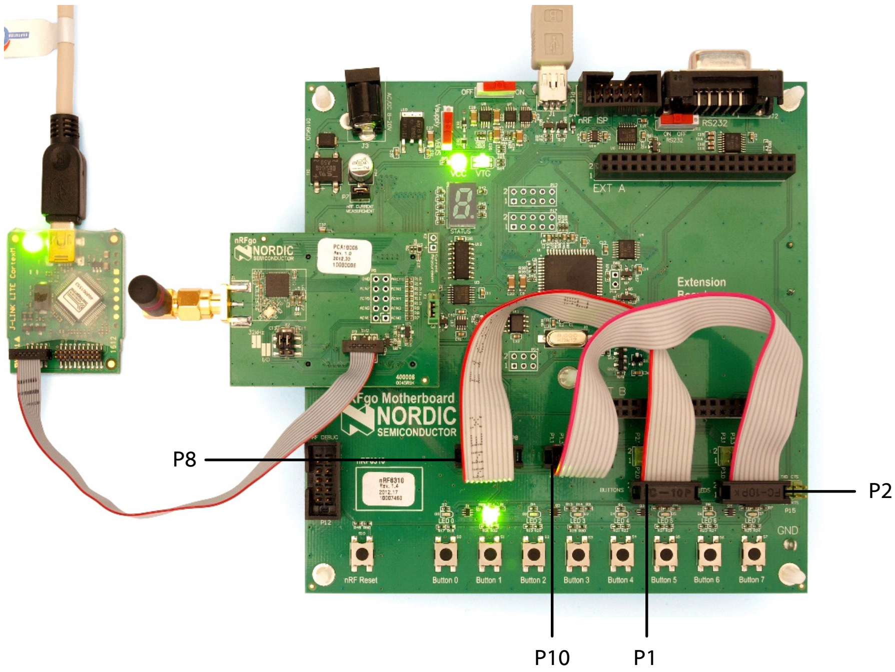
4. Go to http://www.segger.com/IDE_Integration_Keil.html#knownproblems for MDK v4.54. Download JL2CM3 and copy it to **<keil>/ARM/Segger**. This patch is necessary for the SEGGER debugger to work.
5. Go to www.nordicsemi.com and log in to your Nordic My Page account.
6. Select **My Products** from the left menu.
7. Enter the product key (included with this kit) into the Product Key field and click **Add**.
8. Click the **Downloads** link in the Overview, My Products table.
9. Download and run the nRF518 SDK installer. Make sure to choose the **Keil MDK-ARM** installer option.

Connect the hardware

1. Screw the antenna into the SMA connector on the nRF51822 DK SMA connector module.
2. Ensure that header **K3** has a jumper connecting pin 2 and 3. The two upper rows of **P4** should have two jumpers placed vertical matching the 32 kHz marking.



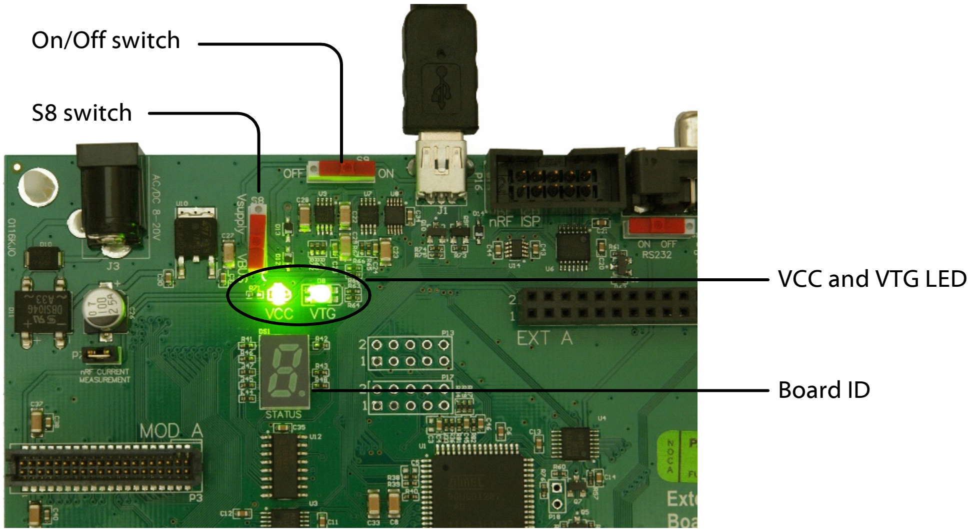
3. Plug the nRF51822 DK SMA connector module into the nRFgo Starter Kit Motherboard (nRF6310).
4. Connect a USB cable from the Motherboard to your computer.
5. Using two 10 pin flat cables (2.54 mm, included in the nRFgo Starter Kit), connect one cable between **P8** (PORT0) and **P1** (BUTTONS) and one between **P10** (PORT1) and **P2** (LEDS) on the nRFGo Motherboard. Make sure the red marking on the cable is always connected to pin 1, as shown below.
6. Connect the SEGGER J-Link board to the nRF51822 module with the 10 pin flat cable (1.27 mm, provided in the Development Kit).
7. Connect a USB cable from the J-Link board to your computer.



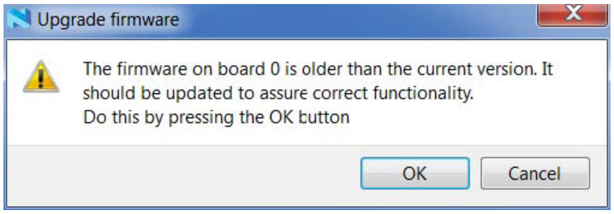
Turn on and set the supply voltage

1. Turn the **S8** switch on the Motherboard to **VBUS**.
2. Turn the **On/Off** switch on the Motherboard to **ON**.

3. The **VCC** and **VTG** LEDs will light up..

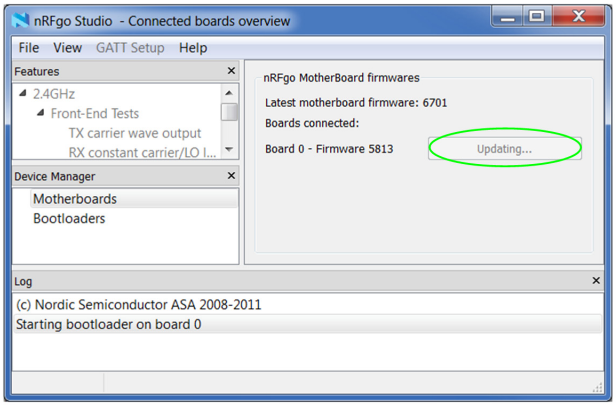


4. Start nRFgo Studio. The Motherboard firmware may require an update, and if needed, you will be prompted to begin the update. If this occurs, click **OK** and wait for the update to complete.



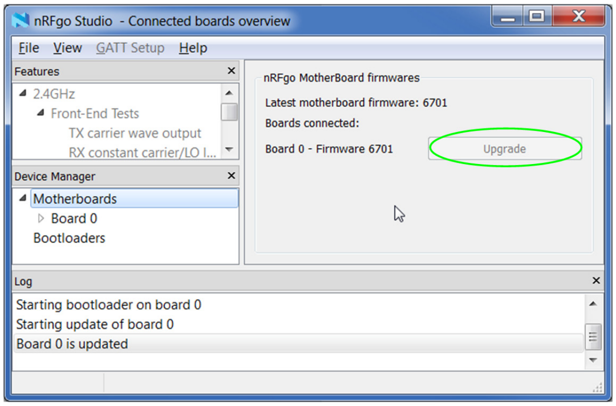
The firmware on board 0 is older than the current version. It should be updated to assure correct functionality.
Do this by pressing the OK button

Note: While updating the firmware, the screen will look similar to the image on the left below. On completion of the update, the "Updating…" message will change to "Upgrade" as seen in the figure on the right; however, this option is not immediately available after the latest update and will be greyed out.



nRFgo MotherBoard firmwares
Latest motherboard firmware: 6701
Boards connected:
Board 0 - Firmware 5813

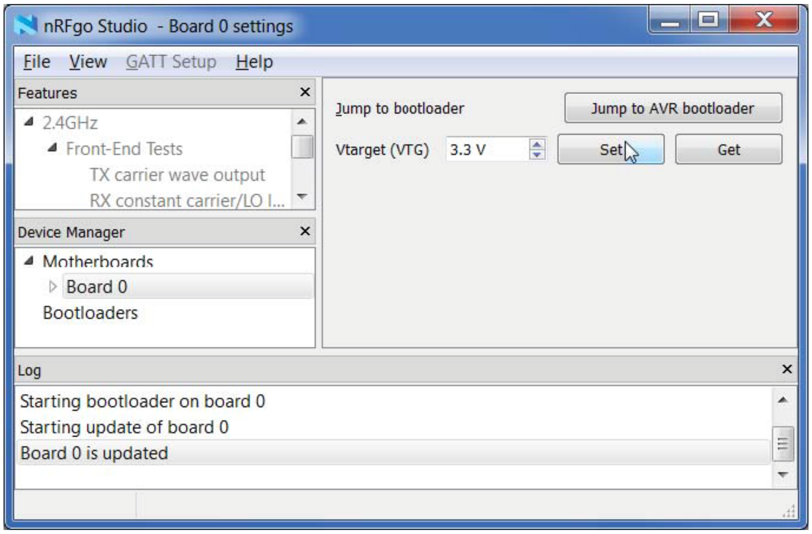
(c) Nordic Semiconductor ASA 2008-2011
Starting bootloader on board 0



nRFgo MotherBoard firmwares
Latest motherboard firmware: 6701
Boards connected:
Board 0 - Firmware 6701

Starting bootloader on board 0
Starting update of board 0
Board 0 is updated

5. Select the desired Motherboard from the Device Manager pane in nRFgo Studio. To find the board ID, look on the 7-segment display on the Motherboard.
6. In nRFgo Studio, you can control the supply voltage (VTG) to the connected nRF51822 module. Make sure the voltage is set to 3.3 V.



Vtarget (VTG) 3.3 V

```
Starting bootloader on board 0
Starting update of board 0
Board 0 is updated
```

Note: Please refer to nRFgo Studio's help file for further information. To access, while in nRFgo Studio, press **F1** and the help file opens.

Note: The nRF51822 device is capable of operating at higher and lower voltages. This can be tested when the debugger is not required for programming or debugging the application. However, for the J-Link Lite Cortex-M (that is included in the kit) to function correctly, the supply voltage must be set to 3.3 V on the Motherboard. At all voltages other than 3.3 V, the J-Link Lite CortexM-9 debugger should be disconnected from the nRF51822 module (SEGGER has debuggers that are capable of debugging in the entire voltage range). The supply voltage operating range of the nRF51822 device is stated in the Product Specification.

Start the Blinky project

1. Locate the Blinky project found under <keil path> **\ARM\Device\Nordic\nRF51822\Board\nrf6310\blinky_example\arm**.
2. Open the Blinky project in Keil µVision by double clicking the **blinky.uvproj** file.
3. From the **Select Target** list, select “nRF51822” and then click **Build** or press **F7** to compile the Blinky project.
4. Click **Load** to download and run the Blinky example firmware. **LED0** to **LED7** on the nRFgo Starter Kit Motherboard should now blink in a sequential order.

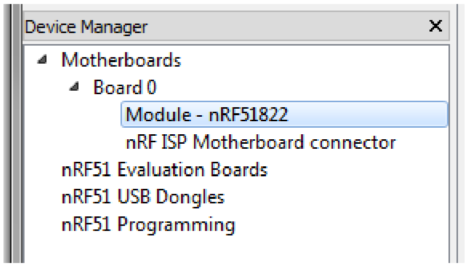
2.1 *Bluetooth* low energy heart rate monitor demo

Download and program the SoftDevice

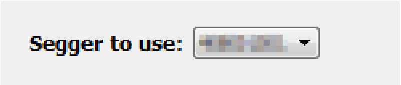
Access the S110 nRF51822 SoftDevice by entering the product key (included with the Development Kit) into the Product Key field in My Page.

To program your device:

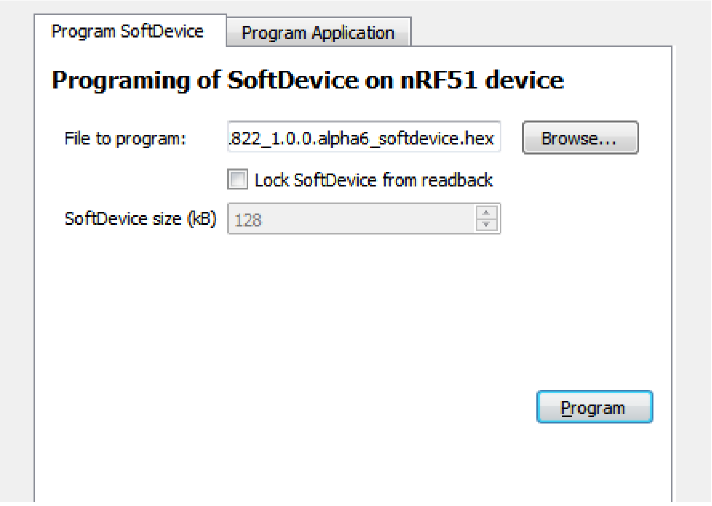
1. Open nRFgo Studio.
2. Open the Device Manager. Select the motherboard your module is connected to.
3. Select the module.



4. If you have more than one SEGGER debugger connected, you need to select which one you want to use.



5. Select the **Program SoftDevice** tab.



Programing of SoftDevice on nRF51 device

File to program: .822_1.0.0.alpha6_softdevice.hex

☐ Lock SoftDevice from readback

SoftDevice size (kB) 128

6. Click **Browse** and navigate to the file you downloaded.
7. Click **Program**.

Compile, program, and run the heart rate monitor demo

1. Locate the Heart Rate demo project found in the folder <keil path> **\ARM\Device\Nordic\nrf51822\board\nrf6310\ble\ble_app_hrs\arm**.
2. Open the Heart Rate demo project in Keil µVision by double clicking the **ble_app_hrs.uvproj** file.
3. Click the **Build** icon or press **F7** to build the project.
4. Only one SEGGER device should be connected to your computer. Make sure it is also connected to the nRF51822 development module to ensure the program is downloaded to the correct target.
5. Go to the **Flash** menu and click **Download** to load the program (or click the **Load** icon).
6. The Heart Rate demo example will start executing. **LED0** should be lit indicating it is advertising.
7. The application advertises for 3 minutes. If a connection isn't made within this period, the application sets nRF51822 in System Off.
8. To start advertising again press **Button 0** (or the **nRF Reset** button)

Install the nRF51822 development dongle (PCA10000)

1. Plug the nRF51822 development dongle into a USB port on your computer.
2. An icon will appear in the lower right corner of your monitor showing that the drivers are being installed. Wait until it is ready.

Note: The development dongle must be unplugged when loading example programs into the flash.

Scan for available *Bluetooth* low energy devices

1. Start the Master Control Panel from the Windows Start menu (**Start > All Programs > Nordic Semiconductor > Master Control Panel**).
2. Make sure the Master Emulator is detected. The Master Emulator item list should show COMnn-xxxxxxxxx (*nn* gives the COM port number; *xxxxxxxxx* is the SEGGER serial number printed on the dongle). Restart the application if it doesn't appear in the item list. Before continuing, make sure you have selected the correct device by verifying the serial number in the item list with the serial number printed on the device.
3. When you use the nRF51822 development dongle for the first time, you must first program it with the Master Emulator Firmware.
    a. In the Master Control Panel menu click **File** and select **Flash Programming**.
    b. Click **Browse**. This opens a browser that automatically points to the location of the **mefw_nrf51822_<version>_firmware.hex** (<version> will be replaced by a number giving the version of the actual firmware).
    c. Select the **Master Emulator Firmware** file and click **Open**.
    d. Click **Program** to start programming the selected device.
    e. When the programming is finished click **Exit** to go back to the main window.
4. Click **Start discovery**. The Master Emulator will scan for available *Bluetooth* low energy devices within range and list them.
5. Select the device that appears in the Discovered Devices list. (The device will identify itself with CompleteLocalName equal to HRS_APPVx.x)
6. Select the device and click **Select device**.
7. Click **Service Discovery**. In the **Service discovery** pane you will see the services and characteristics of the device. On the Motherboard **LED0** will go off and **LED1** will light up indicating that is has gone from advertising to connected.
8. Click **Enable** services. You should see the Heart Rate Measurement characteristic and Battery Level being notified every few seconds with a different value (the Heart Rate Measurement/Battery Level value line will blink green for each notification).

3 Kit content

The nRF51822 Development Kit (DK) consists of hardware and access to software components, documentation, and design files from www.nordicsemi.com.

3.1 nRF51822 Development kit hardware content

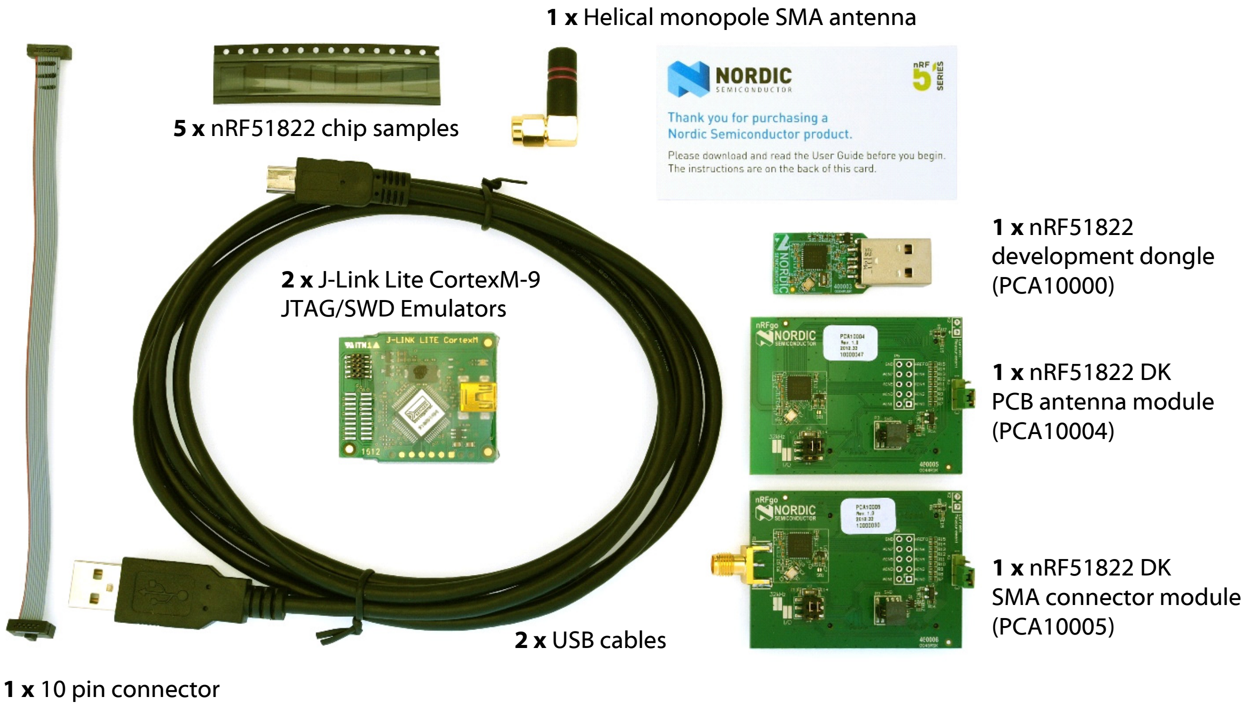

1 x Helical monopole SMA antenna
5 x nRF51822 chip samples
Thank you for purchasing a Nordic Semiconductor product.
Please download and read the User Guide before you begin. The instructions are on the back of this card.
1 x nRF51822 development dongle (PCA10000)
2 x J-Link Lite CortexM-9 JTAG/SWD Emulators
1 x nRF51822 DK PCB antenna module (PCA10004)
1 x nRF51822 DK SMA connector module (PCA10005)
2 x USB cables
1 x 10 pin connector

Figure 1 nRF51822 DK hardware content

3.2 Downloadable content

The nRF51822 Development Kit includes firmware source code, documentation, hardware schematics, and layout files. To access this information, log in to your My Page account, enter your product key, and download the files.

3.2.1 nRF51822 DK software content

- nRFgo Studio
- nRF518 Software Development Kit (SDK)
    - Precompiled HEX files
    - Source code
    - Keil ARM project files
- S110 nRF51822 SoftDevice

3.2.2 nRF51822 DK documentation

- This User Guide
- nRF51 Series Reference Manual
- nRF51822 PPS
- S110 SoftDevice Specification
- nRF518 SDK
- nRF51822 PAN

3.2.3 Schematics, Bill of Materials, PCB layout files, and production files

The ZIP file and its subdirectories contain the hardware design files for the nRF51822 DK.

- Altium Designer files
- PCB layout files
- Production files
    - Assembly drawings
    - Drill files
    - Gerber files
    - Pick and Place files
    - Bill of Materials
- Schematics

4 Development kit configuration

This chapter explains how to download third party content, the development environment setup, and how to program the nRF51822.

4.1 Development environment

ARM compiler/IDE (not included in this kit)

All the source code projects and examples can be compiled and used with the Keil Microcontroller Development Kit (MDK). For full use of the Development Kit source code projects, and to upgrade firmware, download and install the free KEIL MDK-ARM Lite from https://www.keil.com/demo/eval/arm.htm.

J-Link Lite CortexM-9 driver (not included in this kit)

For installing drivers for the SEGGER J-Link Lite CortexM-9, visit www.segger.com and go to the **Downloads** section. Select the J-Link Lite and download the software and documentation. You must correctly install the drivers for the device to use the J-Link debugger with Keil MDK. See ***Appendix A: "Installing drivers and configuring KEIL projects for the SEGGER debugger"*** on page 53.

4.1.1 Programming the nRF51822 device

The nRF51822 device can be programmed from several environments. In this section we will show how to program using Keil MDK-ARM. The nRF51822 DK can be configured to develop proprietary 2.4 GHz protocol-based applications and *Bluetooth* 4.0 single-mode applications.

For development of proprietary 2.4 GHz protocol-based applications, you need:

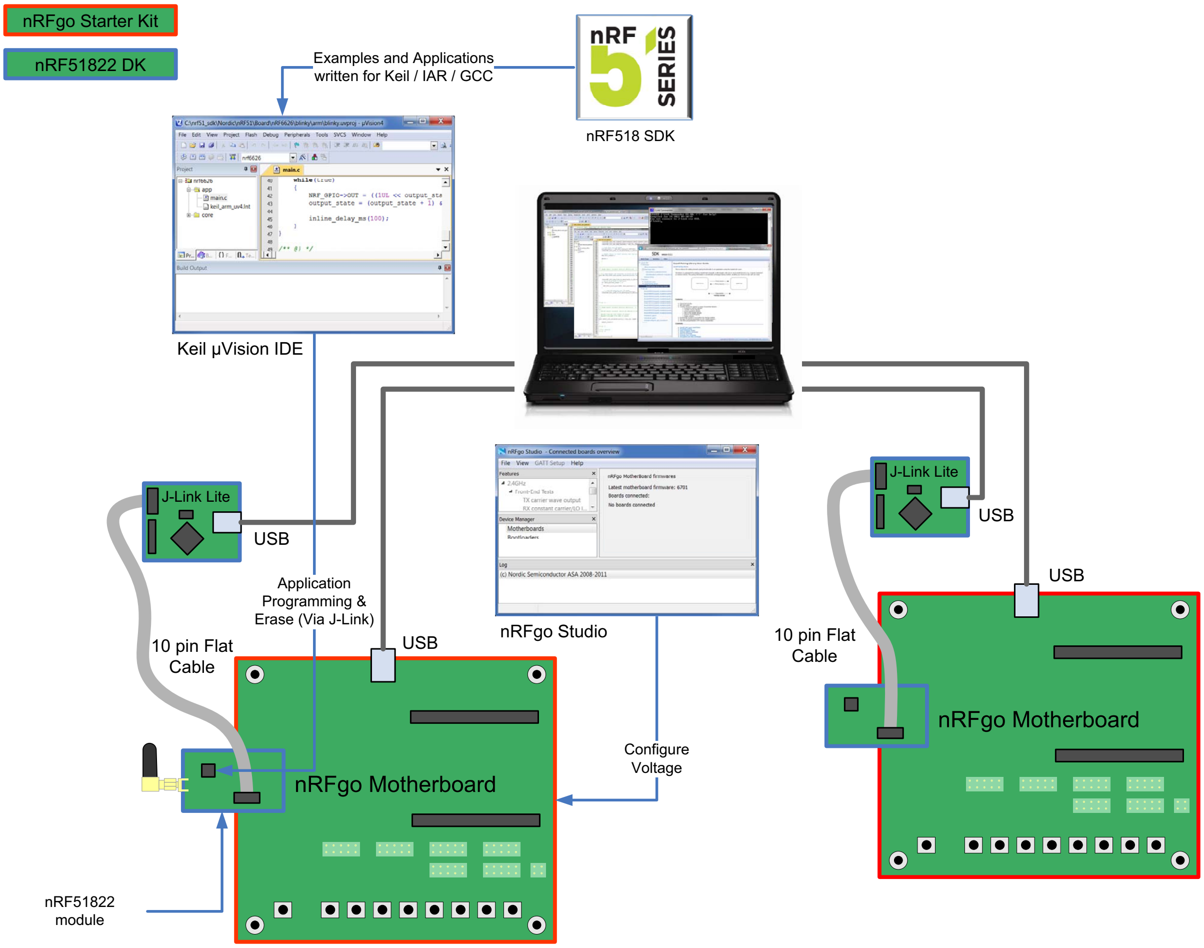
- 2 x Motherboards (from the nRFgo Starter Kit - not included)
- 2 x nRF51822 modules

For development or demonstration of *Bluetooth* 4.0 single-mode applications, you need:

- 1 x Motherboard with attached nRF51822 module
- 1 x nRF51822 development dongle (PCA10000)
- Master Control Panel PC software

Figure 2 and ***Figure 3*** show the relationship between the hardware and software components, and the Motherboard(s).

Note: The Keil µVision IDE is not included in the kit content.



```
    while(true)
    {
        NRF_GPIO->OUT = ((1UL << output_sta
        output_state = (output_state + 1) &

        inline_delay_ms(100);
    }
}

/** @} */
```

nRFgo MotherBoard firmwares

Latest motherboard firmware: 6701

Boards connected:

No boards connected

(c) Nordic Semiconductor ASA 2008-2011

Figure 2 *nRF51822 DK configuration for 2.4 GHz based development*

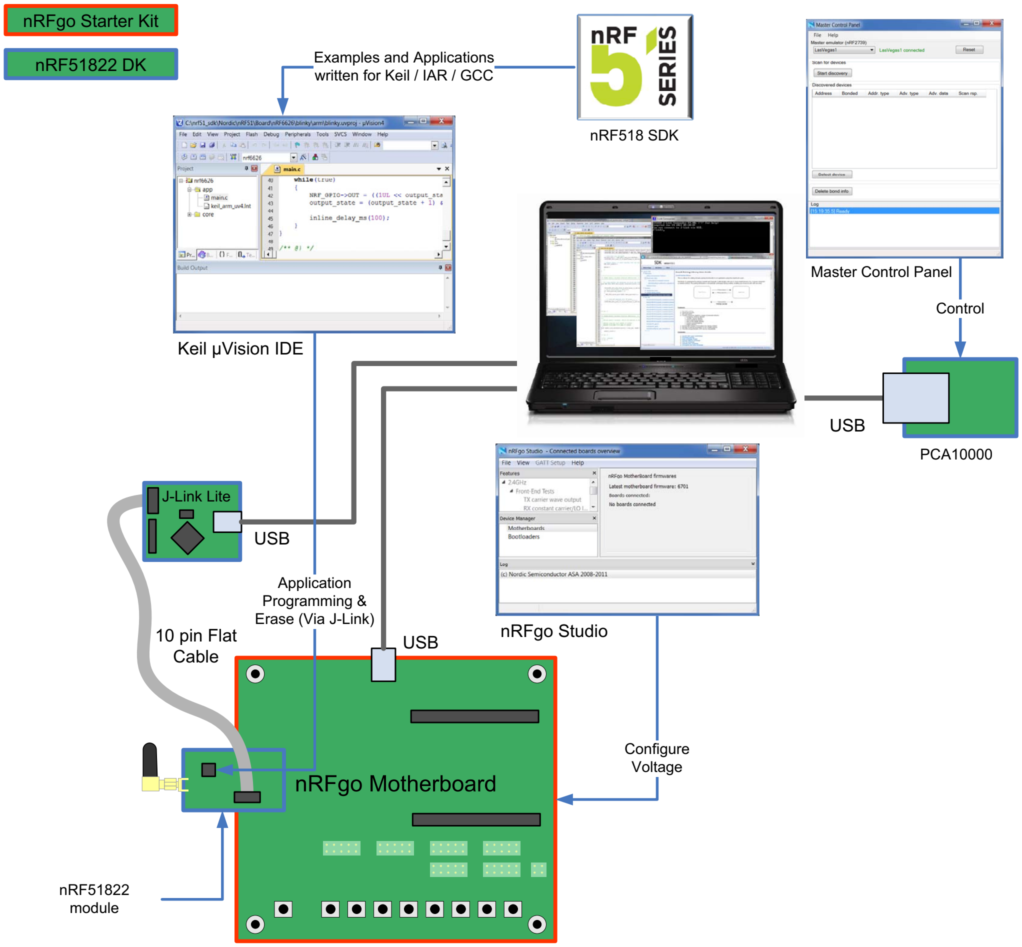


*Figure 3 nRF51822 DK configuration for **Bluetooth** 4.0 single-mode*

4.2 Kit set up

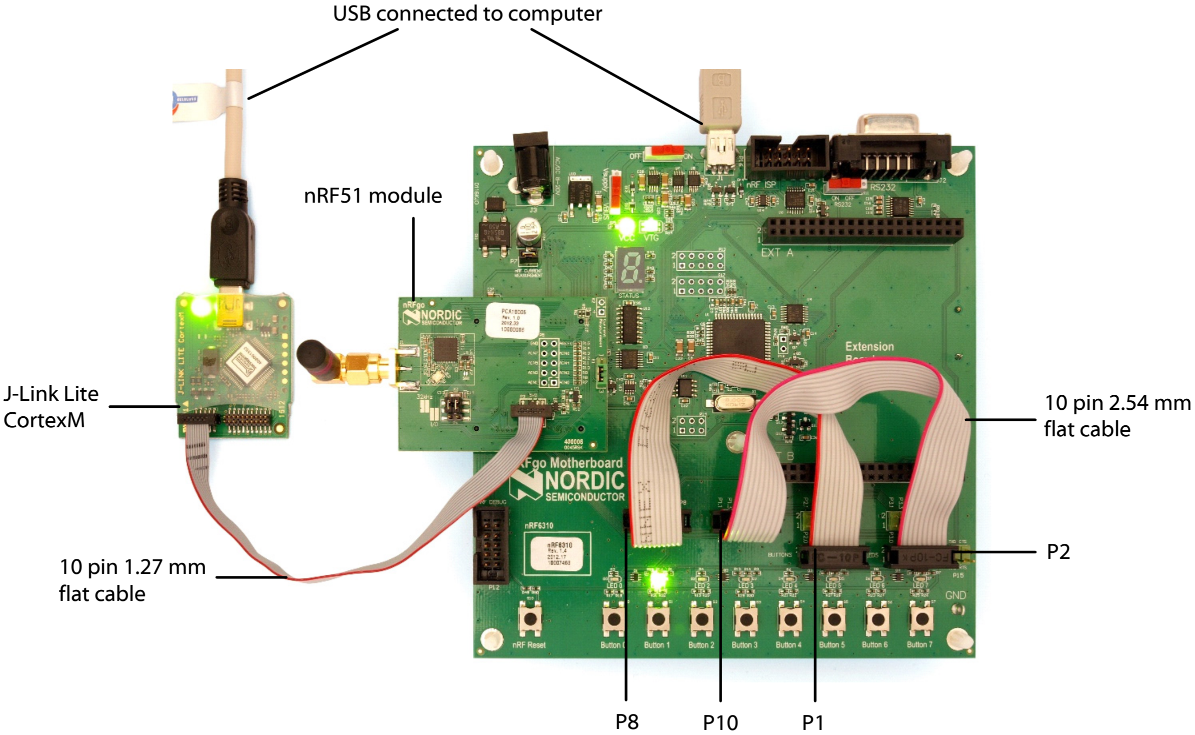


Figure 4 *Kit set up*

4.2.1 nRFgo nRF51822 DK modules

The nRF51822 modules contain the nRF51822 device and its interfaces (see ***section 5.1*** on page 18 for a complete description). The modules are plugged into, and powered from, the Motherboard. Make sure the jumper on header **K1** is placed in the default position as described in ***section 5.1.8*** on page 24. The Motherboard is connected to your computer with a USB cable and turned on as described***"Turn on and set the supply voltage"*** on page 6.

Note: On the Motherboard, the two 10 pin flat cables connecting P8 (PORT0) and P1 (BUTTONS), and P10 (PORT1) and P2 (LEDS) are required to connect the correct I/Os to the LEDs for the example projects in ***"Quick start"*** on page 5and for projects in the SDK.

Note: With the nRFgo Studio application you can control the supply voltage (VTG) to the connected nRF51822 development dongle as described in ***"Turn on and set the supply voltage"*** on page 6**.**

4.2.2 nRF51822 development dongle (PCA10000)

The nRF51822 development dongle (PCA10000) enables you to see the data sent between the dongle and a single nRF51822 device. The nRF51822 development dongle is plugged into a USB port on your computer as described in ***"Compile, program, and run the heart rate monitor demo"*** on page 10.

4.2.3 J-Link Lite CortexM-9 JTAG/SWD Emulator

The programming and debugging (SWD) interface of the nRF51822 device is accessed through a 10 pin connector (**P3**) on the nRF51822 DK module.

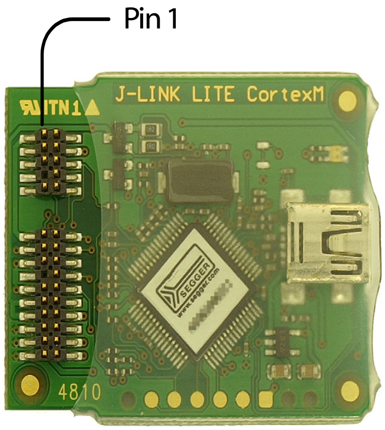


Figure 5 *Pin 1 position on the SEGGER J-Link Lite CortexM-9*

Connect the JTAG/SWD emulator using the 10 pin 1.27 mm flat cable supplied with the kit to the nRF51822 DK module at **P3** and to your computer with a USB cable, as shown in ***Figure 6***.

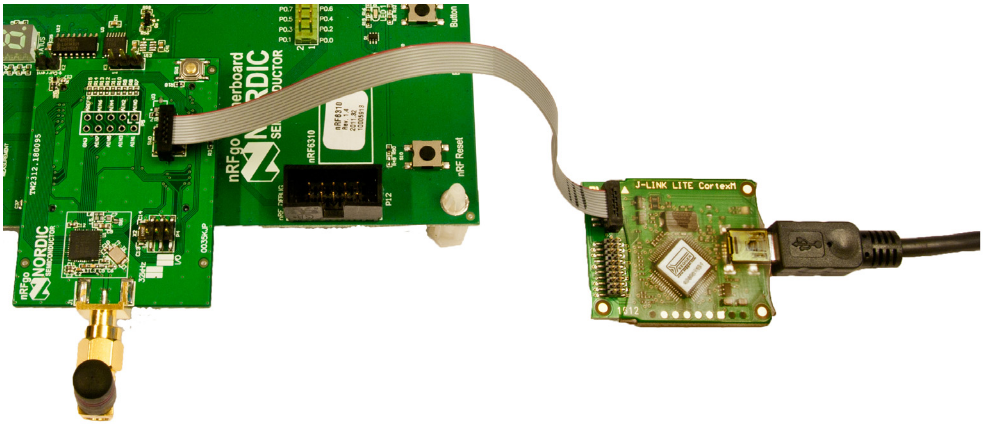


Figure 6 *nRF51822 DK module connected to the SEGGER J-Link Lite CortexM-9*

5 Hardware description

This chapter describes the nRF51822 DK modules and the nRF51822 development dongle (PCA10000).

5.1 nRFgo nRF51822 DK modules

The nRF51822 modules (PCA10004, PCA10005) are delivered with an unprogrammed nRF51822 chip.

5.1.1 Key features

The nRF51822 DK modules have the following key features:

- 2.4 GHz compatible with nRF24L devices
- *Bluetooth* low energy compatible
- nRF51822 IC
- Current Shunt Monitor (CSM) for current measurements
- nRFgo Motherboard integration
- PCB antenna (PCA10004 only)
- SMA connector compatibility (PCA10005 only)
- SWD interface connector for programming and debugging

5.1.2 Hardware pictures

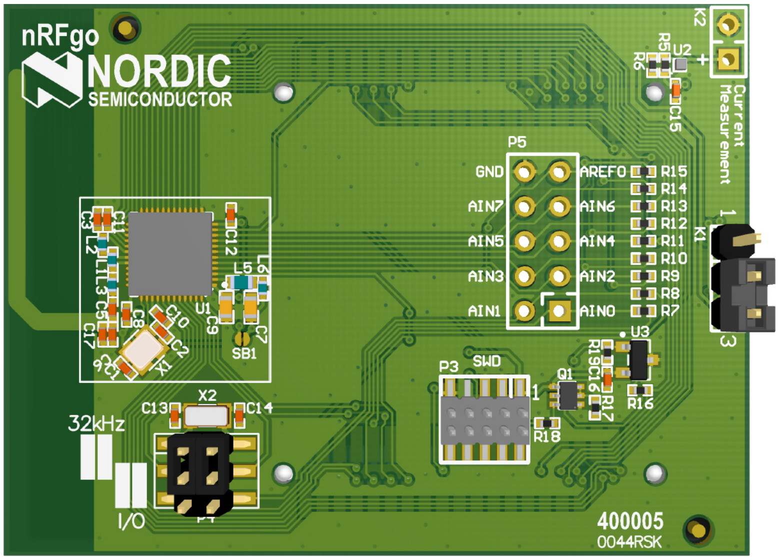


Figure 7 nRF51822 DK module (PCA10004) top

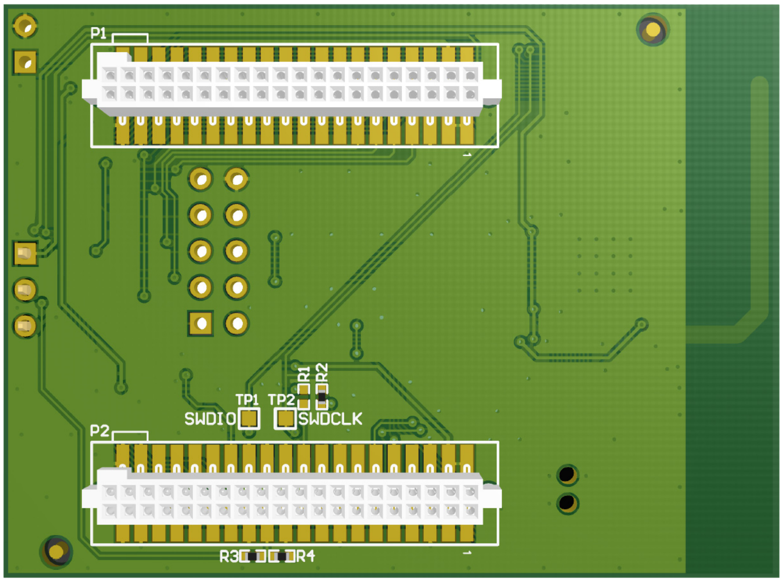


Figure 8 *nRF51822 DK module (PCA10004) bottom*

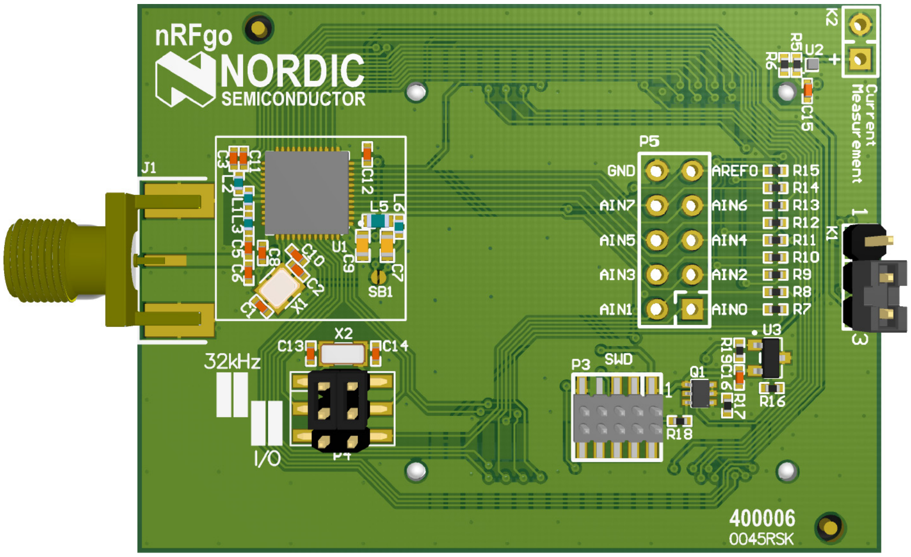


Figure 9 *nRF51822 DK module (PCA10005) top*

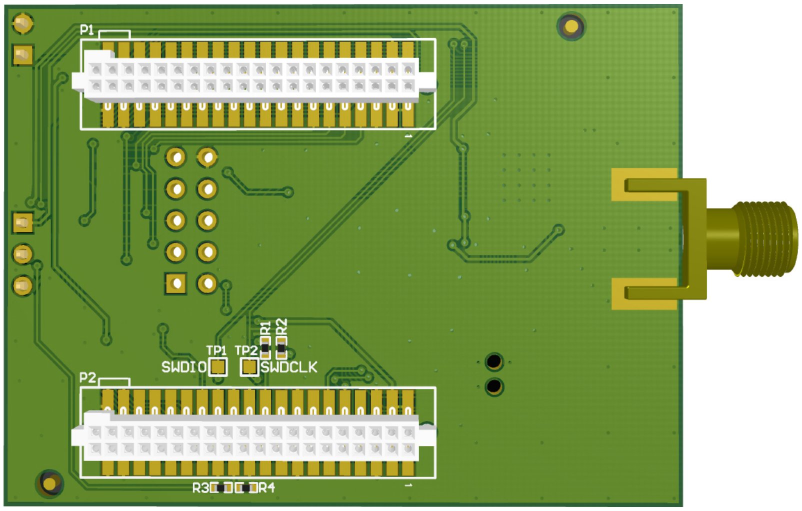


Figure 10 nRF51822 DK module (PCA10005) bottom

5.1.3 Block diagram

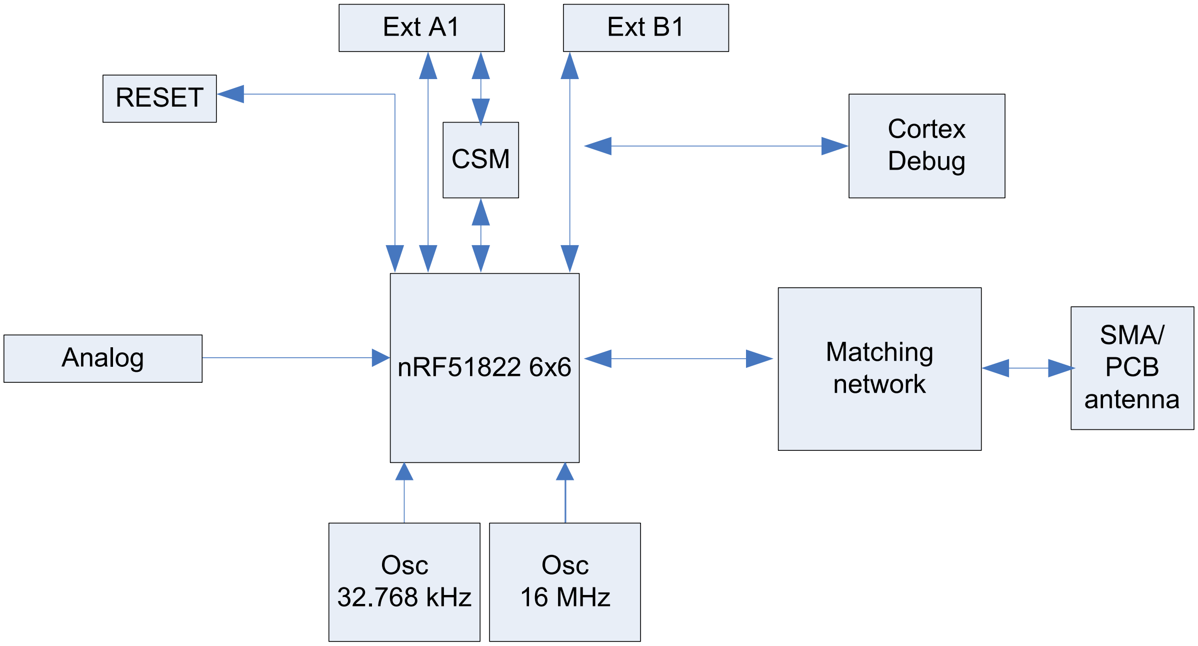
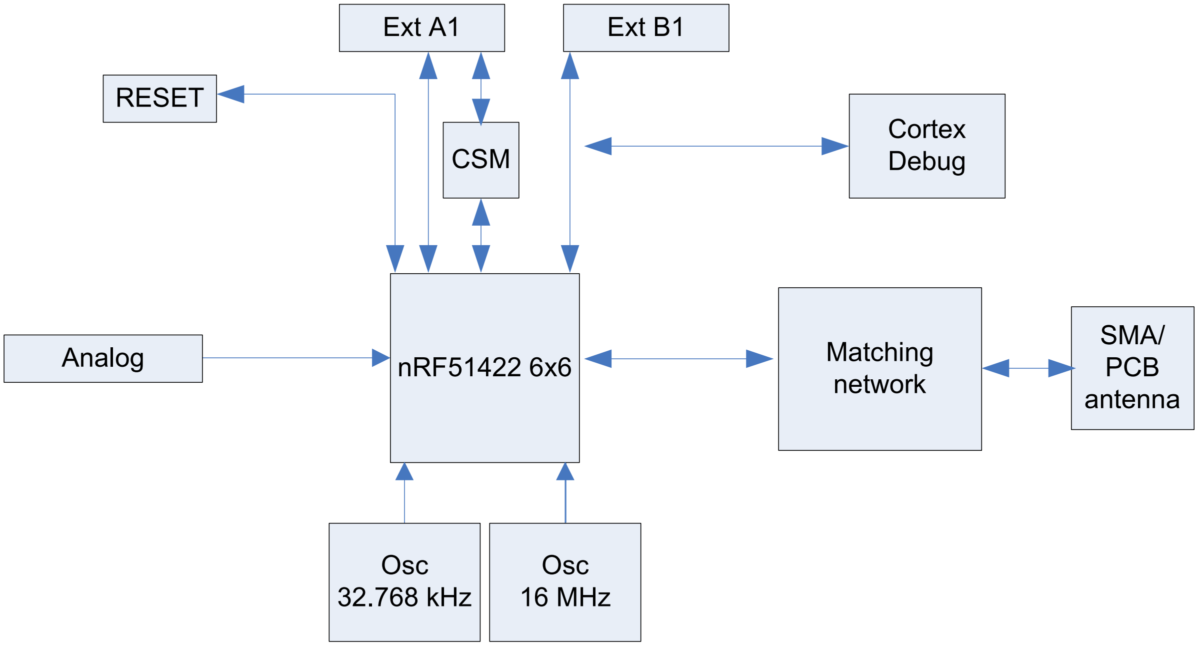


Figure 11 nRF51822 DK module block diagram

5.1.4 nRFgo nRF51822 DK module extension



Connectors **P1** and **P2** on the nRF51822 DK module connects to the nRFgo Motherboard.

VCC	1	2	VCC_nRF'
VCC	3	4	VCC_nRF'
GND	5	6	GND
P0.00	7	8	P0.01
P0.02	9	10	P0.03
P0.04	11	12	P0.05
P0.06	13	14	P0.07
GND	15	16	GND
Not in use	17	18	Not in use
Not in use	19	20	Not in use
GND	21	22	GND
Not in use	23	24	Not in use
Not in use	25	26	RESET
Not in use	27	28	Not in use
P0.08	29	30	P0.09
P0.10	31	32	P0.11
P0.12	33	34	P0.13
P0.14	35	36	P0.15
GND	37	38	GND
GND	39	40	GND

P1

VTG	1	2	VTG
VTG	3	4	VTG
GND	5	6	GND
P0.24	7	8	P0.25
P0.26	9	10	P0.27
P0.28	11	12	P0.29
P0.30	13	14	Not in use
GND	15	16	GND
Not in use	17	18	Not in use
Not in use	19	20	Not in use
GND	21	22	GND
BoardID	23	24	GND
GND	25	26	GND
Not in use	27	28	BoardID_EE
P0.16	29	30	P0.17
P0.18	31	32	P0.19
P0.20	33	34	P0.21
P0.22	35	36	P0.23
GND	37	38	GND
GND	39	40	GND

P2

Figure 12 *nRF51822 DK module connectors - **P1** and **P2***

5.1.5 Analog Inputs

Direct access to the nRF51822 analog input is available on connector **P5** on the nRF51822DKmodule.

Note: **P5** is not mounted on the board.

The analog inputs are routed through the extension connectors to the Motherboard. To avoid noise from the Motherboard, the 0 Ω resistors must be removed.

AIN0	1	2	AIN1
AIN2	3	4	AIN3
AIN4	5	6	AIN5
AIN6	7	8	AIN7
AREF0	9	10	GND

Figure 13 *Analog inputs connector **P5** on the nRF51822 DK board*

5.1.6 SWD interface

The programming and debugging interface is accessed through the 10 pin connector **P3** on the nRF51822 DK module.

VTG	1	2	SWDIO
GND	3	4	SWDCLK
GND	5	6	NC
NC	7	8	NC
GND	9	10	NC

Figure 14 *SWD interface connector P3*

Pin	Label	Description
1	VTG	Reference voltage for programmer
2	SWDIO	Serial Wire Debug Data
3	GND	Ground
4	SWDCLK	Serial Wire Debug Clock
5	GND	Ground
6	NC	No Connection
7	NC	No Connection
8	NC	No Connection
9	GND	Ground
10	NC	No Connection

Table 1 *SWD interface connector pin description*

5.1.7 32.768 kHz crystal

The nRF51822 can use an optional 32.768 kHz crystal (X2) for higher accuracy and lower average power consumption. On the nRF51822 DK module, P0.26 and P0.27 are disconnected from the Motherboard and connected to the two jumpers on the nRF51822 DK module at the position 32 kHz on connector **P4** (Pin 1 and Pin 3 for XL1 and Pin 2 and Pin 4 for XL2). This is the default position of the jumpers when shipped.

> **Note:** The 32.768 kHz crystal has to be selected for the *Bluetooth* examples to work.

If the jumpers are placed in the position I/O on connector **P4** (Pin 3 and Pin 5 for XL1 and Pin 4 and Pin 6 for XL2), P0.26 and P0.27 connects to the Motherboard and can be used for normal I/O.

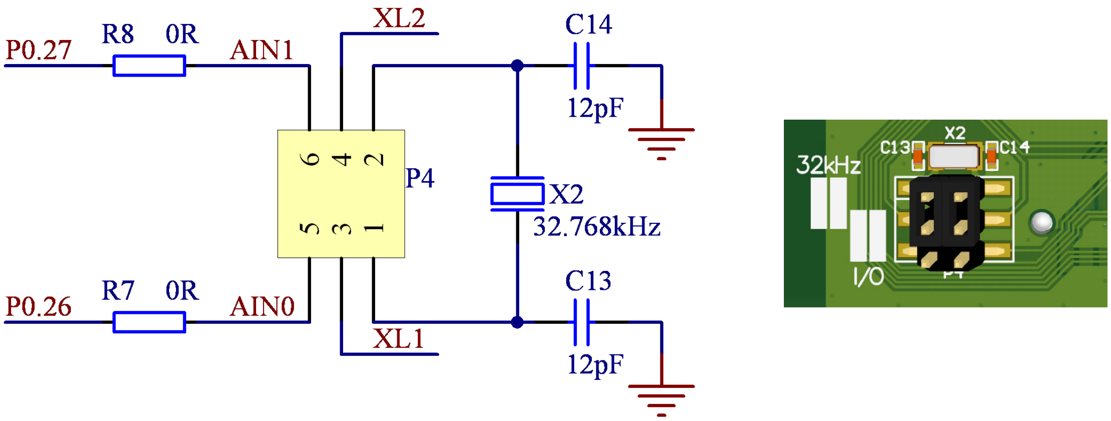


Figure 15 32.768kHz crystal circuit schematic and PCB

5.1.8 Current measurements

The current drawn from the nRF51822 device can be monitored using the Current Shunt Monitor (CSM), INA216 (U2). The gain of the CSM is set to 200 V/V for lowest possible drop voltage.

For current measurements using the CSM, the jumper on connector **K1** must be placed on Pin 2 and Pin 3. This is the default position of the jumpers when shipped. Current on the nRF51822 module is measured on connector **K2** as voltage proportional to the current.

For current measurements using the jumper **P7** on the Motherboard, the jumper on connector **K1** must be placed on Pin 1 and Pin 2.

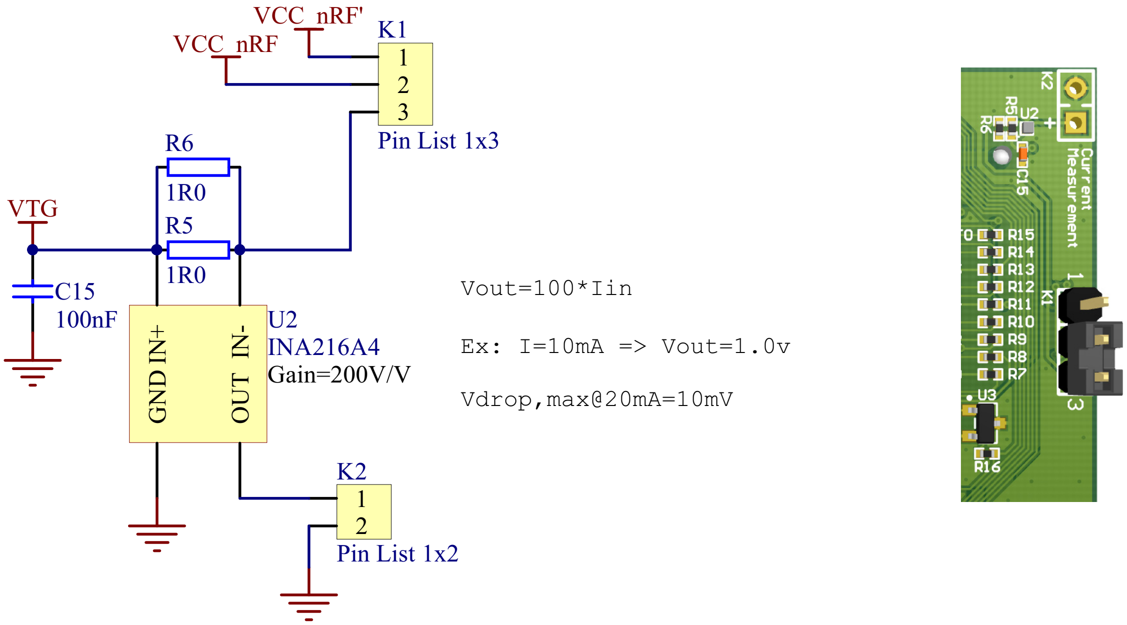


Figure 16 Current measurement circuit schematic and PCB

5.2 nRF51822 development dongle(PCA10000)

The nRF51822 development dongle (PCA10000) can be used as a development platform for nRF51822. It features an on-board programming and debugging solution from SEGGER. In addition to radio communication, the nRF51822 device can communicate with a computer through a virtual COM port provided by the SEGGER chip. The PCA10000 can be loaded with *Bluetooth* Low Energy Master Emulator firmware that when combined with the Master Control Panel, gives you a peer device for nRF51822 that you can use to test the wireless connection.

Note: PCA10000 can be reprogrammed if overwritten.

5.2.1 Key features

The PCA10000 has the following key features:

- nRF51822 IC
- *Bluetooth* low energy compatible
- 2.4 GHz compatible with nRF24L devices
- Virtual COM port

5.2.2 Hardware pictures

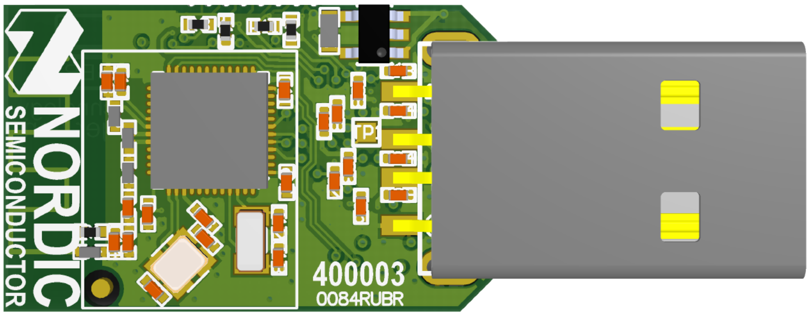


Figure 17 *PCA10000 top side*

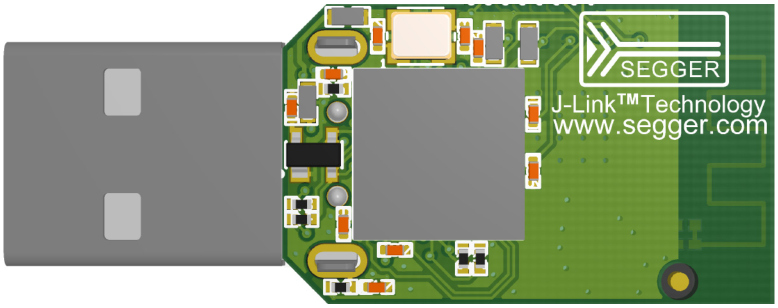


Figure 18 *PCA10000 bottom side*

5.2.3 Block diagram

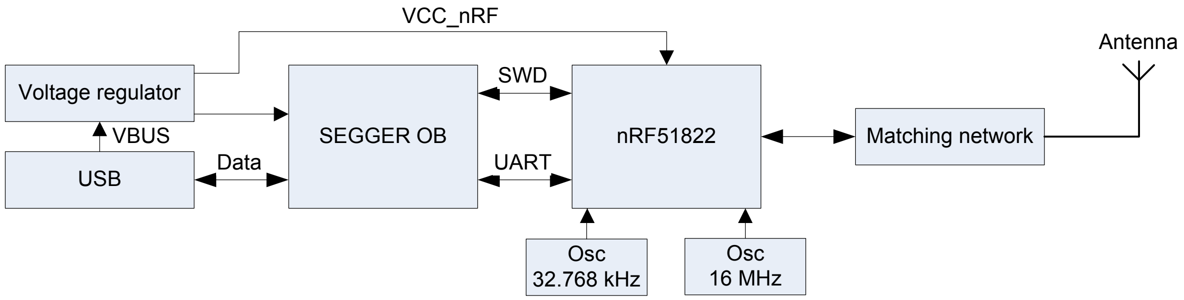


Figure 19 *PCA10000 block diagram*

Figure 20

nRF51822 development dongle		Segger IC
GPIO	**UART**	**UART**
P0.00	RTS	CTS
P0.01	TXD	RXD
P0.02	CTS	RTS
P0.03	RXD	TXD

Table 2 *nRF51822 development dongle UART configuration*

6 Flash programming and application development

nRF51822 is shipped without preprogrammed software. This gives you the option of developing your application directly onto the chip or alternatively, by using our S110 SoftDevice, which is a Bluetooth® low energy peripheral protocol stack solution. For more information, see the S110 SoftDevice Specification.

In this chapter we describe how to program the S110 SoftDevice or another application hex file onto the nRF51822 chip and also how to erase it.

If you want to start developing on the nRF51822 chip without using the S110 SoftDevice see ***section 6.1.4 on page 30***.

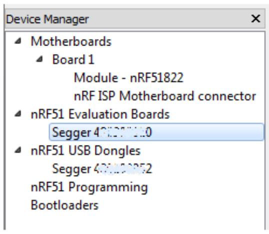
6.1 Programming and erasing flash using nRFgo Studio

Use nRFgo Studio to program a SoftDevice or application hex file onto the nRF51822 chip or to erase the flash content on a chip that has been programmed.

> **Note:** For details on memory organization and protection see the nRF51 Series Reference Manual.

6.1.1 Selecting a board to program

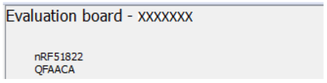
1. Open nRFgo Studio.
2. Select which board to program or erase. This is done in the Device Manager pane in nRFgo Studio.



Device Manager ✕

- Motherboards
  - Board 1
    - Module - nRF51822
    - nRF ISP Motherboard connector
- nRF51 Evaluation Boards
  - Segger [redacted]
- nRF51 USB Dongles
  - Segger [redacted]
- nRF51 Programming
- Bootloaders

3. The nRF51822 Evaluation Boards (PCA10001) and nRF51822 Development Dongle (PAC10000) hardware will show up under the respective sections ***nRF51 Evaluation Boards*** and ***nRF51 USB Dongles***. The devices are identified by the Segger serial number.
4. Select the board directly by clicking on the Segger module listed. The selected board is identified with board type, Segger serial number, and the nRF51 chip.



Evaluation board - xxxxxxx

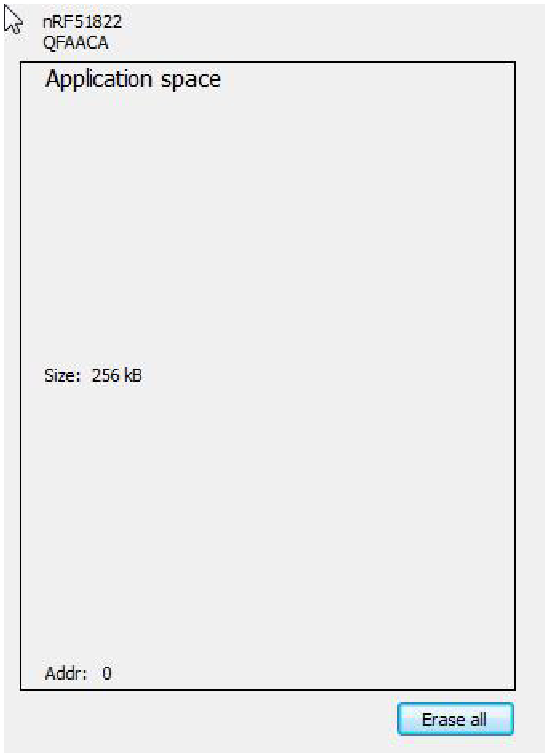
nRF51822
QFAACA

5. The nRF51822 Development Kit modules (PCA10004/ PCA10005) cannot be selected directly. These boards must be selected through the J-Link debugger connected to them. To do this, you either select the actual Module located under ***Motherboard > Boardx*** or by clicking on nRF51 Programming (ref fig “Device Manager”).
6. Select the debugger you want to use from the list (called **Segger to use)** of available J-Link debuggers.

6.1.2 Identifying the nRF51 chip and chip content

When you select a board, nRFgo Studio identifies the nRF51 chip and how its memory is organized. The following chip and memory information is displayed:

- **Identifying the nRF51 chip:** Identifies the chip by name and variant code (for example, nRF51822 QFAACA). If the debugger is not connected to the chip or he debugger has a problem communicating with the chip it will show the following message "*No device detected, did you connect the Segger correctly to a board?*".
- **Memory organization:** Shows how the flash memory is divided, whether into one or two regions and the size of each region. A SoftDevice will always be located at Region 0. You can see the memory size used by the SoftDevice and how much memory is remaining for use.
- **Memory read back protection:** Shows how the read back protection is set. The two possible options are read back protection on Region 0 or read back protection of the whole flash memory. If there is only one region the option is read back protection on (All) or off.
- **Identifying the SoftDevice:** nRFgo Studio tries to identify the firmware located in the chip at Region 0. For the firmware that it recognizes it prints the ID (in clear test) for the unrecognized firmware it prints the FWID number.



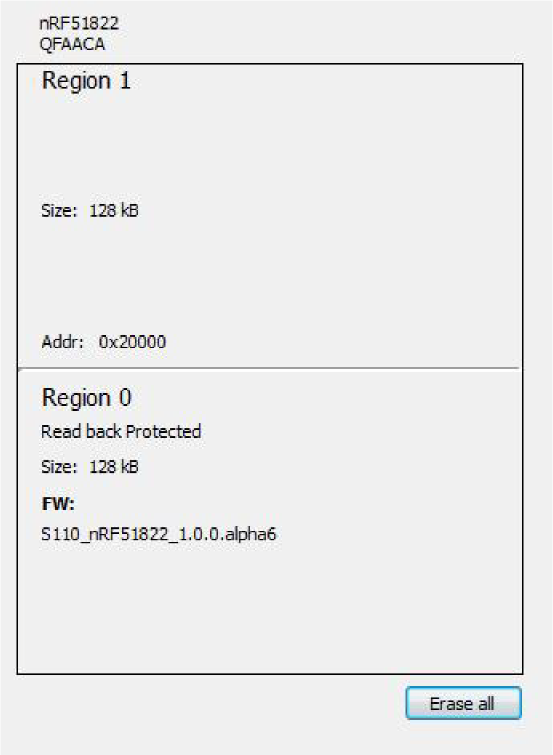
nRF51822
QFAACA

Application space

Size: 256 kB

Addr: 0

Figure 21 *Flash memory organized as one region without read back protection*



nRF51822
QFAACA

Region 1

Size: 128 kB

Addr: 0x20000

Region 0

Read back Protected

Size: 128 kB

FW:

S110_nRF51822_1.0.0.alpha6

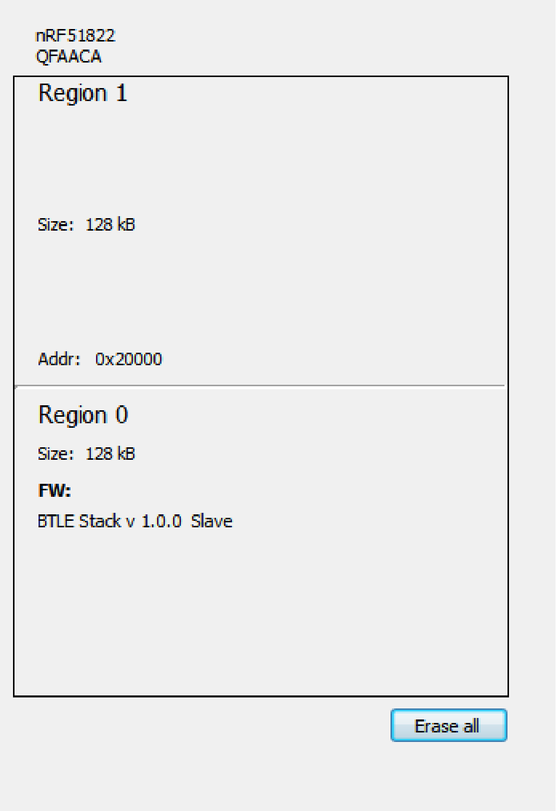
Figure 22 Flash memory organized in two regions with the BTLE stack located a Region 0 and with read back protection on region 0

6.1.3 Erasing All

You should use the Erase All function in the following three situations:

- You have a chip that is programmed with a SoftDevice but you want to remove it and have a blank chip.
- You have programmed an application on top of a SoftDevice and used the option “Lock entire chip from read back”. Once you have performed Erase All, both the application and the SoftDevice will be erased.
- You have programmed an application on a clean chip using nRFgo Studio with the option “Lock entire chip from read back”.

To use the Erase All fucntion, follow the steps in ***section 6.1.1 on page 27***. Then click **Erase all**.



nRF51822
QFAACA

Region 1

Size: 128 kB

Addr: 0x20000

Region 0

Size: 128 kB

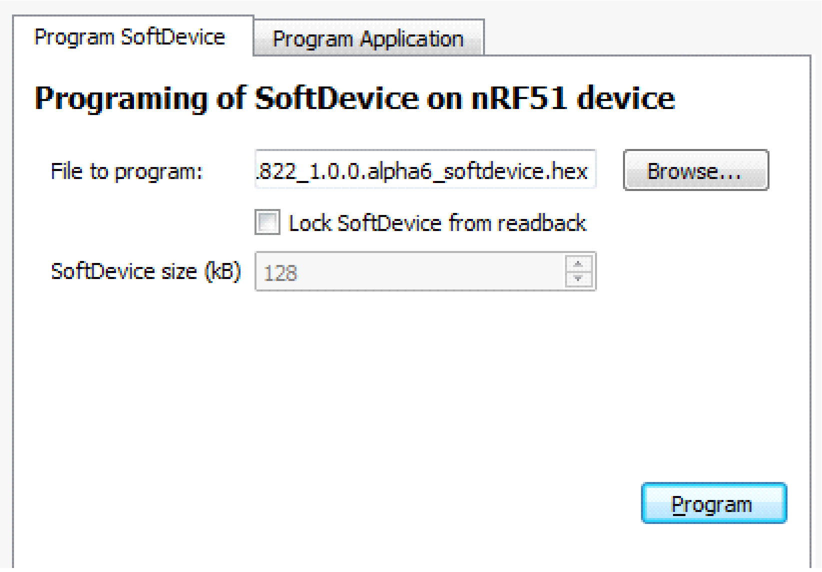
FW:

BTLE Stack v 1.0.0 Slave

Figure 23 *Erasing the SoftDevice*

6.1.4 Programming a SoftDevice

This function lets you program the SoftDevice into the chip.



Programing of SoftDevice on nRF51 device

File to program: .822_1.0.0.alpha6_softdevice.hex

□ Lock SoftDevice from readback

SoftDevice size (kB) 128

1. Follow the steps in ***section 6.1.1 on page 27*** and then select the **Program SoftDevice** tab.
2. Click **Browse** and select the hex file to program.
3. Select whether to enable or disable read back protection of Region 0.
4. Set the SoftDevice size. This sets the size of the flash region 0 and will not be available if the size is defined by the hex file.

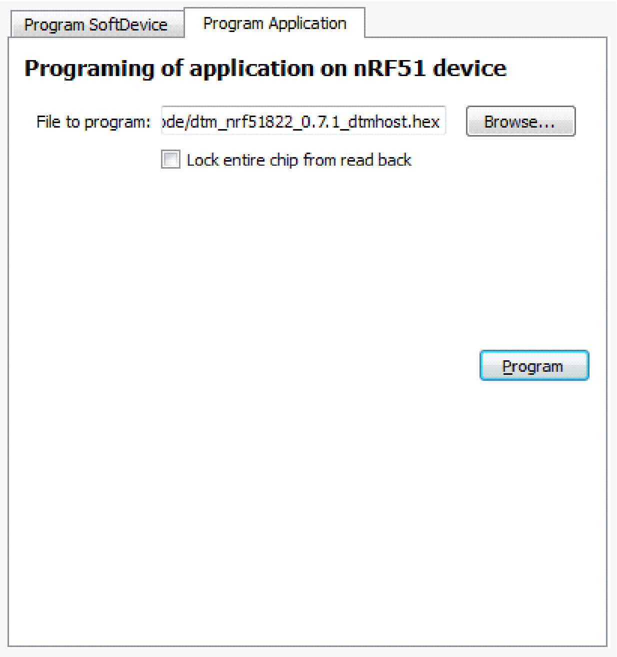
Note: The S110 SoftDevice can be downloaded from www.nordicsemi.com by logging into your MyPage account and entering the product key printed in the Development Kit.

6.1.5 Programming an application

This function lets you program an application onto the chip.

Before nRFgo Studio starts programming it verifies that the hex file matches the actual memory configuration, for example, that an application that requires a SoftDevice is being programmed onto a chip that contains a SoftDevice If it matches it continues with the programming, if not it stops the programming and returns an error message.

This programming will not set up any memory Regions.



Programing of application on nRF51 device

File to program: ›de/dtm_nrf51822_0.7.1_dtmhost.hex

□ Lock entire chip from read back

1. Follow the steps in ***section 6.1.1 on page 27*** and then select the **Program Application** tab.
2. Click **Browse** and select the hex file to program.
3. Select whether to enable or disable read back protection of the entire chip. If you enable read back protection, you will have to do an Erase All to reprogram the chip again.

Note: A chip that is programmed with the “Lock entire chip from read back” enabled will not work with a development toolchain. To make it work you must perform an Erase All.
The “Lock entire chip from read back” function can be used to prevent an accidental overwrite of the chip content.

6.2 Application development

The user application is compiled, linked, and downloaded independently from the SoftDevice. This means that developing and debugging on a chip pre-programmed with a SoftDevice is similar to that of a blank chip. The main differences are memory layout and the call stack size.

6.2.1 Configuring memory layout

Specific SoftDevice versions and stacks could have different requirements. Please review these before proceeding.

The applications vector table must be set up differently depending on whether it will run on a chip that is blank or pre-programmed with a SoftDevice.

The SoftDevice program area starts at address 0x0 and has a predefined size. The application start vector must be placed right after the SoftDevice. The available size has to be set so that it uses the remaining memory for the application. Similarly, the SoftDevice data area starts at the lowest RAM address. The application data area must be placed after the SoftDevice used data area.

The table below shows examples for setting up the start address and size depending on the code and data size used by the SoftDevice. The example is based on a chip with 256 kB of flash and 16 kB of RAM.

Device configuration	SoftDevice		Appl. code start address	Available flash	Appl. data start address	Available RAM
	Flash usage	RAM usage				
Blank chip	0 kB	0 kB	0x0	0x40000	0x2000.0000	0x4000
SoftDevice A	64 kB	2 kB	0x10000	0x30000	0x2000.0800	0x3800
SoftDevice B	128 kB	6 kB	0x20000	0x20000	0x2000.2000	0x2000

Table 3 *SoftDevice memory layout*

The Product Specification describes the total flash memory and RAM available in the device. The amount of flash memory and RAM used by the SoftDevice is described in the SoftDevice Specification.

There are two ways to configure the memory layout:

- Using Keil IDE
- Using a Scatter file

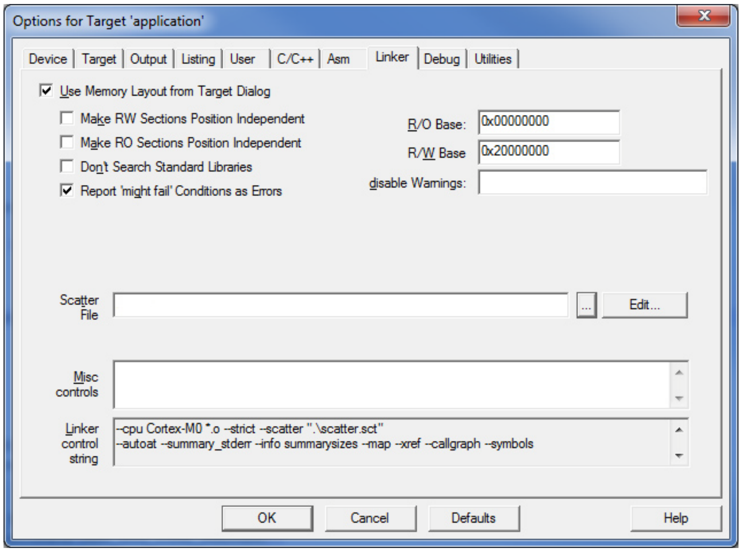
Note: The example code given by Nordic Semiconductor configures the memory layout in Keil IDE. Scatter file loading is not available when using the evaluation version of Keil IDE.

6.2.1.1 Memory layout configuration in Keil IDE

To access the Keil IDE memory layout:

1. Click the **Project** menu and select **Options for Target.** The Options for Target dialog box opens.
2. Select the **Linker** tab.

3. Check **Use memory layout from Target Dialog** and click **OK**.



☑ Use Memory Layout from Target Dialog

☐ Make RW Sections Position Independent

☐ Make RO Sections Position Independent

☐ Don't Search Standard Libraries

☑ Report 'might fail' Conditions as Errors

R/O Base: 0x00000000

R/W Base 0x20000000

disable Warnings:

Scatter File

Misc controls

Linker control string

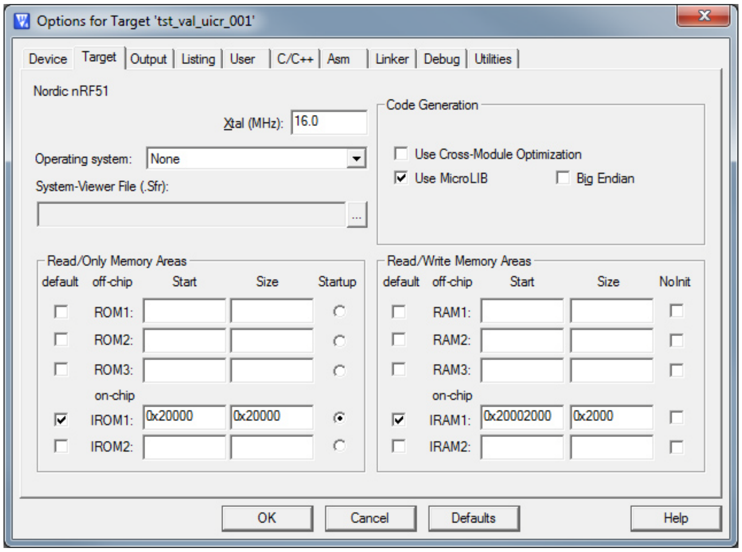
```
--cpu Cortex-M0 *.o --strict --scatter ".\scatter.sct"
--autoat --summary_stderr --info summarysizes --map --xref --callgraph --symbols
```

Figure 24 Keil memory layout

Values for **Start** and **Size** in **Read/Only Memory Areas** and **Read/Write Memory Areas** must be defined.

Below is an example configuration for an application using a chip with 256 kB of flash and 16 kB of RAM and a SoftDevice using 128 kB of flash and 8 kB of RAM (SoftDevice B described in ***Table 3 on page 32***).

- Base flash memory address 0x20000 and available flash size is 0x20000 (128 kB).
- Base RAM memory address 0x20002000 and available RAM size is 0x2000 (8 kB).



Nordic nRF51

Xtal (MHz): 16.0

Operating system: None

System-Viewer File (.Sfr):

Code Generation

☐ Use Cross-Module Optimization

☑ Use MicroLIB ☐ Big Endian

Read/Only Memory Areas

default	off-chip	Start	Size	Startup
☐	ROM1:			○
☐	ROM2:			○
☐	ROM3:			○
	on-chip			
☑	IROM1:	0x20000	0x20000	●
☐	IROM2:			○

Read/Write Memory Areas

default	off-chip	Start	Size	NoInit
☐	RAM1:			☐
☐	RAM2:			☐
☐	RAM3:			☐
	on-chip			
☑	IRAM1:	0x20002000	0x2000	☐
☐	IRAM2:			☐

Figure 25 Memory layout with example SoftDevice

Memory		Description
IROM1	Start	Specify the start address for the application code
	Size	Specify available flash size for the application code
IRAM1	Start	Specify start address for the application data
	Size	Specify available RAM size for the application data

Table 4 *Memory layout*

6.2.1.2 Scatter file

A scatter file specifies to the linker how to gather re-locatable files into load regions (areas in flash/ROM memory). It also informs the linker where, at run time, data (and possibly code) should be located into execution regions. Read/write data may have to be copied from flash to RAM (if compile time initialized) or zeroed, during the boot process. This is done by a library function called scatter loader, which works according to the descriptions set up by the scatter file in the flash/ROM.

Note: The scatter file feature is not available with a Keil Lite installation.

The application scatter file must specify addresses for the load region(s) that are within the physical address range of the flash memory, but not overlap the address range used by the SoftDevice. Similarly, the RAM execution region addresses must be within the physical RAM available, and not overlap the RAM area used by the SoftDevice.

The application interrupt vector should be located at the first available flash address on the application. To locate the reset vector at the beginning of the available flash to the application, the assembler file defining the reset vector declares an area RESET. That definition can then be used in the scatter file as shown in the example below. A template assembler file defining the reset vector is provided in the nRF51 SDK.

Note: The Product Specification describes the total flash memory and RAM available in the device. The amount of flash memory and RAM used by the SoftDevice is described in the SoftDevice Specification.

In the following scatter file example, the flash size of the device is assumed to be 256 kB (0x00040000) and the RAM size of the device is assumed to be 16 kB (0x00004000). RAM is always located from execution address 0x20000000. The example will assume SoftDevice B described above will be used. The range of flash from 0 to 0x00020000 (128 kB) is used by the SoftDevice protocol stack. The range of RAM from 0x20000000 to 0x20002000 (8 kB) is used by the SoftDevice protocol stack.

An application may define multiple load and execution regions if it is desired.

```
Example Scatter file
; *************************************************************
; *** Scatter-Loading Description File                      ***
; *************************************************************
; Scatter file for a program IN APPLICATION SPACE,
; accessing the SoftDevice B through SVCs.
; Make sure that neither Load Region addresses below 0x00020000 nor
; RAM Execution Region addresses below 0x20002000 are used -
; Those regions are reserved for the SoftDevice (protocol stack).

LR_IROM1 0x00020000 0x000020000  {    ; load region size_region
  ER_IROM1 0x00020000 0x000020000  {  ; load address = execution address
   *.o (RESET, +First)
   *(InRoot$$Sections)
   .ANY (+RO)
  }

  RW_IRAM1 0x20002000 0x00002000 { ; RW data
    .ANY (+RW +ZI)
  }
}
```

6.2.2 Shared call stack

The user application shares the call stack with the SoftDevice if the SoftDevice is loaded on the chip. The application must reserve enough memory for both itself and the SoftDevice in the call stack. Call stack size required by the SoftDevice varies between devices and protocol stack versions, and is supplied in the SoftDevice Specification.

The user application sets its call stack size plus the amount needed by the SoftDevice. It then writes the stack pointer at the first address of the application Reset Vector.

> **Note:** Using Keil with the ARMCC toolchain, the call-stack size can be set using the `Stack_Size` definitions in your projects startup file, typically `arm_startup_nrf51.s`.

```
Stack_Size      EQU     0x400 ; The application call-stack size + protocol
call-stack size

                AREA    STACK, NOINIT, READWRITE, ALIGN=3

Stack_Mem       SPACE   Stack_Size

__initial_sp
```

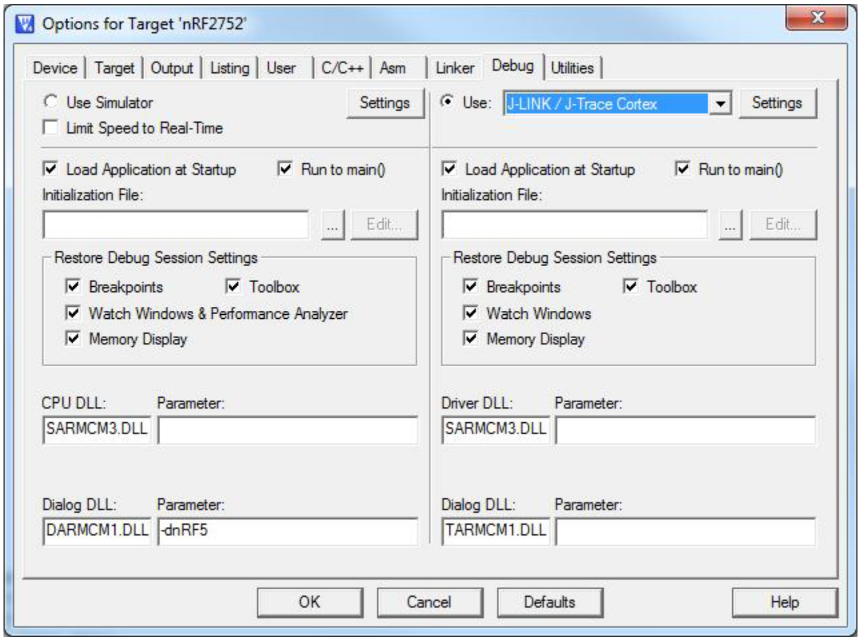
6.2.3 Debugger configuration

Project files delivered in the SDK are configured and ready for download and debugging. If a new application project is used, the debugger must be properly configured. To configure the debugger:

1. In Keil, select **Options for Target** from the Project menu. The Options for Target dialog box appears.
2. Select the **Debug** tab.
3. Apply the **Use** option and select the J-Link/J-Trace debugger from the list.
4. Set **Driver DLL** to SARMCM3.DLL.
5. Set **Dialog DLL** to TARMCM1.DLL.

Other options can be selected as needed. For a proper debugging experience the following are advised:

- Breakpoints
- Load Application at Startup
- Memory Display
- Toolbox
- Watch Windows



Options for Target 'nRF2752'

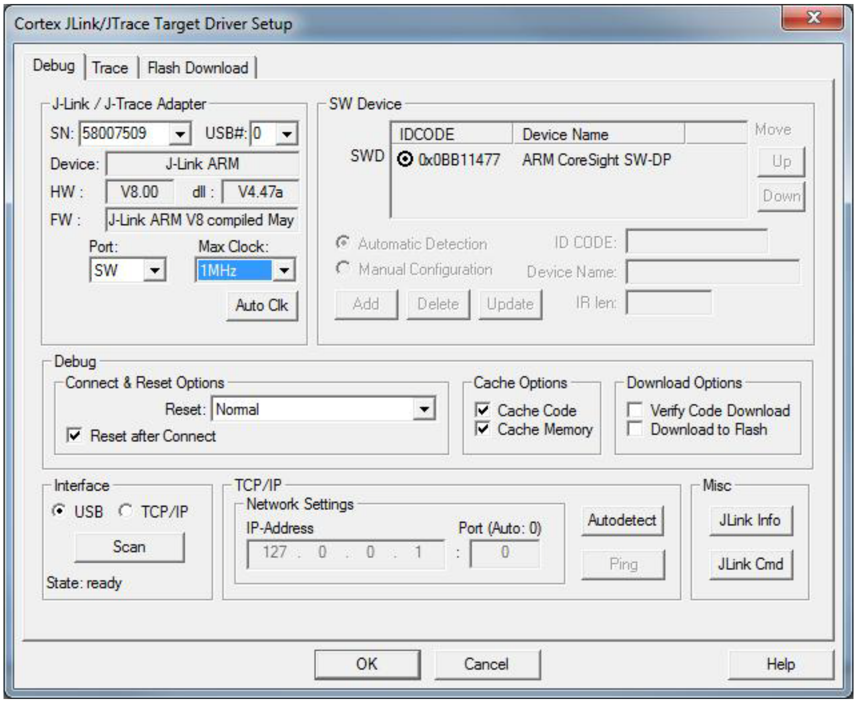
○ Use Simulator
□ Limit Speed to Real-Time
☑ Load Application at Startup ☑ Run to main()
Initialization File:
Restore Debug Session Settings
☑ Breakpoints ☑ Toolbox
☑ Watch Windows & Performance Analyzer
☑ Memory Display
CPU DLL: SARMCM3.DLL Parameter:
Dialog DLL: DARMCM1.DLL Parameter: -dnRF5

● Use: J-LINK / J-Trace Cortex
☑ Load Application at Startup ☑ Run to main()
Initialization File:
Restore Debug Session Settings
☑ Breakpoints ☑ Toolbox
☑ Watch Windows
☑ Memory Display
Driver DLL: SARMCM3.DLL Parameter:
Dialog DLL: TARMCM1.DLL Parameter:

Figure 26 *Debugger options*

6. Click the **Settings** button next to the Use field. Information about debugging protocol and maximum speed needs to be provided.
7. In the **Port** drop-down, **SW** must be selected.
8. In **Max Clock** the maximum speed for the debugging port cannot be exceeded (1 MHz).

A proper configuration is shown in the figure below.



Cortex JLink/JTrace Target Driver Setup

Debug | Trace | Flash Download

J-Link / J-Trace Adapter
SN: 58007509 USB#: 0
Device: J-Link ARM
HW: V8.00 dll: V4.47a
FW: J-Link ARM V8 compiled May
Port: SW Max Clock: 1MHz
Auto Clk

SW Device

	IDCODE	Device Name
SWD	⊙ 0x0BB11477	ARM CoreSight SW-DP

Move Up Down

● Automatic Detection ID CODE:
○ Manual Configuration Device Name:
Add Delete Update IR len:

Debug
Connect & Reset Options
Reset: Normal
☑ Reset after Connect

Cache Options
☑ Cache Code
☑ Cache Memory

Download Options
☐ Verify Code Download
☐ Download to Flash

Interface
● USB ○ TCP/IP
Scan
State: ready

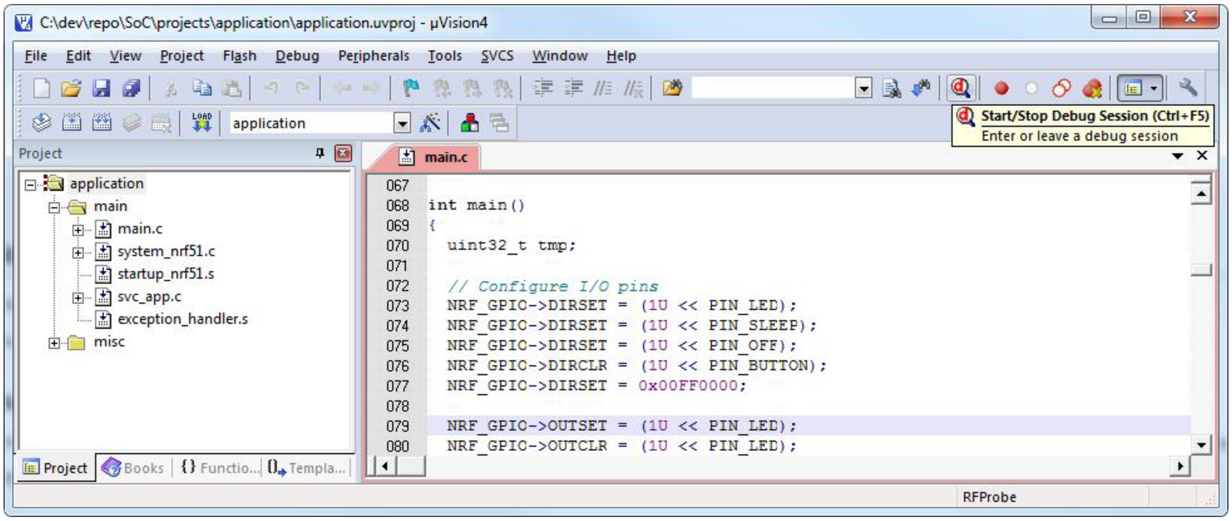
TCP/IP
Network Settings
IP-Address 127 . 0 . 0 . 1
Port (Auto: 0) 0
Autodetect
Ping

Misc
JLink Info
JLink Cmd

OK Cancel Help

Figure 27 Target driver setup

Debugging is initiated by clicking **Start/Stop Debug Session** in the Keil IDE.



C:\dev\repo\SoC\projects\application\application.uvproj - µVision4

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
application
Start/Stop Debug Session (Ctrl+F5)
Enter or leave a debug session
Project
application
main
main.c
system_nrf51.c
startup_nrf51.s
svc_app.c
exception_handler.s
misc
Project Books Functio... Templa...
main.c
RFProbe

```
int main()
{
  uint32_t tmp;

  // Configure I/O pins
  NRF_GPIO->DIRSET = (1U << PIN_LED);
  NRF_GPIO->DIRSET = (1U << PIN_SLEEP);
  NRF_GPIO->DIRSET = (1U << PIN_OFF);
  NRF_GPIO->DIRCLR = (1U << PIN_BUTTON);
  NRF_GPIO->DIRSET = 0x00FF0000;

  NRF_GPIO->OUTSET = (1U << PIN_LED);
  NRF_GPIO->OUTCLR = (1U << PIN_LED);
```

Figure 28 Debugger initiation

Note: If you are using Keil v4.53 or earlier, the default **JLinkSettings.ini** file created automatically by the debugger in the project root is not adequate for debugging and downloading. Copy the **JLinkSettings.ini** file, present in any Nordic Semiconductor project, into the root folder before debugging. The file includes the proper settings for use with nRF51 series devices.

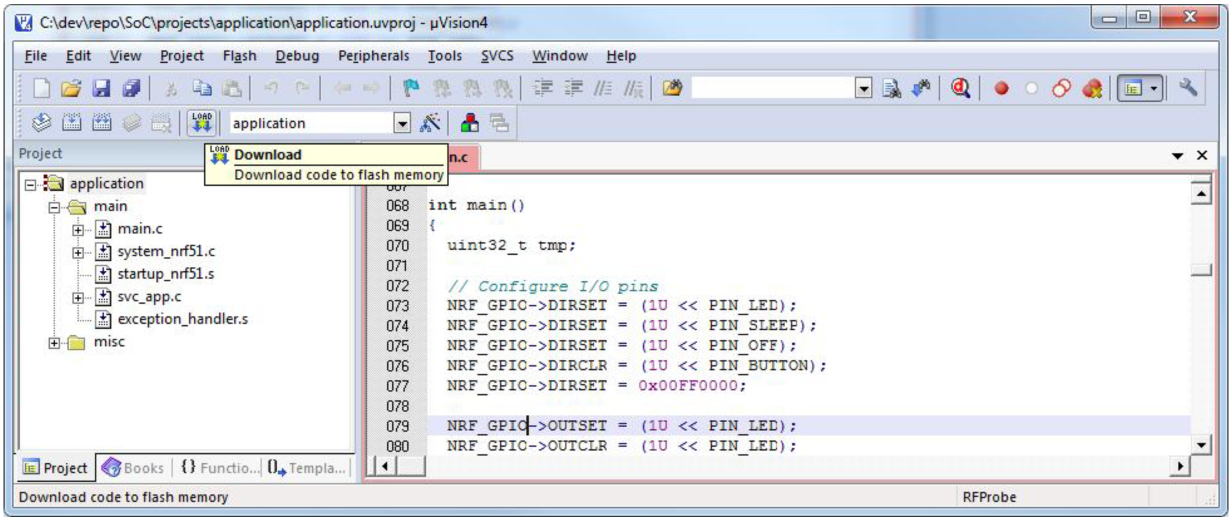
6.2.4 Limitation when debugging on a chip with a SoftDevice

When a SoftDevice is installed in a device, there are certain limitations when debugging.

6.2.5 Programming the device

To guarantee the correct functionality of the SoftDevice, the microcontroller includes a Memory Protection Unit that prevents access to certain resources. The debugger will read this area as 0x0000 (no operation instruction).The flash area occupied by the SoftDevice is write and erase protected. When the SoftDevice radio stack is enabled, the Memory Protection Unit implements a write protection to certain peripherals used by the protocol stack. Protected peripherals are described in the SoftDevice specification.

Once the debugger is properly configured and the application code compiled and linked, downloading the application can be easily done using the Keil IDE download button.



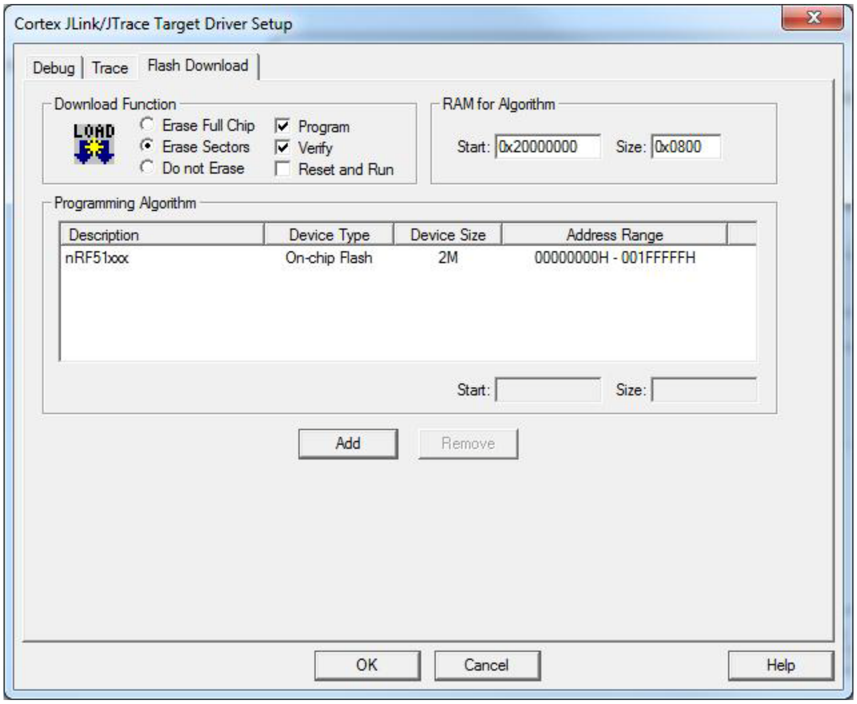
```
int main()
{
  uint32_t tmp;

  // Configure I/O pins
  NRF_GPIO->DIRSET = (1U << PIN_LED);
  NRF_GPIO->DIRSET = (1U << PIN_SLEEP);
  NRF_GPIO->DIRSET = (1U << PIN_OFF);
  NRF_GPIO->DIRCLR = (1U << PIN_BUTTON);
  NRF_GPIO->DIRSET = 0x00FF0000;

  NRF_GPIO->OUTSET = (1U << PIN_LED);
  NRF_GPIO->OUTCLR = (1U << PIN_LED);
```

Figure 29 *Download using Keil IDE*

To configure the download:

1. Select **Options for Target** in the Project menu.
2. Select the **Utilities** tab in the Options for Target dialog box.
3. Click the **Settings** button to configure.
4. Select the **Program** check box.
5. Choose your desired erase option.
6. Select the download algorithm used by Keil IDE.
7. Click **Add** and select the nRF51xxx algorithm from the list.

Note: The nRF51xxx algorithm is installed automatically during the SDK installation. This algorithm is a generic nRF51 series algorithm, which provides download capabilities to all series devices up to 2 MB of flash.



Cortex JLink/JTrace Target Driver Setup

Download Function

○ Erase Full Chip
● Erase Sectors
○ Do not Erase

☑ Program
☑ Verify
□ Reset and Run

RAM for Algorithm

Start: 0x20000000 Size: 0x0800

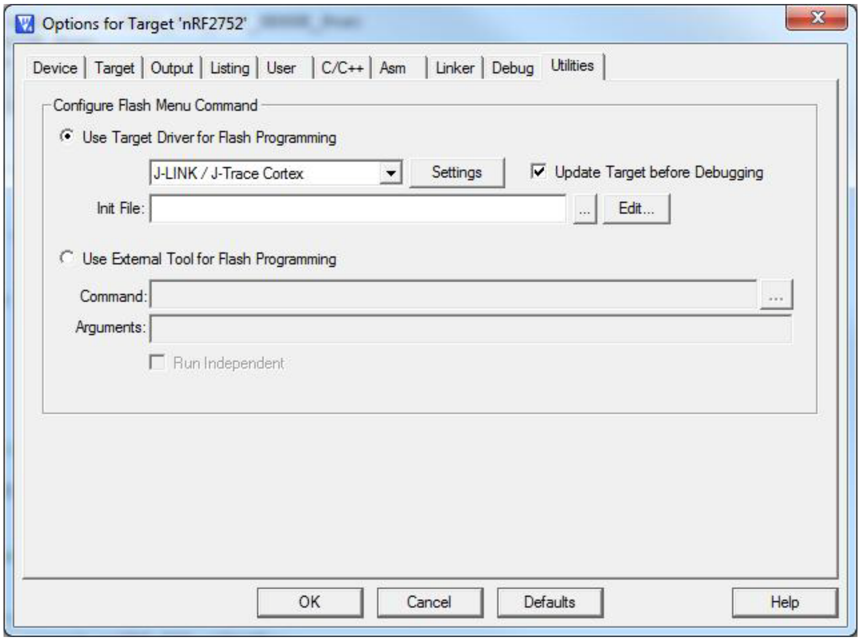
Programming Algorithm

Description	Device Type	Device Size	Address Range
nRF51xxx	On-chip Flash	2M	00000000H - 001FFFFFH

Start: Size:

Figure 30 Selecting erase option

In the Utilities tab, **Use Target Driver for Flash Programming** option must be selected and the available debugger chosen from the list as shown in ***Figure 31***.



Options for Target 'nRF2752' [redacted]

Configure Flash Menu Command

● Use Target Driver for Flash Programming

J-LINK / J-Trace Cortex

☑ Update Target before Debugging

Init File:

○ Use External Tool for Flash Programming

Command:

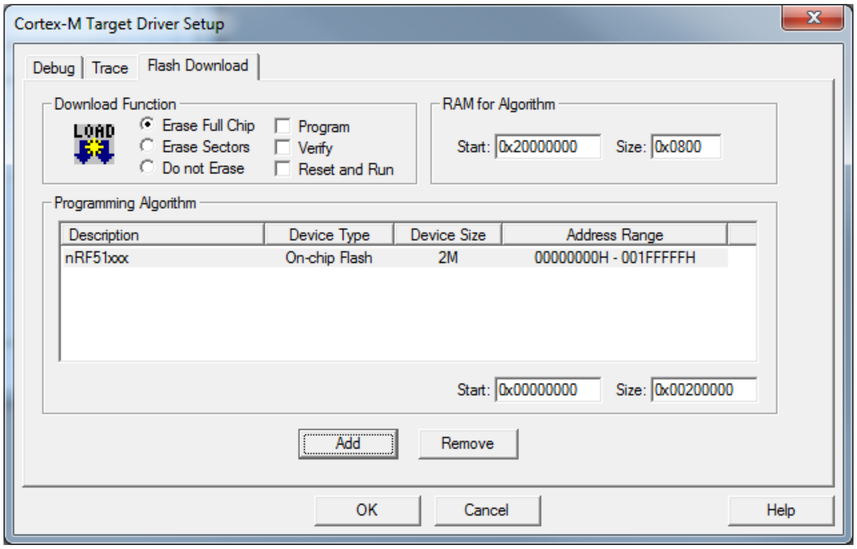
Arguments:

□ Run Independent

Figure 31 Debugger selection

6.2.6 Erasing the device

The flash area available for user application can be erased using the download function in Keil IDE. It can be configured in the Settings menu in the Utilities tab. **Erase Full Chip** must be selected. **Program** and **Verify** should be unchecked. A normal download procedure will erase the device application flash area.



Download Function

● Erase Full Chip
○ Erase Sectors
○ Do not Erase
□ Program
□ Verify
□ Reset and Run

RAM for Algorithm

Start: 0x20000000 Size: 0x0800

Programming Algorithm

Description	Device Type	Device Size	Address Range
nRF51xxx	On-chip Flash	2M	00000000H - 001FFFFFH

Start: 0x00000000 Size: 0x00200000

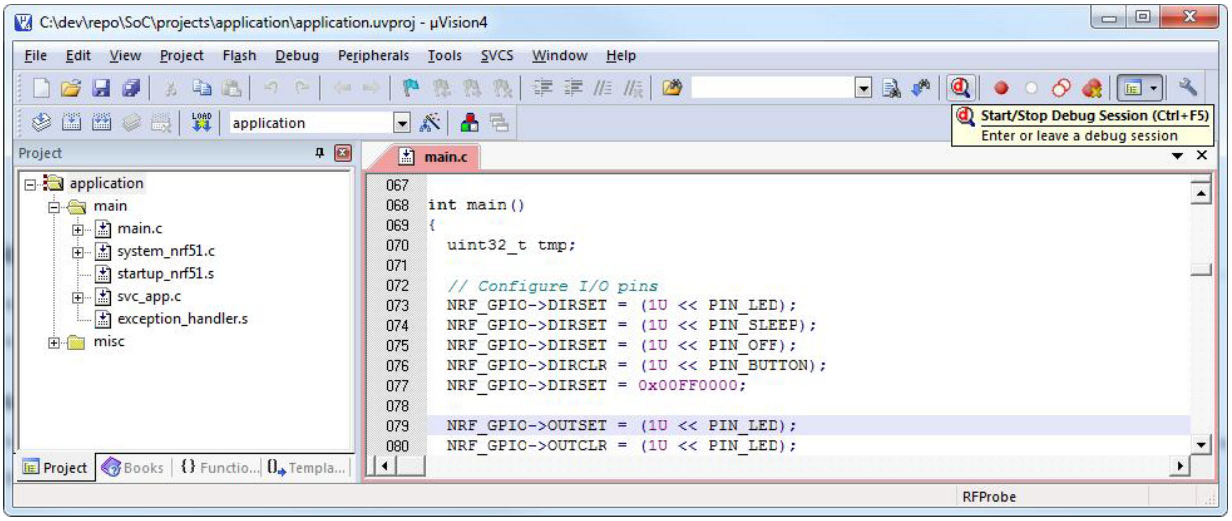
Figure 32 Erasing

To erase the whole device, including the SoftDevice, refer to section ***6.1.3 on page 29*** for instructions.

7 Debugging the nRF51822

For debugging with SEGGER J-Link, see ***Appendix A: "Installing drivers and configuring KEIL projects for the SEGGER debugger"*** on page 53. For general information of how to debug using Keil µVision IDE, we refer to online documentation from Keil at http://www.keil.com/uvision/debug.asp.

Configure the debugger in the **Project** menu and selecting **Options for Target**. Select **Debug**. When you have set this up, you can enter debugging mode by clicking the **Debug** button or **CTRL+F5**.


C:\dev\repo\SoC\projects\application\application.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
application
Start/Stop Debug Session (Ctrl+F5)
Enter or leave a debug session
Project
application
main
main.c
system_nrf51.c
startup_nrf51.s
svc_app.c
exception_handler.s
misc
Project
Books
{} Functio...
Templa...
main.c
RFProbe

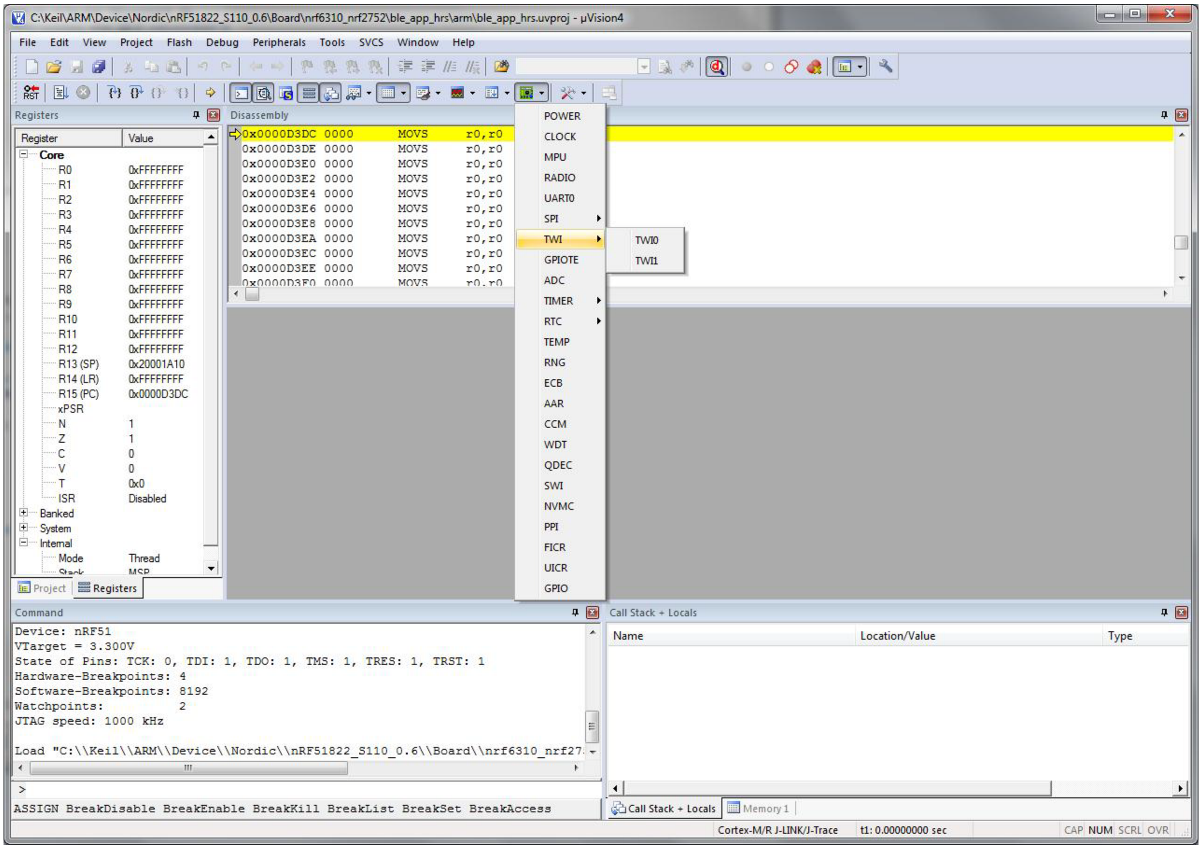
```
int main()
{
  uint32_t tmp;

  // Configure I/O pins
  NRF_GPIO->DIRSET = (1U << PIN_LED);
  NRF_GPIO->DIRSET = (1U << PIN_SLEEP);
  NRF_GPIO->DIRSET = (1U << PIN_OFF);
  NRF_GPIO->DIRCLR = (1U << PIN_BUTTON);
  NRF_GPIO->DIRSET = 0x00FF0000;

  NRF_GPIO->OUTSET = (1U << PIN_LED);
  NRF_GPIO->OUTCLR = (1U << PIN_LED);
```

Figure 33 Start debugging mode

7.1 nRF51 debug features and precautions

7.1.1 System Viewer Windows

The System Viewer Windows enables access to view device peripheral contents, see ***Figure 34*** on page 42. Any values in the registers may be viewed from its respected window, as seen in ***Figure 35*** on page 43. More information can be found at: http://www.keil.com/uvision/db_view_sysview.asp.



C:\Keil\ARM\Device\Nordic\nRF51822_S110_0.6\Board\nrf6310_nrf2752\ble_app_hrs\arm\ble_app_hrs.uvproj - µVision4

Register	Value
Core	
R0	0xFFFFFFFF
R1	0xFFFFFFFF
R2	0xFFFFFFFF
R3	0xFFFFFFFF
R4	0xFFFFFFFF
R5	0xFFFFFFFF
R6	0xFFFFFFFF
R7	0xFFFFFFFF
R8	0xFFFFFFFF
R9	0xFFFFFFFF
R10	0xFFFFFFFF
R11	0xFFFFFFFF
R12	0xFFFFFFFF
R13 (SP)	0x20001A10
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x0000D3DC
xPSR	
N	1
Z	1
C	0
V	0
T	0x0
ISR	Disabled
Banked	
System	
Internal	
Mode	Thread
Stack	MSP

```
0x0000D3DC 0000      MOVS     r0,r0
0x0000D3DE 0000      MOVS     r0,r0
0x0000D3E0 0000      MOVS     r0,r0
0x0000D3E2 0000      MOVS     r0,r0
0x0000D3E4 0000      MOVS     r0,r0
0x0000D3E6 0000      MOVS     r0,r0
0x0000D3E8 0000      MOVS     r0,r0
0x0000D3EA 0000      MOVS     r0,r0
0x0000D3EC 0000      MOVS     r0,r0
0x0000D3EE 0000      MOVS     r0,r0
0x0000D3F0 0000      MOVS     r0,r0
```

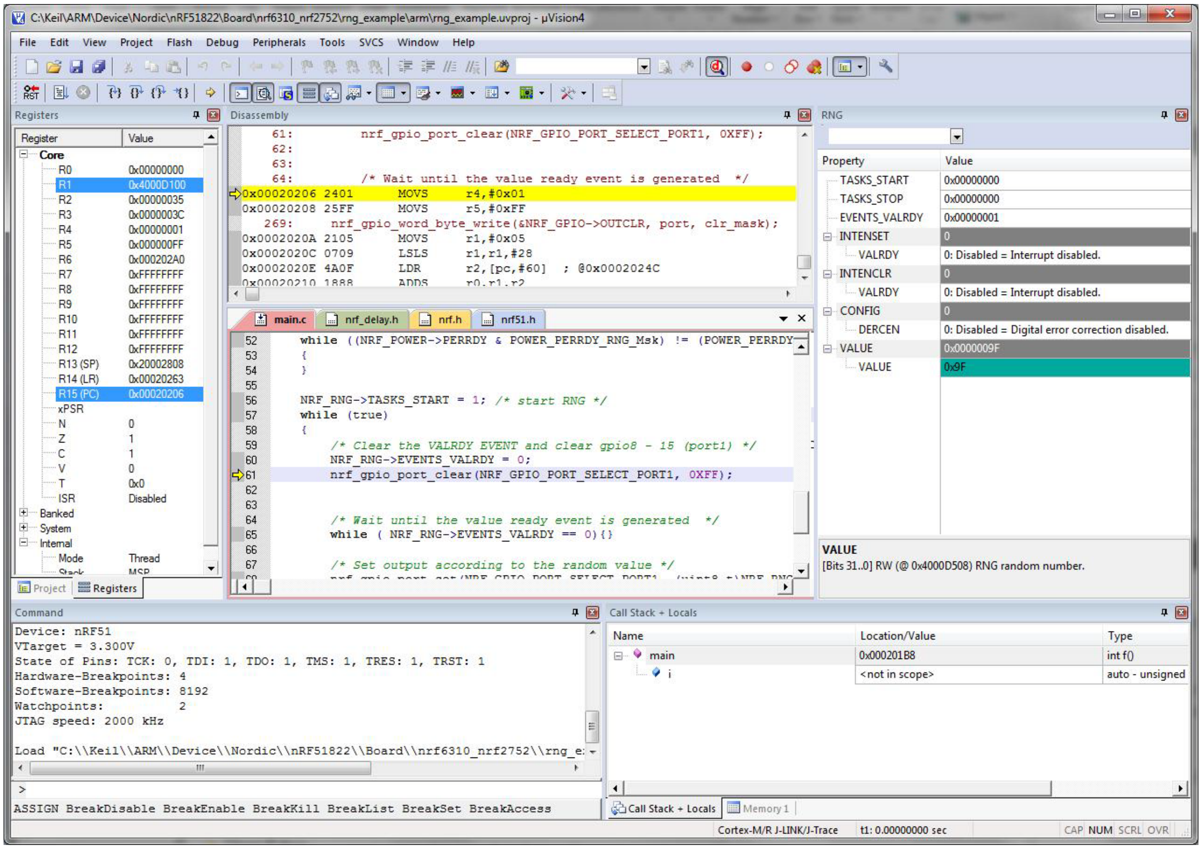
```
Device: nRF51
VTarget = 3.300V
State of Pins: TCK: 0, TDI: 1, TDO: 1, TMS: 1, TRES: 1, TRST: 1
Hardware-Breakpoints: 4
Software-Breakpoints: 8192
Watchpoints:          2
JTAG speed: 1000 kHz

Load "C:\\Keil\\ARM\\Device\\Nordic\\nRF51822_S110_0.6\\Board\\nrf6310_nrf27
```

```
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess
```

Name	Location/Value	Type

Cortex-M/R J-LINK/J-Trace t1: 0.00000000 sec CAP NUM SCRL OVR

Figure 34 *System Viewer Windows*



Register	Value
Core	
R0	0x00000000
R1	0x4000D100
R2	0x00000035
R3	0x0000003C
R4	0x00000001
R5	0x000000FF
R6	0x000202A0
R7	0xFFFFFFFF
R8	0xFFFFFFFF
R9	0xFFFFFFFF
R10	0xFFFFFFFF
R11	0xFFFFFFFF
R12	0xFFFFFFFF
R13 (SP)	0x20002808
R14 (LR)	0x00020263
R15 (PC)	0x00020206
xPSR	
N	0
Z	1
C	1
V	0
T	0x0
ISR	Disabled
Banked	
System	
Internal	
Mode	Thread
Stack	MSP

```
    61:         nrf_gpio_port_clear(NRF_GPIO_PORT_SELECT_PORT1, 0XFF);
    62:
    63:
    64:         /* Wait until the value ready event is generated  */
0x00020206 2401      MOVS     r4,#0x01
0x00020208 25FF      MOVS     r5,#0xFF
   269:     nrf_gpio_word_byte_write(&NRF_GPIO->OUTCLR, port, clr_mask);
0x0002020A 2105      MOVS     r1,#0x05
0x0002020C 0709      LSLS     r1,r1,#28
0x0002020E 4A0F      LDR      r2,[pc,#60]  ; @0x0002024C
0x00020210 1888      ADDS     r0,r1,r2
```

```
    while ((NRF_POWER->PERRDY & POWER_PERRDY_RNG_Msk) != (POWER_PERRDY
    {
    }

    NRF_RNG->TASKS_START = 1; /* start RNG */
    while (true)
    {
        /* Clear the VALRDY EVENT and clear gpio8 - 15 (port1) */
        NRF_RNG->EVENTS_VALRDY = 0;
        nrf_gpio_port_clear(NRF_GPIO_PORT_SELECT_PORT1, 0XFF);


        /* Wait until the value ready event is generated  */
        while ( NRF_RNG->EVENTS_VALRDY == 0){}

        /* Set output according to the random value */
```

Property	Value
TASKS_START	0x00000000
TASKS_STOP	0x00000000
EVENTS_VALRDY	0x00000001
INTENSET	0
VALRDY	0: Disabled = Interrupt disabled.
INTENCLR	0
VALRDY	0: Disabled = Interrupt disabled.
CONFIG	0
DERCEN	0: Disabled = Digital error correction disabled.
VALUE	0x0000009F
VALUE	0x9F

VALUE
[Bits 31..0] RW (@ 0x4000D508) RNG random number.

```
Device: nRF51
VTarget = 3.300V
State of Pins: TCK: 0, TDI: 1, TDO: 1, TMS: 1, TRES: 1, TRST: 1
Hardware-Breakpoints: 4
Software-Breakpoints: 8192
Watchpoints:          2
JTAG speed: 2000 kHz

Load "C:\\Keil\\ARM\\Device\\Nordic\\nRF51822\\Board\\nrf6310_nrf2752\\rng_e:
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess
```

Name	Location/Value	Type
main	0x000201B8	int f()
i	<not in scope>	auto - unsigned

Figure 35 System viewer window of RNG register

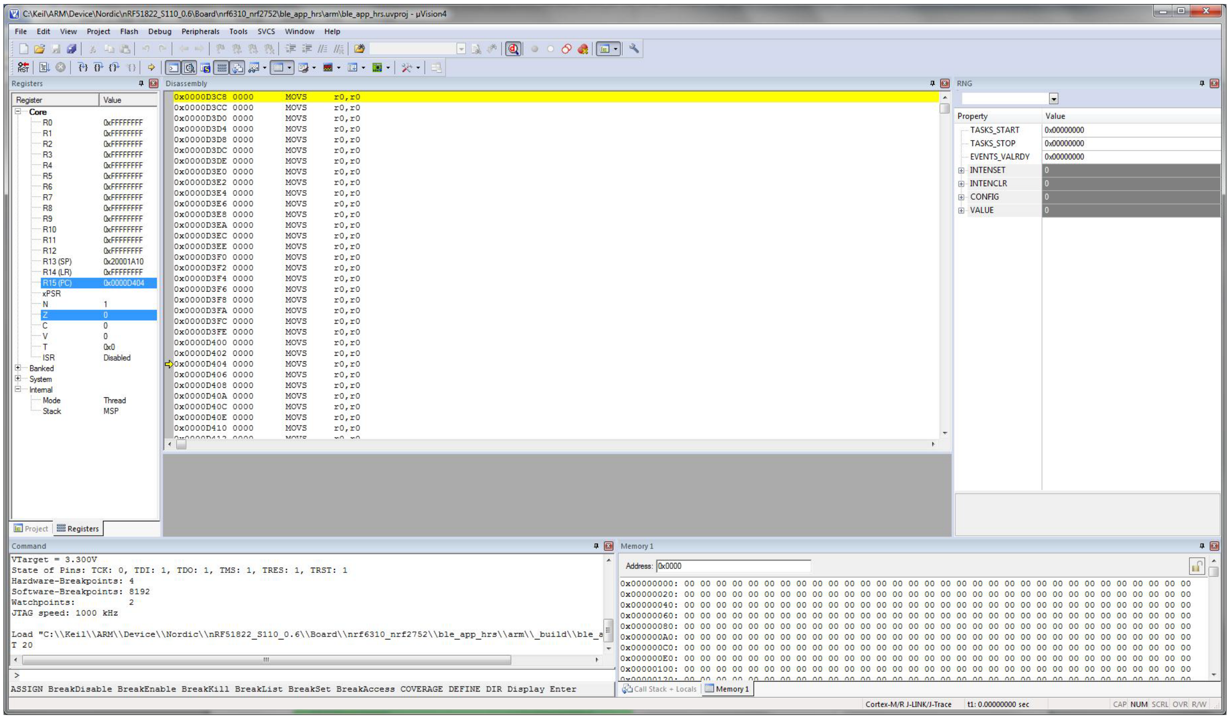
7.1.2 Debugging an application when a read back protected SoftDevice is present

Debugging applications with a SoftDevice present behaves as a described in http://www.keil.com/uvision/debug.asp, except when program counter is in Region 0 on a SoftDevice with read back protection enabled.

Information on the SoftDevice configuration and memory resource mapping can be found in the SoftDevice Specification document.

Code words from addresses in the protected area will always return zero to the debugger and thus debugging the application- as seen in *7.1.3* on page 45. Any values in peripheral registers that are restricted or blocked by the SoftDevice will be invisible to the debugger as well. See the SoftDevice Specification for more information.

Note: Stepping into instructions that interact with the protected area (SVC calls) may use a lot of time before returning to the next instruction (sometimes greater than 10 minutes). We recommend avoiding single stepping to the protected area, but instead set break point right after SVC calls while debugging and run the application to the actual breakpoint see *Figure 37* on page 45. The “step over” function **F10** may also be used instead to step over SVC calls to avoid delays when entering read back protected area.



C:\Keil\ARM\Device\Nordic\nRF51822_S110_0.6\Board\nrf6310_nrf2752\ble_app_hrs\arm\ble_app_hrs.uvproj - µVision4

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help

Registers

Register	Value
Core	
R0	0xFFFFFFFF
R1	0xFFFFFFFF
R2	0xFFFFFFFF
R3	0xFFFFFFFF
R4	0xFFFFFFFF
R5	0xFFFFFFFF
R6	0xFFFFFFFF
R7	0xFFFFFFFF
R8	0xFFFFFFFF
R9	0xFFFFFFFF
R10	0xFFFFFFFF
R11	0xFFFFFFFF
R12	0xFFFFFFFF
R13 (SP)	0x20001A10
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x0000D404
xPSR	
N	1
Z	0
C	0
V	0
T	0x0
ISR	Disabled
Banked	
System	
Internal	
Mode	Thread
Stack	MSP

Disassembly

```
0x0000D3C8 0000      MOVS     r0,r0
0x0000D3CC 0000      MOVS     r0,r0
0x0000D3D0 0000      MOVS     r0,r0
0x0000D3D4 0000      MOVS     r0,r0
0x0000D3D8 0000      MOVS     r0,r0
0x0000D3DC 0000      MOVS     r0,r0
0x0000D3DE 0000      MOVS     r0,r0
0x0000D3E0 0000      MOVS     r0,r0
0x0000D3E2 0000      MOVS     r0,r0
0x0000D3E4 0000      MOVS     r0,r0
0x0000D3E6 0000      MOVS     r0,r0
0x0000D3E8 0000      MOVS     r0,r0
0x0000D3EA 0000      MOVS     r0,r0
0x0000D3EC 0000      MOVS     r0,r0
0x0000D3EE 0000      MOVS     r0,r0
0x0000D3F0 0000      MOVS     r0,r0
0x0000D3F2 0000      MOVS     r0,r0
0x0000D3F4 0000      MOVS     r0,r0
0x0000D3F6 0000      MOVS     r0,r0
0x0000D3F8 0000      MOVS     r0,r0
0x0000D3FA 0000      MOVS     r0,r0
0x0000D3FC 0000      MOVS     r0,r0
0x0000D3FE 0000      MOVS     r0,r0
0x0000D400 0000      MOVS     r0,r0
0x0000D402 0000      MOVS     r0,r0
0x0000D404 0000      MOVS     r0,r0
0x0000D406 0000      MOVS     r0,r0
0x0000D408 0000      MOVS     r0,r0
0x0000D40A 0000      MOVS     r0,r0
0x0000D40C 0000      MOVS     r0,r0
0x0000D40E 0000      MOVS     r0,r0
0x0000D410 0000      MOVS     r0,r0
0x0000D412 0000      MOVS     r0,r0
```

RNG

Property	Value
TASKS_START	0x00000000
TASKS_STOP	0x00000000
EVENTS_VALRDY	0x00000000
INTENSET	0
INTENCLR	0
CONFIG	0
VALUE	0

Command

```
VTarget = 3.300V
State of Pins: TCK: 0, TDI: 1, TDO: 1, TMS: 1, TRES: 1, TRST: 1
Hardware-Breakpoints: 4
Software-Breakpoints: 8192
Watchpoints:          2
JTAG speed: 1000 kHz

Load "C:\\Keil\\ARM\\Device\\Nordic\\nRF51822_S110_0.6\\Board\\nrf6310_nrf2752\\ble_app_hrs\\arm\\_build\\ble_a
T 20
```

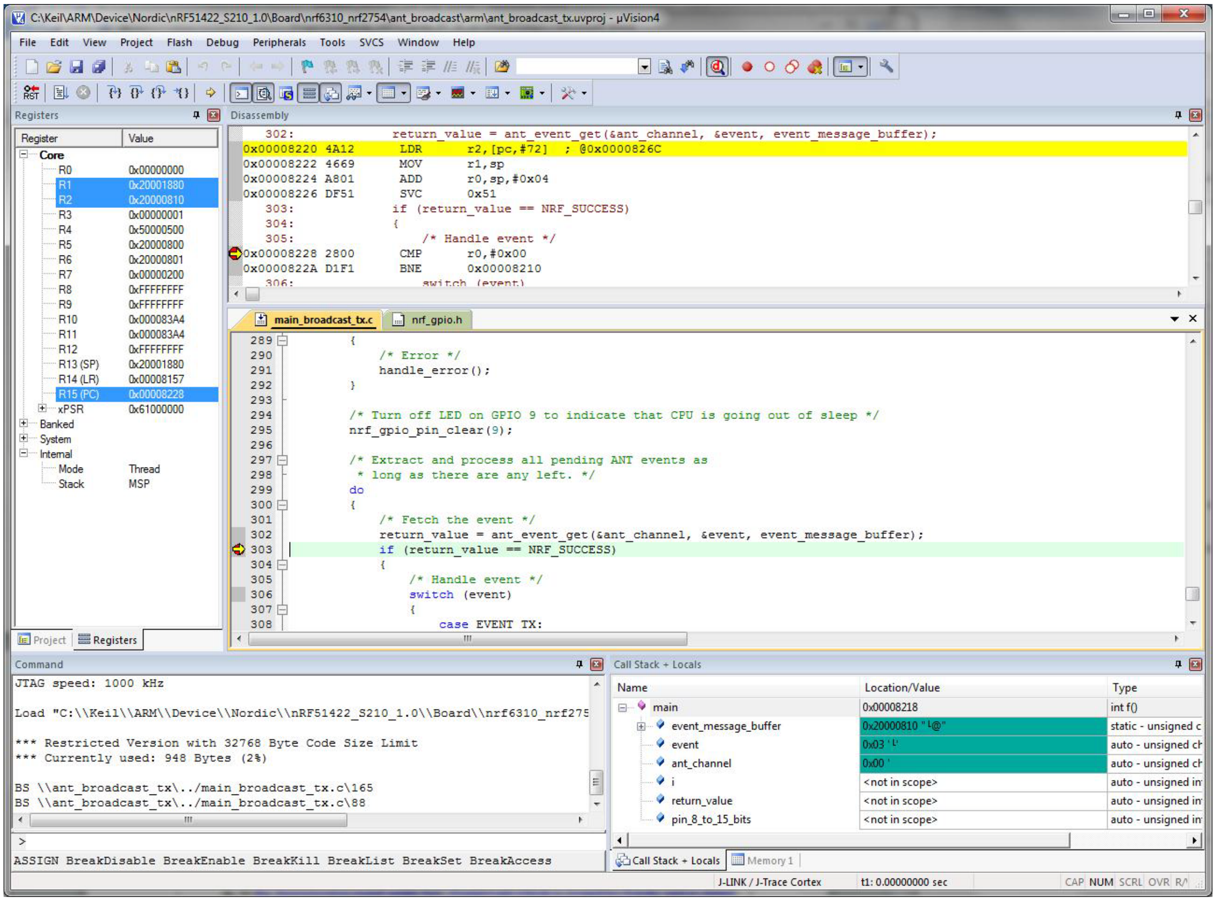
```
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE DEFINE DIR Display Enter
```

Memory 1

Address: 0x0000

```
0x00000000: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000000A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000000C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000000E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00000120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Figure 36 Debugger information for a setup with SoftDevice enabled in protected area



C:\Keil\ARM\Device\Nordic\nRF51422_S210_1.0\Board\nrf6310_nrf2754\ant_broadcast\arm\ant_broadcast_tx.uvproj - µVision4

Register	Value
Core	
R0	0x00000000
R1	0x20001880
R2	0x20000810
R3	0x00000001
R4	0x50000500
R5	0x20000800
R6	0x20000801
R7	0x00000200
R8	0xFFFFFFFF
R9	0xFFFFFFFF
R10	0x000083A4
R11	0x000083A4
R12	0xFFFFFFFF
R13 (SP)	0x20001880
R14 (LR)	0x00008157
R15 (PC)	0x00008228
xPSR	0x61000000
Banked	
System	
Internal	
Mode	Thread
Stack	MSP

```
   302:            return_value = ant_event_get(&ant_channel, &event, event_message_buffer);
0x00008220 4A12      LDR      r2,[pc,#72]  ; @0x0000826C
0x00008222 4669      MOV      r1,sp
0x00008224 A801      ADD      r0,sp,#0x04
0x00008226 DF51      SVC      0x51
   303:            if (return_value == NRF_SUCCESS)
   304:            {
   305:                /* Handle event */
0x00008228 2800      CMP      r0,#0x00
0x0000822A D1F1      BNE      0x00008210
   306:                switch (event)
```

```
        {
            /* Error */
            handle_error();
        }

        /* Turn off LED on GPIO 9 to indicate that CPU is going out of sleep */
        nrf_gpio_pin_clear(9);

        /* Extract and process all pending ANT events as
         * long as there are any left. */
        do
        {
            /* Fetch the event */
            return_value = ant_event_get(&ant_channel, &event, event_message_buffer);
            if (return_value == NRF_SUCCESS)
            {
                /* Handle event */
                switch (event)
                {
                    case EVENT_TX:
```

```
JTAG speed: 1000 kHz

Load "C:\\Keil\\ARM\\Device\\Nordic\\nRF51422_S210_1.0\\Board\\nrf6310_nrf275

*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 948 Bytes (2%)

BS \\ant_broadcast_tx\../main_broadcast_tx.c\165
BS \\ant_broadcast_tx\../main_broadcast_tx.c\88

>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess
```

Name	Location/Value	Type
main	0x00008218	int f()
event_message_buffer	0x20000810 "└@"	static - unsigned c
event	0x03 '└'	auto - unsigned ch
ant_channel	0x00 ''	auto - unsigned ch
i	<not in scope>	auto - unsigned in
return_value	<not in scope>	auto - unsigned in
pin_8_to_15_bits	<not in scope>	auto - unsigned in

J-LINK / J-Trace Cortex   t1: 0.00000000 sec   CAP NUM SCRL OVR R/

Figure 37 Setup with breakpoint after SVC call

7.1.3 Setting a breakpoint using SEGGER J-Link debugger

For SEGGER version 4.52c or earlier, if a breakpoint is set using the SEGGER J-Link debugger, while the system is running, the CPU will be halted for approximately 5 - 10 ms. The unavailability of the CPU during this time may cause the SoftDevice to get out of sync, leading to an invalid state. When this is discovered, the SoftDevice will assert. We encourage you to avoid setting breakpoints on a running system.

8 Testing the physical layer with Direct Test Mode

The Direct Test Mode (DTM) interface enables you to test the RF parameters/performance of the *Bluetooth* low energy radio design. It can be used for performance testing, tuning your prototypes, and compliance testing. This interface is compliant with the description in the *Bluetooth* Specification, Version 4.0, Volume 6, Part F.

DTM testing requires that you program the chip with the DTM application. The DTM is accessed through a dedicated UART interface on nRF51822 module and is only available in test mode. Please see the nRF51822 Preliminary Product Specification for information on the interface and how to enter test mode.

The DTM UART interface features:

- 1 stop bit
- 8 data bits
- 19200 baud rate
- No flow control (meaning no RTS/CTS)
- No parity

8.1 Kit setup for UART

The Direct Test Mode is accessed through the UART on the nRF51822 module. The following steps show how to get set up for DTM testing.

1. Mount one of the nRF51822 modules (either PCA10004 or PCA10005) onto the nRFgo Motherboard.
2. Connect a double cable, or two single cables, between **P15** and **P8** on the Motherboard as shown in ***Figure 38***. The UART data pins RXD and TXD will be present on the I/O port pins P0.1 and P0.3—make sure the RXD/TXD labels match for each wire.
3. The RS232 header (**P15**) is connected to the RS232 serial port interface (**J2**) through a RS232 converter. Connect a serial cable from RS232 to the serial port on your computer. Make sure switch **S11** is ON.
4. Connect the Motherboard to your computer using a USB cable.
5. Connect the J-Link Lite CortexM-9 to the nRF51822 module with a 10 pin connector; a USB cable connects the J-Link Lite CortexM-9 to your computer.

You are now ready to begin testing with Direct Test Mode in nRFgo Studio.

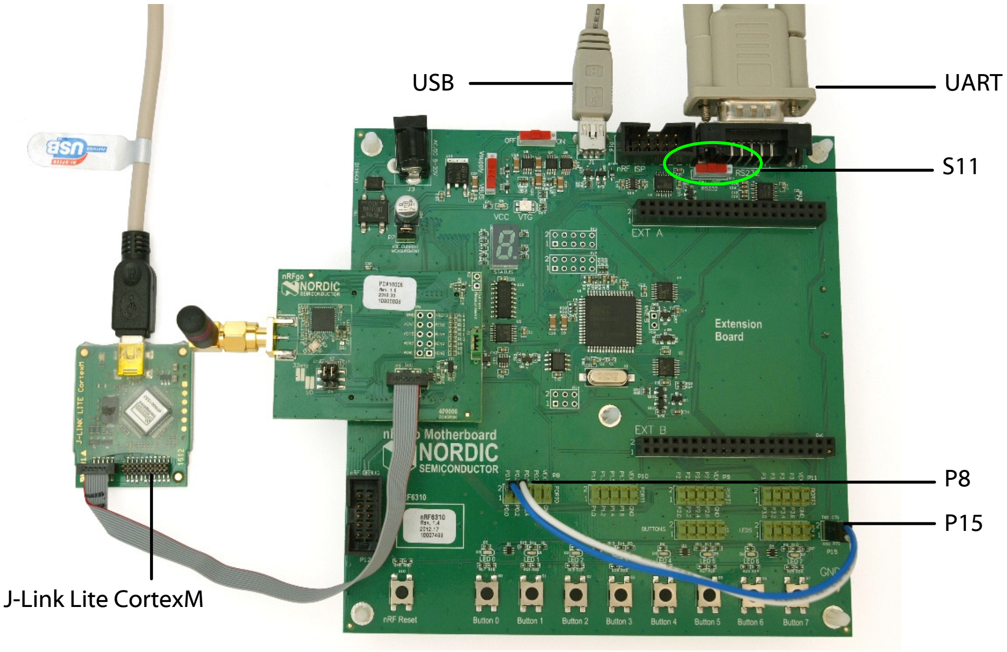


Figure 38 UART setup for DTM

P8	P15
P0.1	TXD
P0.3	TXD

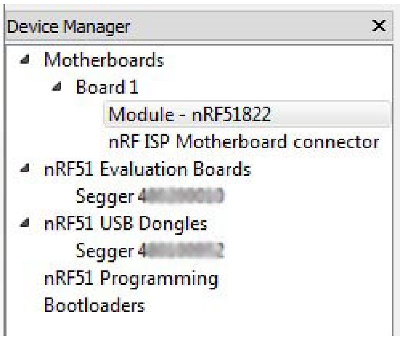
The DTM is designed for use with *Bluetooth* test equipment. If you don't have a *Bluetooth* tester you can access the interface using this Development Kit and nRFgo Studio to run the tests.

Note: In nRFgo Studio, the Program button under Direct Test Mode UART Interface will not work on the nRF51 platform.

8.2 Programming the nRF51822 module

The nRF51822 module has to be programmed with the DTM application found in the nRF518 SDK using nRFgo Studio.

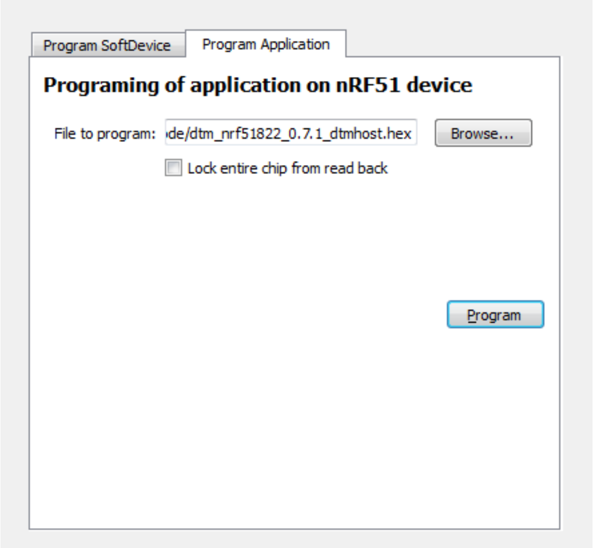
1. Open nRFgo Studio. The nRF51822 module that is plugged in to the Motherboard will be listed in the Device Manager pane under Motherboard.



Device Manager ✕

- Motherboards
  - Board 1
    - Module - nRF51822
    - nRF ISP Motherboard connector
- nRF51 Evaluation Boards
  - Segger 4[redacted]
- nRF51 USB Dongles
  - Segger 4[redacted]
- nRF51 Programming
- Bootloaders

2. Select the nRF51822 module. This opens the programming interface that is used to program the device.
3. Select the **Program Application** tab. Verify that the correct J-Link Lite CortexM debugger is selected by checking that the serial number on the debugger matches the **Segger to use**. Use the drop-down menu to select a different device.



Programing of application on nRF51 device

File to program: ›de/dtm_nrf51822_0.7.1_dtmhost.hex

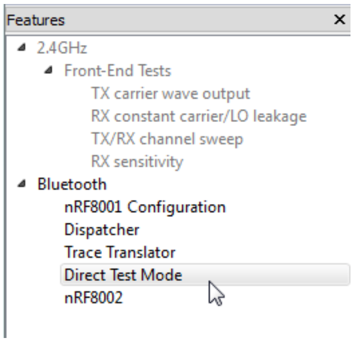
□ Lock entire chip from read back

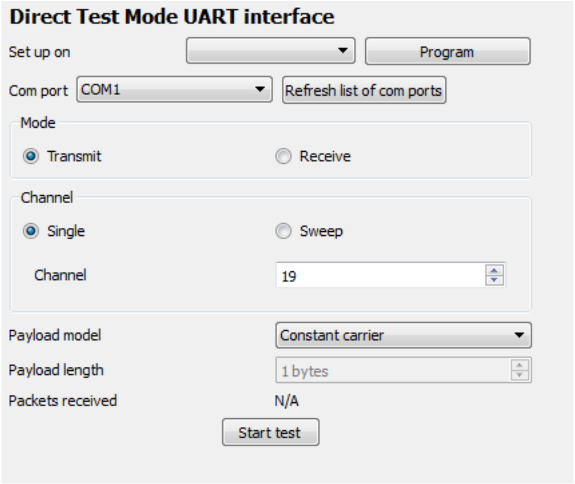
4. Make sure the chip is empty before programming the DTM application. If the chip already has the SoftDevice programmed onto it, you have to click **Erase ALL** to delete the chip.
5. To start programing click **Browse** and navigate to **<install folder>\nordic\nrf51822\board\nrf6310\ble\direct_test_mode**.
6. Select the HEX file **dtm_nrf51822**_<*version*>_**dtmhost.hex** and click **Open**.
7. nRFgo Studio is now ready to program the nRF51822 device with the given DTM application. Click **Program** to begin.

You are now ready to start DTM testing either using the Direct Test Mode UART interface included in nRFgo Studio or connecting 3rd party equipment.

8.3 DTM testing using nRFgo Studio

Select **Direct Test Mode** in the Feature pane underneath *Bluetooth* to start the Direct Test Mode UART interface in nRFgo Studio.





Direct Test Mode UART interface

Set up on: Program

Com port: COM1 — Refresh list of com ports

Mode: ● Transmit ○ Receive

Channel: ● Single ○ Sweep

Channel: 19

Payload model: Constant carrier

Payload length: 1 bytes

Packets received: N/A

Start test

Note: Set up on and Program are not functional when testing on the nRF51 platform.

For details on how to use the Direct Test Mode UART interface see the nRFgo Studio help file (or press **F1** when in Studio).

9 Software Development Kit

The nRF518 Software Development Kit (SDK) for the nRF51822 chip and the S110 SoftDevice enables you to develop applications for the following protocol stacks:

- *Bluetooth* low energy (using the S110_nRF51822 SoftDevice)
- Proprietary 2.4 GHz, including Nordic's Gazell protocol
- Non-concurrent combinations of *Bluetooth* low energy and proprietary 2.4 GHz

9.1 Installing the nRF518 SDK

The nRF518 SDK is a part of the DK downloadable content available from your My Page account, see ***section 3.2 on page 12***. The SDK is downloaded as a MSI file (a windows installer) and is installed by running the application. When installing the SDK you can select: Keil MDK Support, Master Control Panel, and/or Custom install.

- **Keil MDK support** - installs Keil µVision 4 example project files, flash programming algorithm for J-Link debugger, and Nordic nRF51 series device database file for Keil.
- **Custom install** - installs a software archive to a customizable location.

Note: The Keil MDK Support option will only be available if you already have the Keil MDK toolchain installed.

10 Troubleshooting

The nRF51822 device on the nRF51822 module does not respond when I try to contact it. What has happened?

Verify that the jumper on connector **K3** on the nRF51822 module is set in the position Pin 2 and Pin 3.

When I connect multiple SEGGER J-Link debugger boards to the PC, µVision is not able to recognize them correctly.

This is a known limitation with µVision in MDK v4.53 or earlier that will be fixed on later versions. Upgrade to version 4.54 or later.

The debugger doesn't work.

Please refer to ***4.2.1 "nRFgo nRF51822 DK modules" on page 16***.

On my 32 bit Windows XP machine, I get an error message with code 2908 when reinstalling either the nRF514 or nRF518 SDK.

Installing or reinstalling either the nRF518 or the nRF514 SDK *after* the nRF518 SDK has been uninstalled will cause error message code 2908 during installation. Use the Task manager (**Ctrl**+**Shift**+**Esc**) to end the task **nRF514/8 SDK Setup**. Drivers included in nRF518 will still be installed (if they are not already installed).

The debugger seems to freeze while debugging.

If running a SoftDevice that has been programmed with the "Lock SoftDevice from Readback" enabled (see section ***section 6.1 on page 27***, the debugger will halt while stepping to an SVC instruction. You should set the breakpoint after the SVC instruction and run the application to the breakpoint, or step over any SVC instructions. See ***section 7.1.2 on page 43*** for more details.

Software gets out of sync while debugging.

Setting/modifying breakpoints on a running system using the SEGGER debugger will halt the CPU, which may result in software that is out of sync. You should avoid setting breakpoints while the system is running.

The drop-down menu in the Master Control Panel doesn't display any serial numbers. What has happened?

Verify that the Master Control Panel software and the drivers for Segger OB (JLinkCDCInstaller) have been installed and that the nRF51 development dongle (PCA10000) has been plugged into a USB port on your computer.

The Master Control Panel connects to the nRF51 Development Dongle (PCA10000) but reports "No response from master emulator" in the Log?

You haven't programmed the nRF51 Development Dongle with the Master Emulator Firmware before starting to use it. See **Scan for available Bluetooth low energy devices** in ***chapter 2 on page 5*** for details on how to program the Master Emulator Firmware.

My project used to work, but after trying out another project using the SoftDevice, it fails.

Ensure that the memory layout in your project matches the memory layout on the chip. See ***section 6.2.1 on page 32*** on how to set up memory configuration.

Appendix A: Installing drivers and configuring KEIL projects for the SEGGER debugger

The following describes the steps required to install the software and use the SEGGER J-Link Lite debugger with Keil µVision for nRF51 series devices based on J-Link software version 4.52b or later.

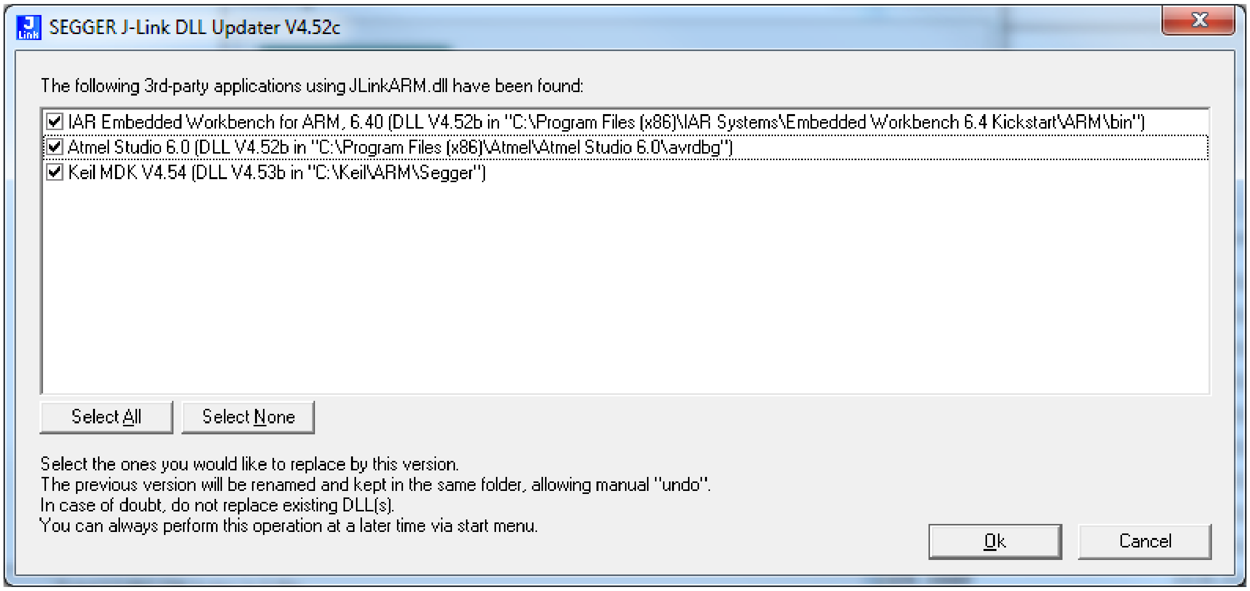
Prerequisite

You should have Keil µVision with ARM-MDK that is tested and working with MDK version 4.54.

> **Note:** All projects in the nRF518SDK are preset to work with the SEGGER debugger. Only the following step ***Download and install SEGGER drivers*** is needed.

Download and install SEGGER drivers

1. Download the latest SEGGER J-Link software and documentation pack from http://www.segger.com/jlink-software.html.
2. Download and run the J-Link Software (version 4.52b or later) and documentation pack for Windows from http://www.segger.com/jlink-software.html. The serial number from your SEGGER J-Link hardware is needed.
3. During installation you will be prompted to select the IDE that should be updated with the latest SEGGER DLLs. Check the box for **Keil MDK** and any other IDEs you want to use with SEGGER.



SEGGER J-Link DLL Updater V4.52c

The following 3rd-party applications using JLinkARM.dll have been found:

☑ IAR Embedded Workbench for ARM, 6.40 (DLL V4.52b in "C:\Program Files (x86)\IAR Systems\Embedded Workbench 6.4 Kickstart\ARM\bin")
☑ Atmel Studio 6.0 (DLL V4.52b in "C:\Program Files (x86)\Atmel\Atmel Studio 6.0\avrdbg")
☑ Keil MDK V4.54 (DLL V4.53b in "C:\Keil\ARM\Segger")

Select the ones you would like to replace by this version.
The previous version will be renamed and kept in the same folder, allowing manual "undo".
In case of doubt, do not replace existing DLL(s).
You can always perform this operation at a later time via start menu.

4. Go to http://www.segger.com/IDE_Integration_Keil.html#knownproblems for MDK v4.54. Download JL2CM3 and copy it to **<keil>/ARM/Segger**. This patch is necessary for the SEGGER debugger to work.

5. Plug in the J-Link Lite CortexM-9 module with USB cable. The LED will blink while the driver installation occurs. Wait until the LED is continually lit, without blinking

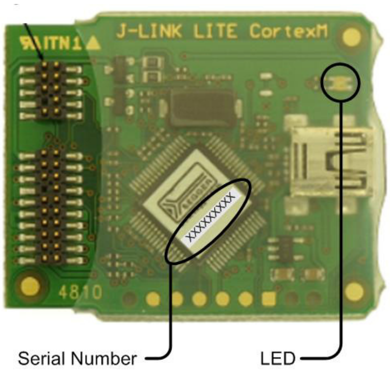


__Figure 39__ J-Link Lite CortexM-9 serial number location

Configuring KEIL projects for SEGGER debugger for first time use

Create **JLinkSettings.ini** file with the contents shown in ***Figure 45*** on page 58. The file **JLinkSettings.ini** should be saved in the same folder as Keil µVision project (uvproj) file.

1. Open Keil µVision IDE by double-clicking an example project file. The Target Options window will open.
2. Click the **Target Options** button on the toolbar or click **Project** menu and select **Options for Target**

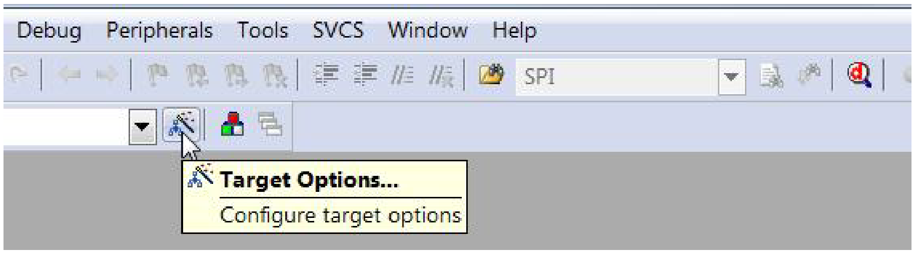
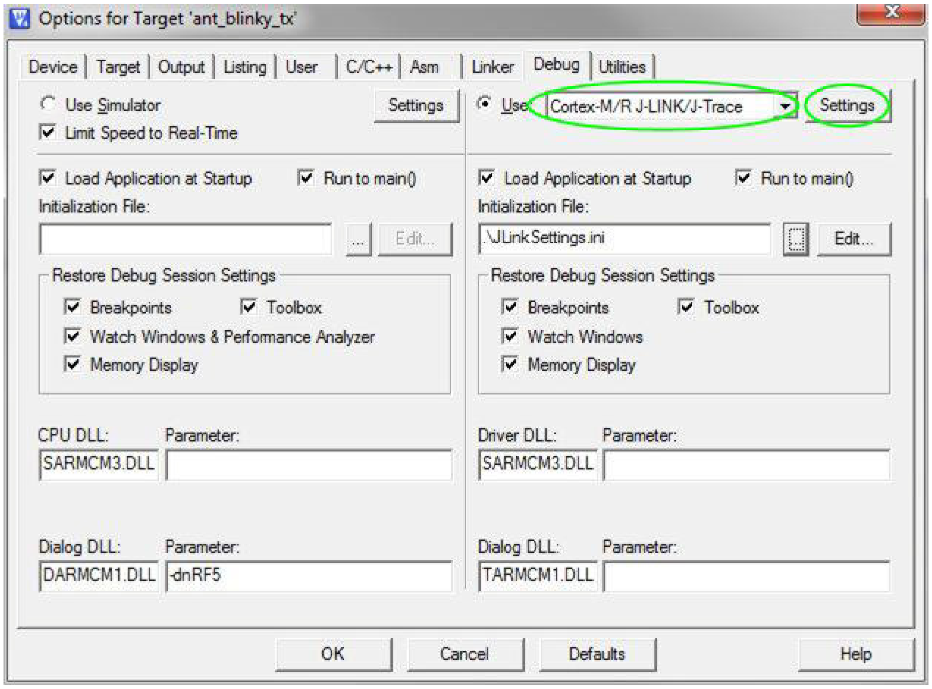


Figure 40 *Keil Target configuration*

3. Under the **Debug** tab in the Use list, select **J-LINK / J-Trace Cortex** option as shown in ***Figure 41***.



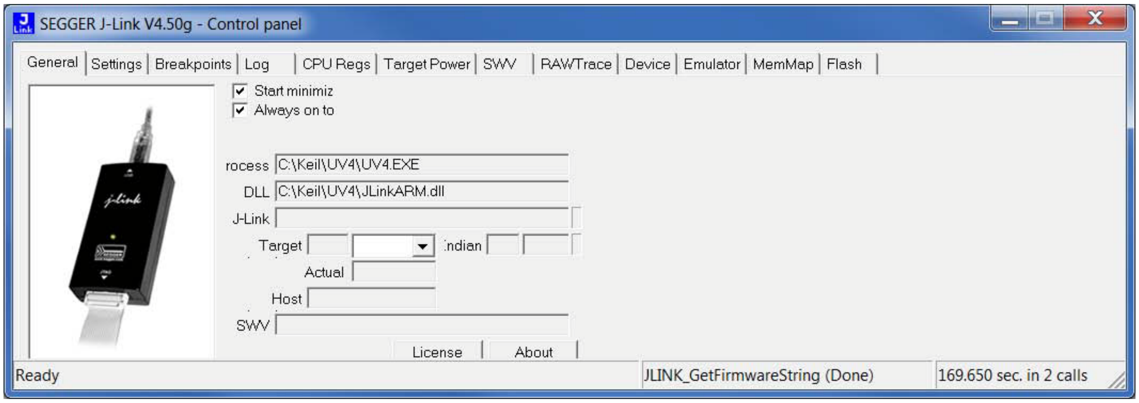
○ Use Simulator
☑ Limit Speed to Real-Time
☑ Load Application at Startup ☑ Run to main()
Initialization File:
Restore Debug Session Settings
☑ Breakpoints ☑ Toolbox
☑ Watch Windows & Performance Analyzer
☑ Memory Display
CPU DLL: SARMCM3.DLL Parameter:
Dialog DLL: DARMCM1.DLL Parameter: -dnRF5

● Use: Cortex-M/R J-LINK/J-Trace
☑ Load Application at Startup ☑ Run to main()
Initialization File: .\JLinkSettings.ini
Restore Debug Session Settings
☑ Breakpoints ☑ Toolbox
☑ Watch Windows
☑ Memory Display
Driver DLL: SARMCM3.DLL Parameter:
Dialog DLL: TARMCM1.DLL Parameter:

Figure 41 *Selecting JLink debugger in Keil*

4. Click the **Settings** button shown in ***Figure 41.*** Both the SEGGER Control Panel and the Keil Target Driver Setup will open.

Note: If the SEGGER J-Link Lite firmware requires an update, before the SEGGER Control Panel or Keil target Driver Setup open, you will be prompted with the message “A new firmware version is available for the connected emulator”. In this case, click **OK**.



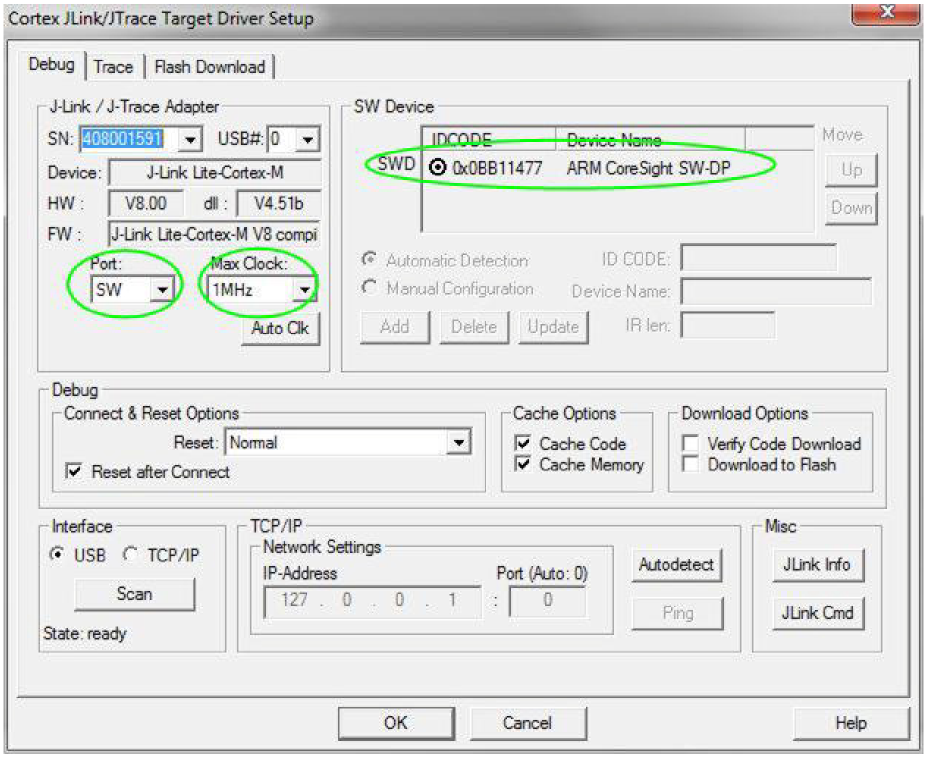
☑ Start minimiz
☑ Always on to

rocess C:\Keil\UV4\UV4.EXE
DLL C:\Keil\UV4\JLinkARM.dll

Ready | JLINK_GetFirmwareString (Done) | 169.650 sec. in 2 calls

Figure 42 SEGGER control panel

5. Click the **Debug** tab shown in the figure. Set Port to **SW** and Max Clock to **1MHz**, as shown in *Figure 43*. Make sure that SN and IDCODE are populated properly and click **OK**.



SN: 408001591 USB#: 0
Device: J-Link Lite-Cortex-M
HW: V8.00 dll: V4.51b
FW: J-Link Lite-Cortex-M V8 compi
Port: SW Max Clock: 1MHz

	IDCODE	Device Name
SWD	● 0x0BB11477	ARM CoreSight SW-DP

● Automatic Detection ID CODE:
○ Manual Configuration Device Name:
IR len:

Reset: Normal
☑ Reset after Connect

☑ Cache Code
☑ Cache Memory

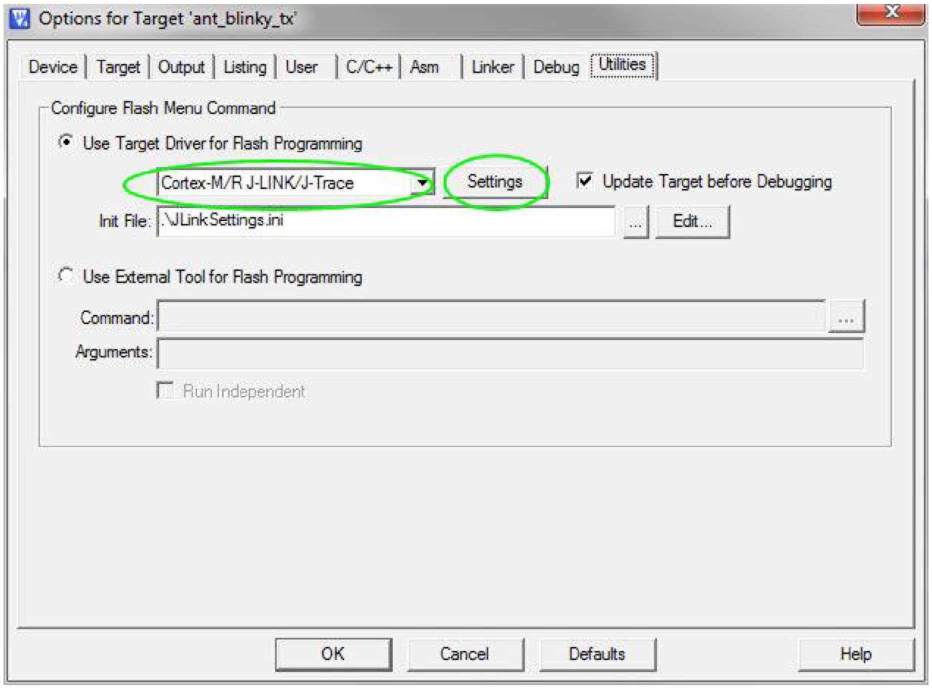
□ Verify Code Download
□ Download to Flash

● USB ○ TCP/IP
State: ready

IP-Address 127 . 0 . 0 . 1 Port (Auto: 0) : 0

Figure 43 Debug settings

6. Select the J-Link device for target programing and provide the appropriate flash algorithm.



Options for Target 'ant_blinky_tx'

Configure Flash Menu Command

● Use Target Driver for Flash Programming

Cortex-M/R J-LINK/J-Trace

☑ Update Target before Debugging

Init File: .\JLinkSettings.ini

○ Use External Tool for Flash Programming

Command:

Arguments:

☐ Run Independent

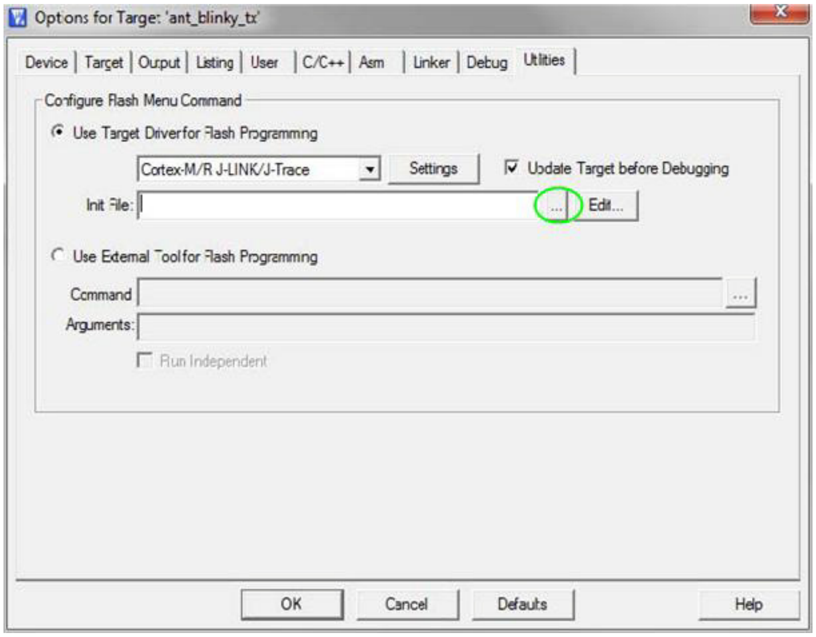
Figure 44 Flash settings

7. If the J-Link serial number appears in the SN field, the device is properly installed. The default settings can be accepted by clicking **OK**, closing both the SEGGER Control Panel and Keil target Driver Setup.

JLinkSettings.ini file

Tho improve the debug experience while debugging with a SoftDevice, change AllowSimulation = 1 to AllowSimulation = 0 in your default **JLinkSettings.ini** file under your project.

1. Enter the utilities settings from KEIL target options (**ALT**+**F7**) and click the **...** button to select the **JLinksettings.ini** file.



Options for Target: 'ant_blinky_tx'

Configure Flash Menu Command

● Use Target Driver for Flash Programming

Cortex-M/R J-LINK/J-Trace

☑ Update Target before Debugging

Init File:

○ Use External Tool for Flash Programming

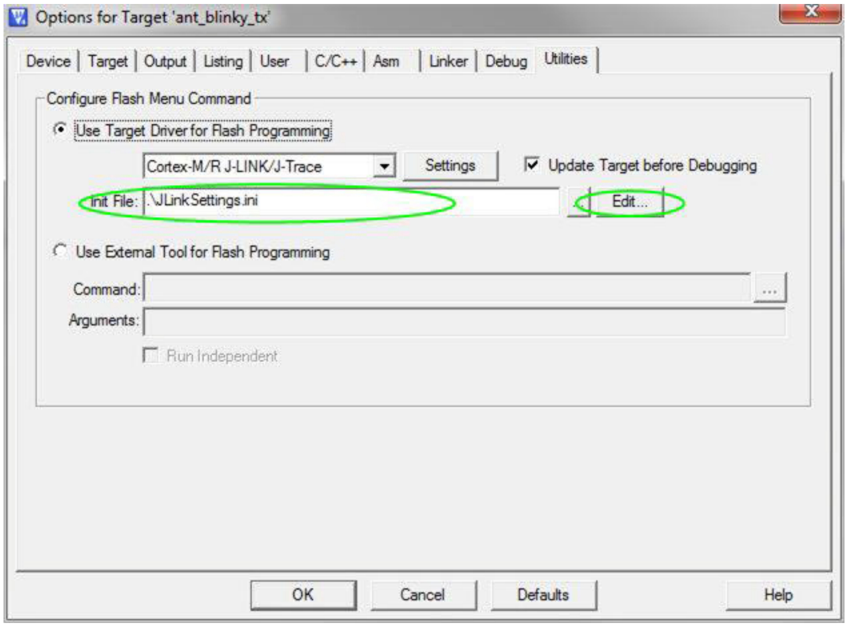
Command:

Arguments:

☐ Run Independent

Figure 45 Locating JLinkSettings.ini

2. Click **Edit** and set the JLinkSettings.ini settings as specified in JLinkSettings.ini table.



Options for Target 'ant_blinky_tx'

Configure Flash Menu Command

● Use Target Driver for Flash Programming

Cortex-M/R J-LINK/J-Trace

☑ Update Target before Debugging

Init File: .\JLinkSettings.ini

○ Use External Tool for Flash Programming

Command:

Arguments:

☐ Run Independent

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.

Life support applications

Nordic Semiconductor's 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.

Contact details

For your nearest dealer, please see http://www.nordicsemi.com.
Information regarding product updates, downloads, and technical support can be accessed through your My Page account on our homepage.

Main office: Otto Nielsens veg 12
7052 Trondheim
Norway

Phone: +47 72 89 89 00
Fax: +47 72 89 89 89

Mailing address: Nordic Semiconductor
P.O. Box 2336
7004 Trondheim
Norway

DNV
NORWEGIAN ACCREDITATION
No. S03
NS-EN ISO 9001 CERTIFICATED FIRM

Revision history

Date	Version	Description
September 2012	1.0	

ARM statement

Keil, µVision, and Cortex are trademarks of ARM Limited. All other brands or product names are the property of their respective holders.