Getting Started

Contents

nRF5 SDK v11.0.0-2.alpha

Getting Started

The SDK provides example applications that you can run on your development kit to ensure that everything is set up correctly. After these tests, you can use the examples as starting point to develop your own applications.

To quickly get started, run some precompiled examples. There are both simple precompiled examples that can be tested by observing blinking LEDs on the board and more complex examples that can communicate with your smartphone using *Bluetooth* low energy. These precompiled examples do not require you to set up the full toolchain. Therefore, programming and testing them should not take more than a few minutes.

If everything works as expected, set up your development kit and toolchain. For nRF52 Development Kits, see Setting up the development kit. For nRF51 Development Kits, follow the steps in section 3 (Getting started) in the nRF51 Development Kit User Guide. Then install the SDK.

To test the development kit setup, we recommend that you use the Blinky Example. The steps that are described in Compiling and running a first example will walk you through testing your setup with the Blinky example.

After testing your setup, you should program a SoftDevice and run an example that uses the SoftDevice.

Running precompiled examples

Installing the SDK

Compiling and running a first example

Running examples that use a SoftDevice

Using the SDK with other boards

This document was last updated on Fri Dec 18 2015.
Please send us your feedback about the documentation! For technical questions, visit the Nordic Developer Zone.

1. nRF5 SDK v11.0.0-2.alpha

Running precompiled examples

To quickly test an example without having to install the full toolchain, program a precompiled example application on your development board.

The zip version of the SDK provides precompiled HEX files for most of the supplied examples. For examples that require a SoftDevice, the SoftDevice is included in the HEX file.

Note that the SDK does not include precompiled ANT+ examples. To run ANT+ examples, you must therefore set up your toolchain. Follow the instructions in Running examples that use a SoftDevice and Running the example to program the SoftDevice and example.

Perform the following steps to program a precompiled example:

1. Download the latest repository file nRF5_SDK_x.x.x_xxxxxxx.zip (for example, nRF5_SDK_11.0.0-2.alpha_1a2b3c4.zip) from developer.nordicsemi.com.
2. Extract the zip file to the directory that you want to use to work with the SDK.
3. Connect the board to your computer with a USB cable. The computer will recognize it as a standard USB drive.
4. In the SDK directory, navigate to the example that you want to test and open the `hex` subdirectory. For example, for the Blinky Example, navigate to `examples\peripheral\blinky\hex`.
5. Select the HEX file that corresponds to your development board and copy it to the board USB drive.

To test the example, follow the testing instructions in the Examples documentation. Depending on the example that you programmed, it might be sufficient to observe the LEDs on the board. Other examples require you to connect to the board via UART, Master Control Panel, or a phone app.

This document was last updated on Fri Dec 18 2015.
Please send us your feedback about the documentation! For technical questions, visit the Nordic Developer Zone.

Installing the SDK

Before you install the SDK, make sure that you have all required tools installed. For nRF52 Development Kits, see Setting up the development kit. For nRF51 Development Kits, follow the steps in section 3 (Getting started) in the nRF51 Development Kit User Guide.

The SDK is delivered both as CMSIS Packs and as a plain zip archive. You can choose to work with the CMSIS Packs if you use Keil 5.12 or later. Otherwise, work with the zip version.

Working with the zip version

If you decide to work with the zip version, complete the following steps to set up your environment:

1. Download the repository file nRF5_SDK_x.x.x_xxxxxxx.zip (for example, nRF5_SDK_11.0.0-2.alpha_1a2b3c4.zip) from developer.nordicsemi.com.
2. Extract the zip file to the directory that you want to use to work with the SDK.
3. Install the nRF5 MDK:
   - If you use Keil 5, open an example project. Keil will then prompt you to install the nRF_DeviceFamilyPack. If Keil does not prompt you, open the Pack Installer, click the **Check For Updates** button, and install the latest nRF_DeviceFamilyPack.
   - If you use Keil 4, double-click the nRF5x_MDK_x_x_x_Keil4.msi file in the extracted directory to install the MDK.
   - If you use IAR, double-click the nRF5x_MDK_x_x_x_IAR.msi file in the extracted directory to install the MDK.
   - If you use GCC, the MDK is delivered with the SDK and does not need to be installed separately.

Installing CMSIS Packs

To install all CMSIS Packs for the SDK for the first time, complete the following steps:

1. Download the packs file nRF5_SDK_x.x.x_xxxxxxx_packs.zip (for example, nRF5_SDK_11.0.0-2.alpha_1a2b3c4_packs.zip) from developer.nordicsemi.com.
2. Extract the .zip file to a temporary directory, for example `C:\tmp\nRF5_SDK`.
3. Run the install.bat file that is located in the extracted directory and follow the instructions.

After the installation, you can access the components and examples from the Keil Pack Installer. Alternatively, you can install the packs directly from the Pack Installer.

To update to newer versions of the SDK, click the **Check For Updates** button in the Pack Installer and select which packs you would like to update.

This document was last updated on Fri Dec 18 2015.
Please send us your feedback about the documentation! For technical questions, visit the Nordic Developer Zone.

Compiling and running a first example

To start developing your own application, you should first test that you set up your toolchain correctly. To do so, compile, program, and run a very simple example that does not use a SoftDevice, for example the Blinky Example.

Make sure that you have all required tools installed and that the board is connected to your computer. For nRF52 Development Kits, see Setting up the development kit. For nRF51 Development Kits, follow the steps in section 3 (Getting started) in the nRF51 Development Kit User Guide.

Erasing the board

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

If no SoftDevice has been installed on the board, simply open Keil and select **Flash > Erase**.

If a SoftDevice has been installed on the board or if you are not sure, use nRFgo Studio to make sure that you erase the whole board. Select your device in the Device Manager pane. Then click the **Erase All** button. If you prefer to use a command line tool instead, use nrfjprog (which is installed with nRFgo Studio).

Running the example

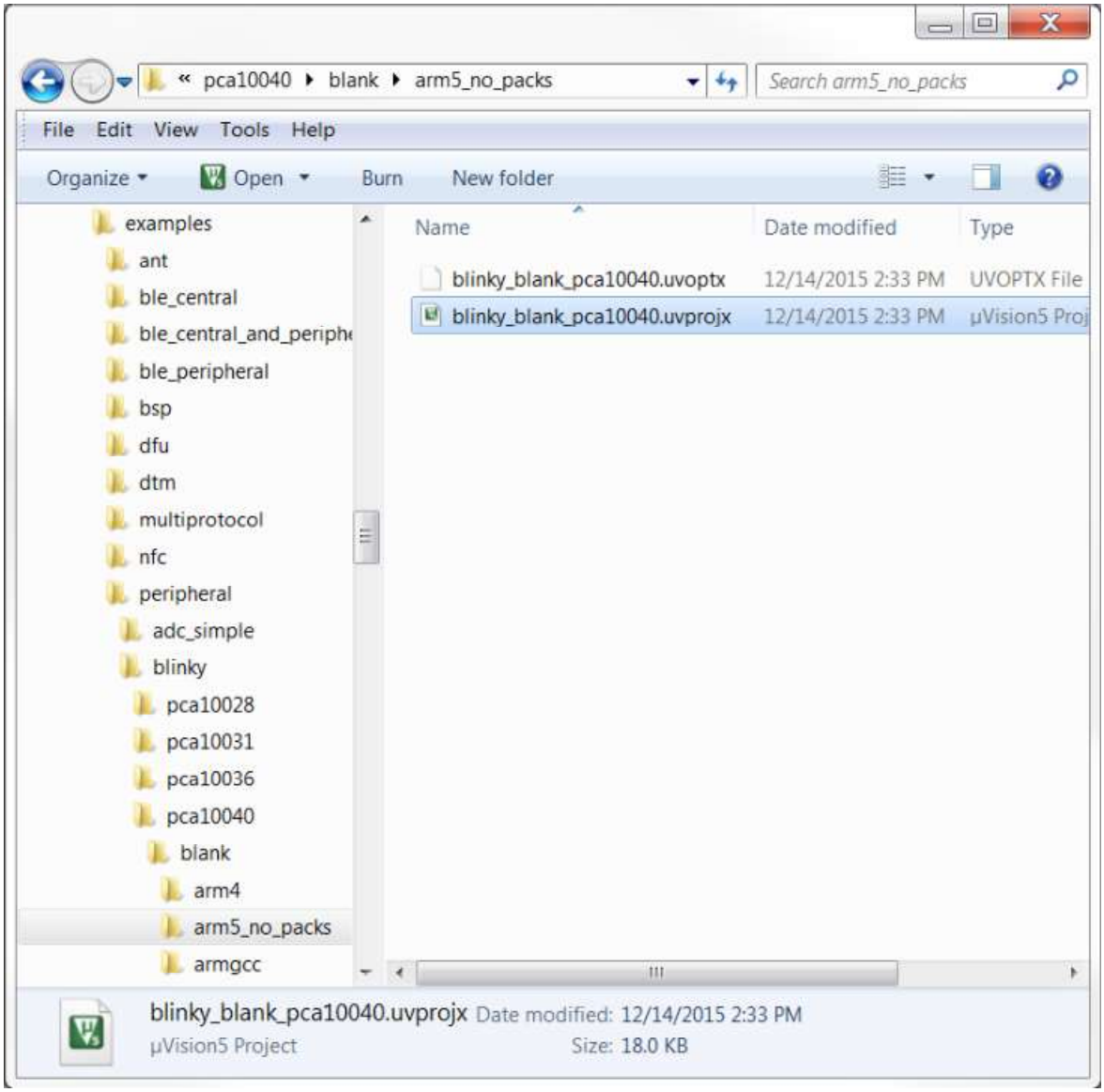
The steps for compiling the example differ depending on if you use the zip archive of the SDK or CMSIS Packs:

- Zip archive
- CMSIS Packs

Zip archive

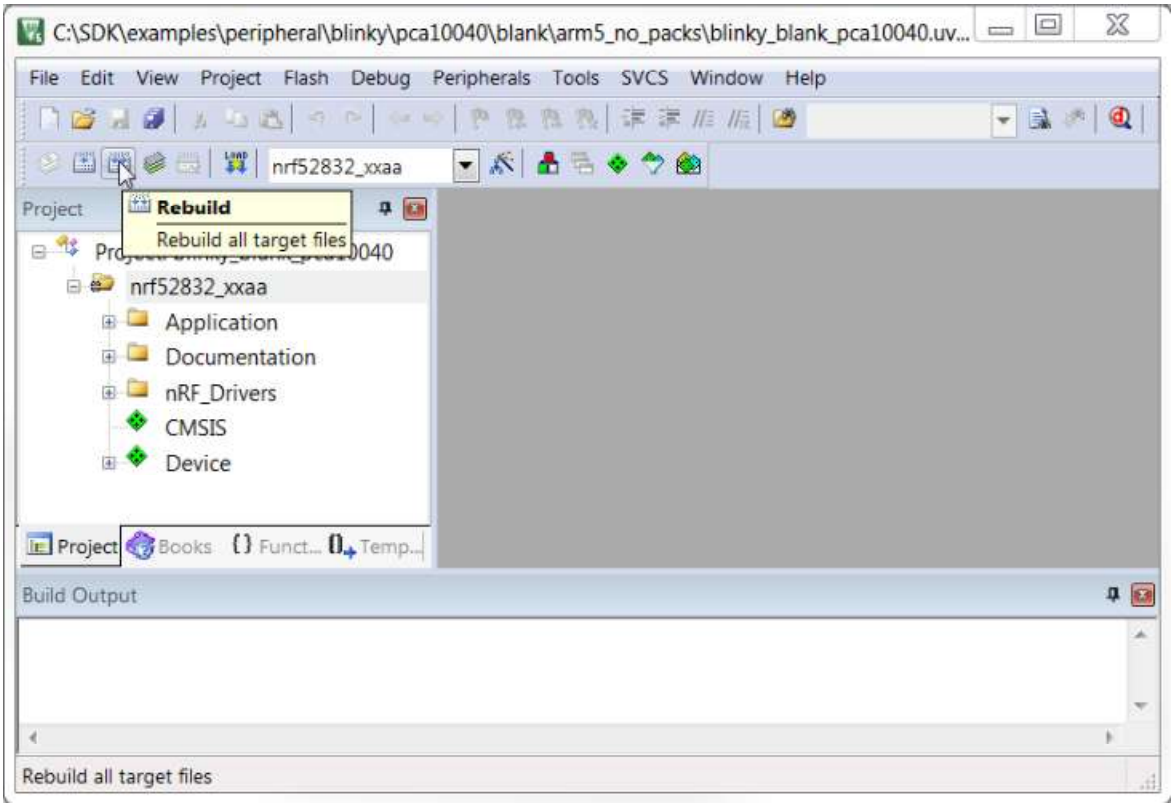
If you use the zip archive of the SDK, perform the following steps to run the example:

1. Navigate to the folder that contains the example project:
   a. Open the folder that contains the SDK files, thus the folder where you extracted the SDK zip file.
   b. Select the example that you want to run, for example, `...\examples\peripheral\blinky`.
   c. Select the board that you want to use, for example, `...\pca10040`.
   d. If you want to run the example with a SoftDevice, select the SoftDevice. Otherwise, select **blank**. So for running the Blinky Example without a SoftDevice, select `...\blank`.
   e. Select the toolchain that you want to use. The following steps assume that you are using Keil 5 without packs, so select `...\arm5_no_packs`.



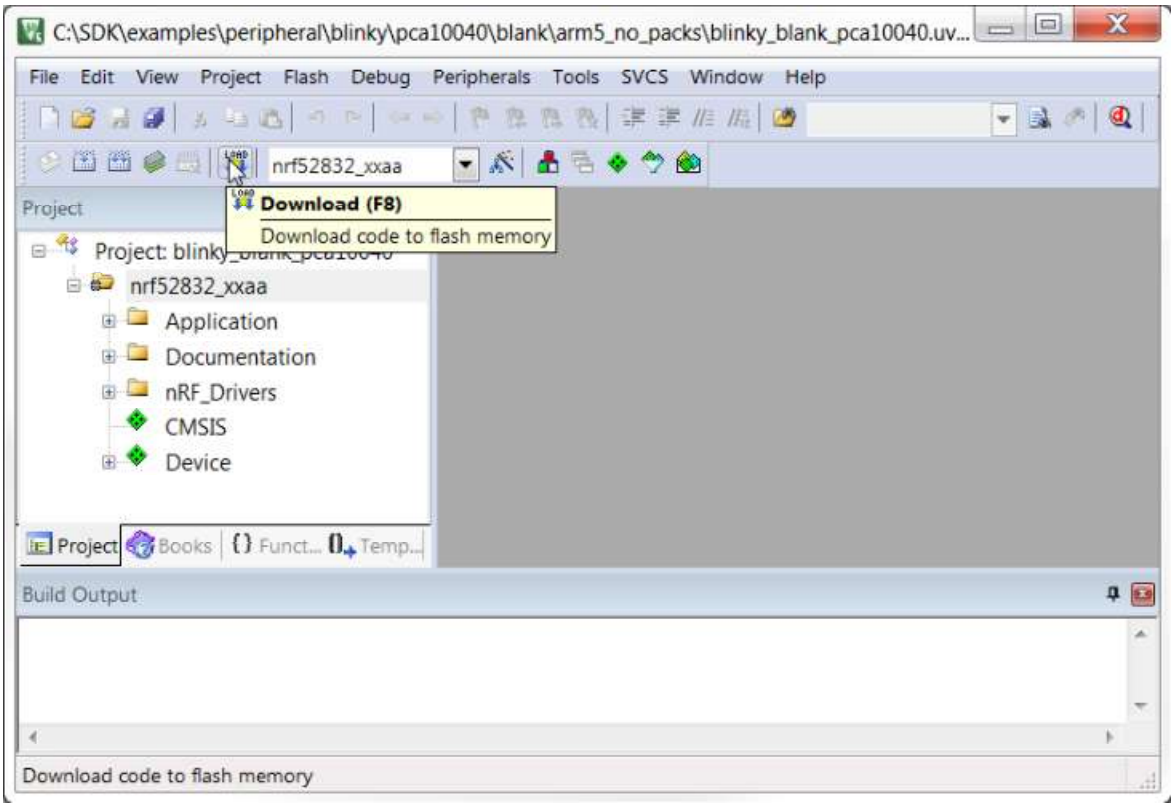
Navigate to the example project

2. Double-click the uvprojx file to open the project in Keil.
3. Build the project to compile all files.



Rebuild all target files

4. Download the code to flash memory to flash the project to the board. If Keil prompts you to update to the latest firmware version, select Yes.



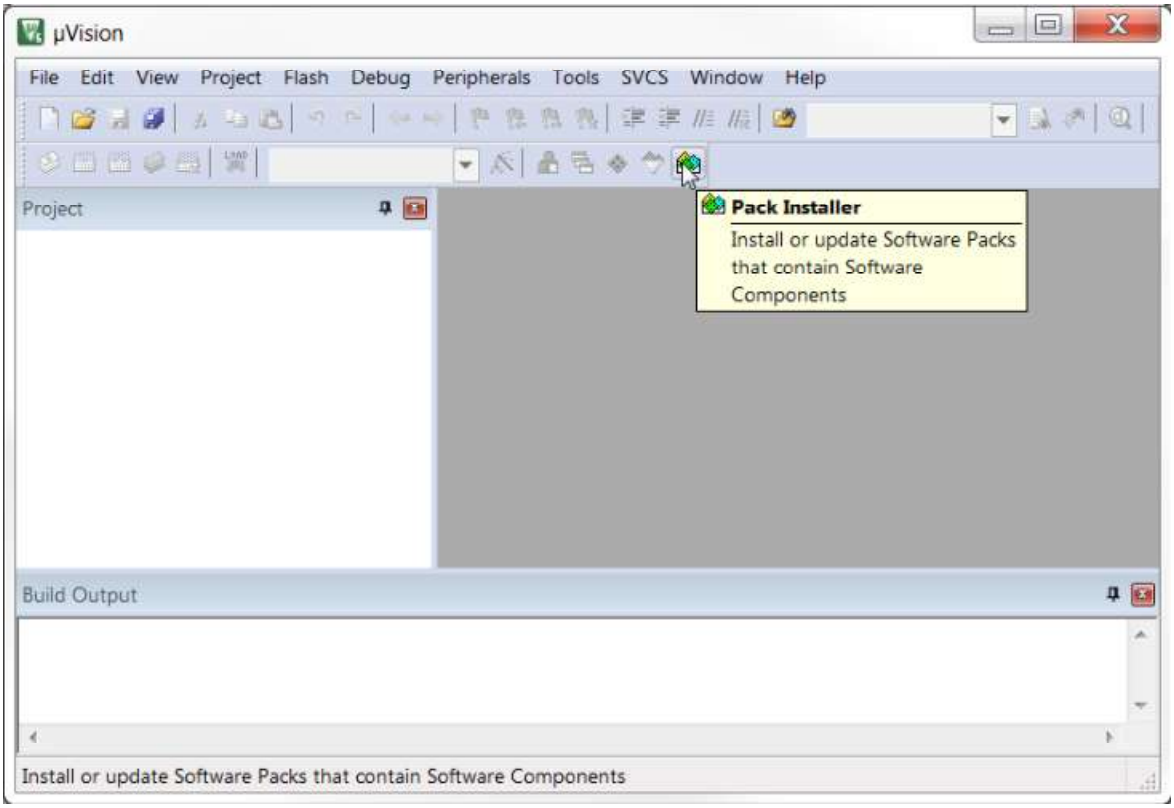
Download code to flash memory

5. Test the example as described in the Testing section of the example documentation (Testing Blinky).

CMSIS Packs

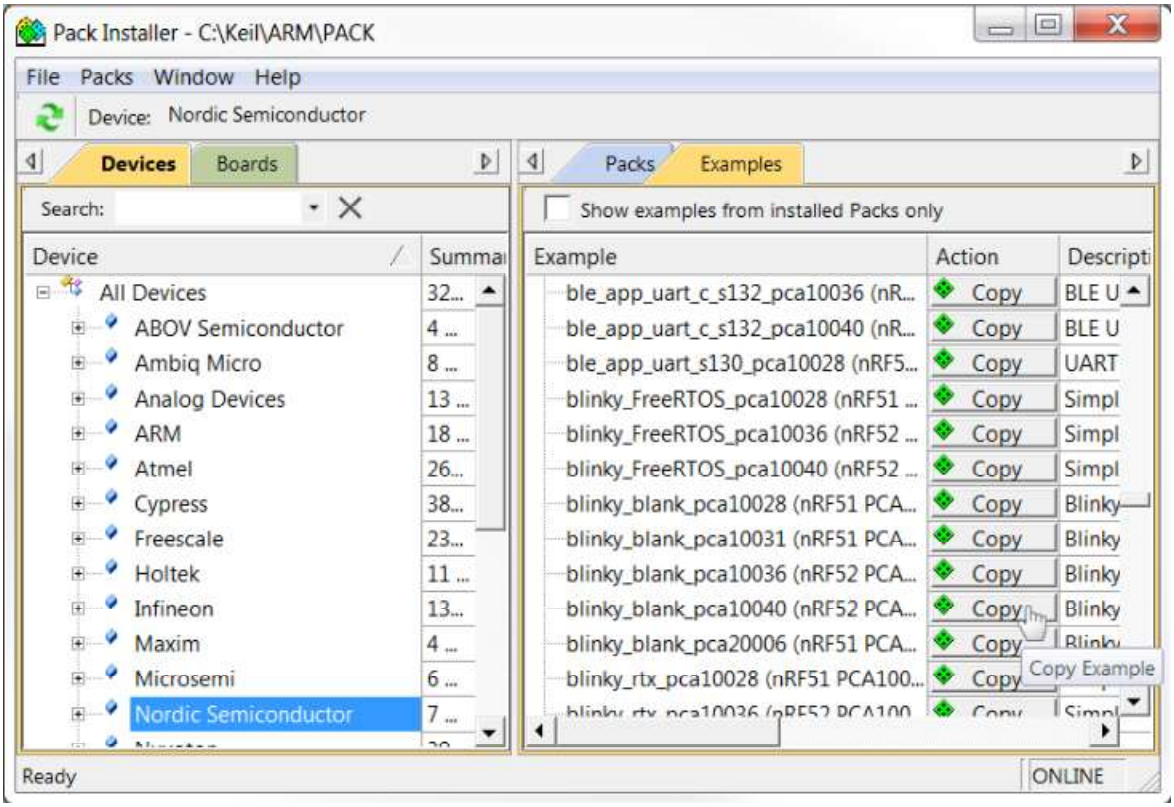
If you use CMSIS Packs, perform the following steps to run the example:

1. In Keil, open the Pack Installer.



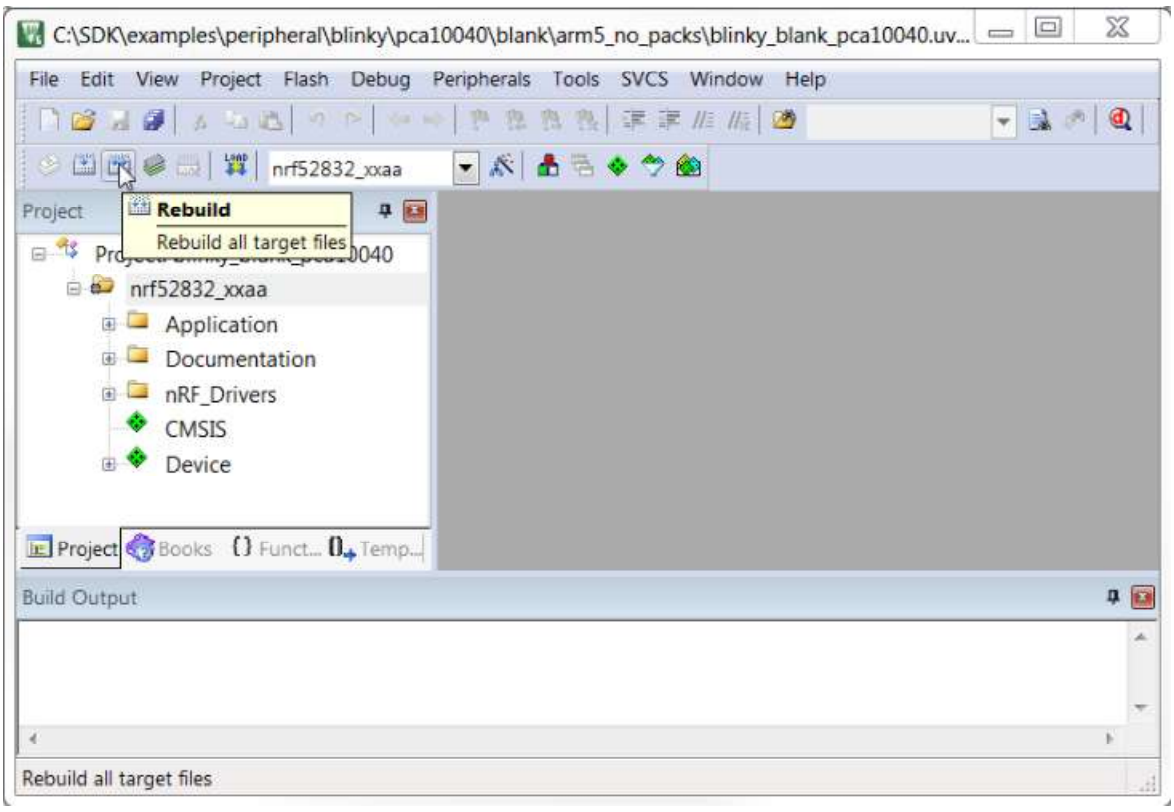
Open the Pack Installer

2. Select the Devices tab and the Examples tab.
3. In the Device pane, select Nordic Semiconductor to filter the examples that are displayed in the Example pane and display only Nordic Semiconductor examples. You can also filter by boards by selecting the desired board on the Boards tab.
4. Locate the example that you want to run, for example **blinky_blank_pca10040**.
5. Click the Copy button next to the example.



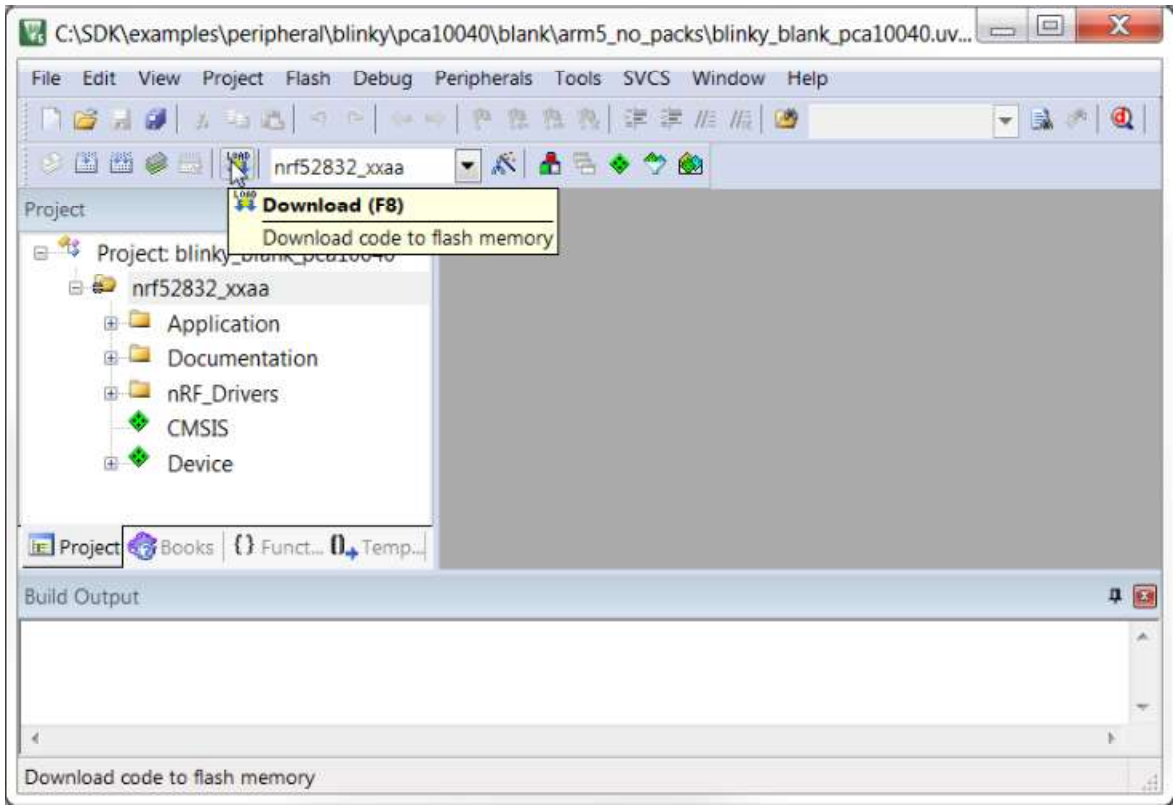
Copy the example

6. In the window that is displayed, enter the folder where you want to store the example code and select both options.
7. Click OK. The project is copied to the directory that you specified and opened in Keil.
8. Build the project to compile all files.



Rebuild all target files

9. Download the code to flash memory to flash the project to the board. If Keil prompts you to update to the latest firmware version, select Yes.



Download code to flash memory

10. Test the example as described in the Testing section of the example documentation (Testing Blinky).

This document was last updated on Fri Dec 18 2015.
Please send us your feedback about the documentation! For technical questions, visit the Nordic Developer Zone.

Running examples that use a SoftDevice

Before you can run more advanced examples that use *Bluetooth* or ANT, you must program the SoftDevice on the board.

After you programmed the SoftDevice, follow the steps described in Running the example to run an example that uses the SoftDevice. For example, test the Heart Rate Application, Broadcast, or BLE Heart Rate Collector Example examples.

Programming SoftDevices

The SoftDevice binary is located in folder `components\softdevice\SoftDevice\hex` in the SDK, where *SoftDevice* is the name of the SoftDevice. You can also download SoftDevices from nordicsemi.com. Note that ANT SoftDevices for the nRF52 Series are not distributed by Nordic Semiconductor. See the Running ANT examples on the nRF52 Series section for information about how to run examples that use these SoftDevices.

There are several methods to program the SoftDevice:

- Using nRFgo Studio
- From an example project within ARM Keil
- Using the GCC makefile of an example

nRFgo Studio

To program the SoftDevice using nRFgo Studio, perform the following steps:

1. Open nRFgo Studio.
2. In the Device Manager, select the nRF5 Development board that you are working with (identified by the SEGGER serial number).
3. Select the **Program SoftDevice** tab.
4. Click **Browse** and navigate to the SoftDevice file that you want to use.
5. Click **Program**.

nRFgo Studio will erase any existing SoftDevice and program the selected SoftDevice.

ARM Keil

To program the SoftDevice using an example project in Keil, perform the following steps:

1. Open an example project in Keil. The example must require a SoftDevice.
2. Instead of the default target, select the target to flash the SoftDevice, for example, **flash_s132_nrf52_2.0.0-7.alpha_softdevice**.
3. Click **Options for Target**.
4. Select the Debug pane and click the **Settings** button for the J-Link / J-TRACE Cortex.
5. Select the J-Link / J-Trace Adapter corresponding to the serial number that is printed on your device.
6. Click **OK** to close the dialogs.
7. In the main window, click **Download** to program the SoftDevice.

Keil will erase any existing SoftDevice and program the appropriate SoftDevice for the example.

GCC makefile

To program the SoftDevice using a GCC makefile, perform the following steps:

1. Open a command prompt in the folder that contains the makefile of an example. The example must require a SoftDevice.
2. Ensure that nrfjprog.exe is in the path, thus the path to nrfjprog.exe is part of the PATH environment variable. The makefile issues a call to the nrfjprog tool.
3. Run the following command: **make flash_softdevice**

Running the makefile will erase any existing SoftDevice and program the appropriate SoftDevice for the example.

Running ANT examples on the nRF52 Series

ANT SoftDevices for the nRF52 Series are not distributed by Nordic Semiconductor. You can download them from thisisant.com.

There are no Keil or GCC targets to program these SoftDevices. You must therefore use nRFgo Studio to program them.

The SDK does not include the header files for the ANT SoftDevices for the nRF52 Series. Therefore you must add them to your project before you can compile your application. To do so, extract the downloaded zip file that contains the SoftDevice. If you are working with the zip version of the SDK, copy the SoftDevice headers to `components/softdevice/SoftDevice/headers`. If you are using Keil packs, copy the files into a `headers` subfolder in your example folder.

Make sure that ANT_LICENSE_KEY is uncommented in the `nrf_sdm.h` file that you copied. The included license key can be used for evaluation, but before releasing a product, it must be replaced with a valid commercial license key. For more information about licensing, see thisisant.com.

The nRF5 SDK supports the current Nordic Semiconductor development board. You can also enable support for another board by selecting a specific supported Nordic Semiconductor board or by defining a custom board. In this way, you can run the SDK examples on any target board.

Supported boards

By default, the SDK project files must be used with specific Nordic Semiconductor boards, usually the most current board. However, the SDK provides support for other boards as well.

The following boards are supported:

Board	#define
nRF6310 (part of nRFgo Starter Kit)	BOARD_NRF6310
PCA10000 (nRF51822 USB dongle)	BOARD_PCA10000
PCA10001 (part of nRF51822 Evaluation Kit)	BOARD_PCA10001
PCA10002 (nRF51422 USB dongle)	BOARD_PCA10002
PCA10003 (part of nRF51422 Evaluation Kit, BLE + ANT)	BOARD_PCA10003
PCA10028 (part of nRF51422 Evaluation Kit, Arduino form factor)	BOARD_PCA10028
PCA10031 (nRF51422 USB dongle)	BOARD_PCA10031
PCA10036 (part of nRF52 Preview Development Kit)	BOARD_PCA10036
PCA10040 (part of nRF52 Development Kit)	BOARD_PCA10040
PCA20006 (nRF51822 Beacon board)	BOARD_PCA20006
WT51822 (Wavetek shield)	BOARD_WT51822
N5DK1 (Dynastream N5 Starter Kit)	BOARD_N5DK1
Custom board (definition in custom_board.h)	BOARD_CUSTOM

Enabling support for a board

To enable support for an older Nordic Semiconductor board or a custom board, you must include a define statement for the board that you want to use before you compile the code.

According to the define statement, the suitable board support file is selected. The board support file defines the peripherals, thus the location of LEDs and buttons, for the selected platform. The header files for supported boards are located in the directory `examples\bsp`. The actual selection of the file according to the define statement is done in boards.h.

Depending on the device on the legacy board, you might need to change the memory layout. For example, all nRF51 examples assume that you are using the 32 kB variant of nRF51, so if you are using a variant with 16 kB RAM, you must decrease the size of IRAM1 by 16 kB (0x4000 in hex). In Keil, click **Project > Options for Target '...'** and modify the values for "Read/Write Memory Area". For GCC, change the linked *.ld file in the Makefile.

Adding support for a custom board

To add support for a custom board, you must create a custom board support file with the name custom_board.h. This file must be located in a directory in the Include path. You can then select to use the custom board by adding the define statement `#define BOARD_CUSTOM`.

The easiest way to create the custom_board.h file is to start with an existing platform definition file and adapt it to your needs.

A platform example definition looks as follows:

```
// Defines the number of LEDs. In this example, there is a single RGB LED.
#define LEDS_NUMBER    3

// Defines which PIOs control the color of the LED.
#define LED_START      21
#define LED_RGB_RED    21
#define LED_RGB_GREEN  22
#define LED_RGB_BLUE   23
#define LED_STOP       23

// Defines an RGB LED as 3 single LEDs.
#define BSP_LED_0 LED_RGB_RED
#define BSP_LED_1 LED_RGB_GREEN
#define BSP_LED_2 LED_RGB_BLUE
```

```
#define LED_RGB_RED_MASK    (1<<LED_RGB_RED)
#define LED_RGB_GREEN_MASK  (1<<LED_RGB_GREEN)
#define LED_RGB_BLUE_MASK   (1<<LED_RGB_BLUE)

#define BSP_LED_0_MASK    (1<<BSP_LED_0)
#define BSP_LED_1_MASK    (1<<BSP_LED_1)
#define BSP_LED_2_MASK    (1<<BSP_LED_2)

#define LEDS_MASK      (BSP_LED_0_MASK | BSP_LED_1_MASK | BSP_LED_2_MASK)

// Defines which LEDs are lit when the signal is low. In this example,
// all LEDs are lit.
#define LEDS_INV_MASK  LEDS_MASK

// Defines the user buttons. In this example, there are no user buttons.
#define BUTTONS_NUMBER 0
#define BUTTONS_MASK   0x00000000

// Defines the UART connection with J-Link.
#define RX_PIN_NUMBER  11
#define TX_PIN_NUMBER  9
#define CTS_PIN_NUMBER 10
#define RTS_PIN_NUMBER 8
#define HWFC           true
```

This document was last updated on Fri Dec 18 2015.
Please send us your feedback about the documentation! For technical questions, visit the Nordic Developer Zone.