N5 Starter Kit User Manual

N5DK1

D00001617 Rev 3.0

T +1 403.932.9292 F +1 403.932.6521 dynastream.com thisisant.com
#201, 100 Grande Blvd., Cochrane, Alberta, Canada T4C 0S4

Copyright Information and Usage Notice

The Dynastream Innovations Inc. ANT Products described by the information in this document are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Dynastream product could create a situation where personal injury or death may occur. If you use the Products for such unintended and unauthorized applications, you do so at your own risk and you shall indemnify and hold Dynastream and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Dynastream was negligent regarding the design or manufacture of the Product.

Restricted Use of Development Kits

Development Kits are intended for use solely by design engineers for the purpose of evaluating the feasibility of ultra low-power wireless data communications applications. The user's evaluation must be limited to use of an assembled Development Kit within a laboratory setting which provides for adequate shielding of RF emission which might be caused by operation of the Development Kit following assembly. The assembled Development Kit must not be operated in a residential area or any area where radio devices might be subject to harmful electrical interference. Hardware contained in the Development Kit may not be certified for use by the FCC in accordance with Part 15, or to other known standards of operation governing radio emissions. Distribution and sale of this Development Kit is intended solely for use in future development of devices which may be subject to FCC regulation, or other authorities governing radio emission. This Development Kit may not be resold by users for any purpose. Operation of the Development Kit in the development of future devices is deemed within the discretion of the user and the user shall have all responsibility for any compliance with any FCC regulation or other authority governing radio emission of such development or use. All products developed by the user must be approved by the FCC or other authority governing radio emission prior to marketing or sale of such products and user bears all responsibility for obtaining the authority's prior approval, or approval as needed from any other authority governing radio emission. If user has obtained the Development Kit for any purpose not identified above, user should return the Development Kit to Dynastream Innovations Inc. immediately. The Development Kit is an experimental device, and Dynastream makes no representation with respect to the adequacy of the Development Kit in developing ultra low-power wireless data communications applications or systems. The Development Kit and products based on the technology in the Development Kit operate on shared radio channels. Any Products using ANT technology must be designed so that a loss of communications due to radio interference or otherwise will not endanger either people or property, and will not cause the loss of valuable data. Dynastream assumes no liability for the performance of products which are designed or created using the Development Kit.

Reference Design Disclaimer

The references designs and codes provided with the development kit may be used with ANT devices only and remain the copyrighted property of Dynastream Innovations Inc. The reference designs and codes are being provided on an "as-is" basis and as an accommodation, and therefore all warranties, representations, or guarantees of any kind (whether express, implied or statutory) including, without limitation, warranties of merchantability, non-infringement, or fitness for a particular purpose, are specifically disclaimed.

About the User Manual

- This user manual is to facilitate the evaluation and prototyping of solutions based on the N5 Starter Kit
- This user manual is for design engineers who are using the N5 Starter Kit to evaluate ANT as a wireless sensor network solution and develop applications based on ANT. The development kit is not intended as an end product or for use by individuals who do not have a professional background in data communications. Refer to the Copyright Information and Usage Notice page for detailed information and usage restrictions.

Revision History

Revision	Effective Date	Description
1.0	March 2014	First Release
1.1	March 28/2014	Updated Links
1.2	June 2014	Added Restricted Use of Development Kits and Reference Design Disclaimer notices.
1.3	July 2014	Removed EEPROM board description. Updated USB interface board driver installation. Updated N5 Starter Kit SDK description.
2.0	November 2014	Added documents to related documents section Added section on loading SoftDevice onto Module. Added section on basic Keil usage for building reference code. Added section on terminology to describe sets of development kit components. Added usage of IO Tx demo together with Rx demo. Added description and usage of the following demos: • ANT Background Scanning Demo • ANT Relay Demo • ANT Auto Shared Channel Demo • ANT DFU/Bootloader Demo • ANT Debug
3.0	July 2015	Updated the installation process Added description and usage of the following demos: • ANT Scalable Channel Demo • ANT Scalable Encrypted Demo • ANT Scan and Forward Demo • ANT Asynchronous Controller using Continuous Scanning Mode Added section on converting Nordic examples to run on the N5 Starter Kit Added section on updating from N5 Starter Kit SDK version 2 to version 3

Table of Contents

List of Figures

List of Tables

1 Overview

The N5 Starter Kit is the development kit for the N5 ANT SoC module series, a turnkey, ultra low power wireless module based on the Nordic Semiconductor nRF51422 System on Chip solution.

This development kit includes a comprehensive set of hardware components and provides access to software tools to allow users to evaluate, design and prototype using the ANT Wireless Protocol. Table 1-1 below lists the hardware components included in the N5 Starter Kit.

Table 1-1. N5 Starter Kit Contents

Quantity	Part
2	N5 Modules
1	I/O Board
1	Battery Board
1	USB Interface Board
1	Segger J-link Programmer

1.1 N5 Modules

The N5 module series uses Nordic Semiconductor's nRF51422, the industry's first System on Chip (SoC) solution that supports both ANT and Bluetooth low energy depending on the loaded protocol stack with advantages in power, size, cost, speed and security built directly into the protocol layer. The nRF51422 integrates a 32-bit ARM® Cortex™ M0 CPU with 256KB flash, 16KB RAM, and analog and digital peripherals. Benefits will reach consumers in the form of new products with even longer battery life, expanded functionality and security, further simplified operation and overall value.

The N5 modules included in the N5 Starter Kit are the N548M5CB, which is a N548M8CB soldered onto 20mm x 20mm carrier boards. The N548M5CB includes an antenna, onboard 32k and 16M crystal clock, DC-DC converter and 13 GPIOs with 6 analogue inputs.

Figure 1-1. N548M5CB

When ordered separately, the N548M5CB modules are pre-programmed with the S210 SoftDevice and reference ANT network processor application code in order to function as a generic ANT RF module when connected to an application controller. Both pre-programmed images can be easily replaced via the onboard SWD interface pins using off-the-shelf ARM programmers.

1.2 Battery Board

Table 1-2 describes each of the numbered components shown on the battery board in Figure 1-2.

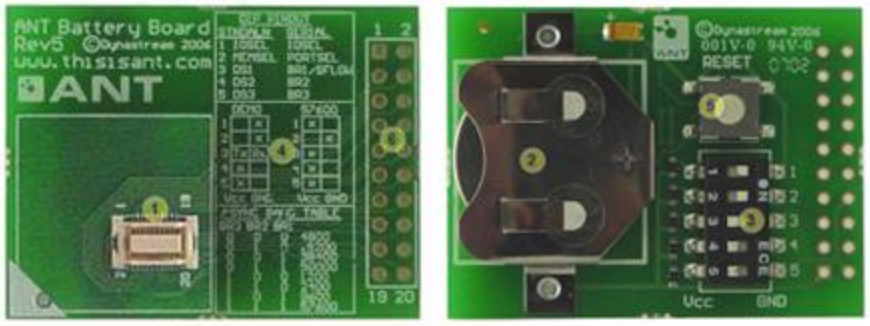


Figure 1-2. Battery Board

Table 1-2. Battery Board Description

Number	Component	Description
1	Module Socket	Used for plugging in an ANT module or an I/O interface board
2	Battery Slot	Used to power the demo mode setup (fits a 2032 coin cell)
3	Dipswitches	Used to select the state of the module's 5 configuration lines.
4	Dipswitch Instructions	Silkscreen instructions showing Dipswitch pin-out, Default Baud Rate configuration, and Baud Rate table
5	Reset Button	Resets the module
6	Interface Header	0.1" module interface header. See Table 1-4 below for the pin-out of this 20-pin header (not provided on board).

1.3 I/O Interface Board

Table 1-3 describes each of the numbered components shown on the battery board in Figure 1-3.

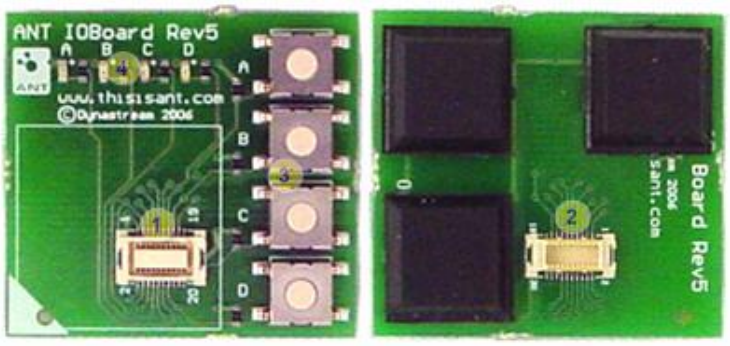


Figure 1-3. I/O Interface Board

Table 1-3. I/O Interface Board Description

Number	Component	Description
1	Module Socket	Used for plugging in an ANT module
2	Connector	Used to plug onto the battery board
3	Buttons	Used as inputs to 4 of the module's IO pins. See Table 1-4 below. When a button is released its IO line is pulled up with 1MΩ resistor. When a button is pressed its IO line is grounded.
4	LEDs	Used as outputs of 4 of the module's IO pins. See Table 1-4 below. An LED turns ON when its line is low and OFF when its line is high.

Table 1-4. N5 Starter Kit Boards Stack Pin-outs

Molex Pin #	nRF51 Pins	Battery Board Interface Header Pin #	Battery Board Input	IO Board Components	USB Interface Board Header
1	Vcc	1			1
2	P005	5	Switch 1		5
3	P012	3,11		Button C	3
4	P015	4,15		LED C	4
5	P006	18	Switch3		
6	SWDCLK	7			7
7	P024	19	Switch 4		
8	P003	14		LED B	
9	P009	20	Switch 5		
10	RST/SWDIO	6	Reset Button		6
11	P000	17	Switch 2		
12	P008	13		LED A	
13	P030	16		LED D	
14	P011	12		Button D	
15	P002	10		Button B	10
16	P021	N/C			
17	P023	9		Button A	9
18	P001	N/C			
19	GND	2,8			2,8
20	P004	N/C			

1.4 USB Interface Board

Figure 1-4. USB Interface Board

Table 1-5 describes each of the numbered components shown on the battery board in Figure 1-4.

Table 1-5. USB Interface Board Description

Number	Component	Description
1	Module Socket	Used for plugging in an ANT module
2	Interface Header	0.1" module interface header. See Table 1-4 for the pin-out of this 10-pin header (not provided on board).

1.5 Segger J-Link Lite Programmer

The Segger J-Link Lite Cortex M (cable and board) provided is used to connect the ANT module to a PC for programming as shown in Figure 1-5.

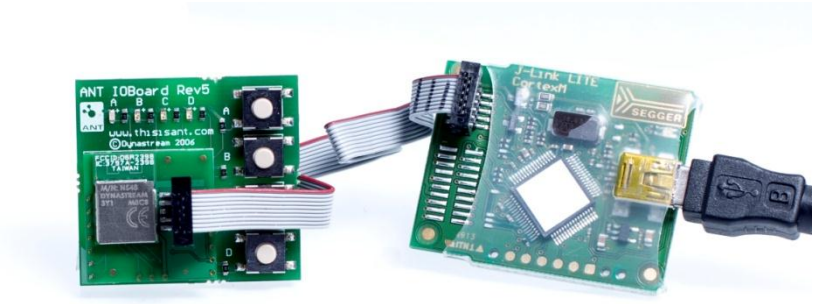


Figure 1-5. Connecting the Segger Cable

NOTE: More advanced J-Link programmers are available from Segger. Refer to section 8 for details.

1.6 Technical Resources

1.6.1 Documentation

To learn more about ANT, the following documents are available on www.thisisant.com. To access some of these documents it may be necessary to create an account.

Table 1-6. Related Documents

Document	Description
N5 Module Datasheet	The technical specification of the N5 Module Series.
ANT Message Protocol and Usage Document	Describes the ANT protocol and software interface.
ANT AN RSSI Extended Information	Describes how to enable and decode RSSI information on ANT devices.
ANT AN ANT Channel Search and Background Scanning Channel	Describes the operation and usage of a background scanning channel.
ANT AN Over the Air Firmware Updates using ANT-FS	Describes a transport mechanism based on ANT-FS to transfer software images to an embedded device.
ANT AN Auto Shared Channel Master Example	Describes a mechanism for assigning shared addresses to nodes connecting to a shared channel.
ANT-FS Technical Specification	Describes the ANT-FS (ANT File Share) mechanism for transferring files wirelessly between devices.
ANTware II User Manual	Describes of the usage of the PC utility ANTware II.
ObservANT User Manual	Describes the usage of the PC utility ObservANT.

1.6.2 Support

Technical support for the ANT wireless protocol is available via Tech FAQs and the ANT Developer Forum:

http://www.thisisant.com/developer/resources/tech-faq/

http://www.thisisant.com/forum/

Technical support for the hardware implementation (including radio performance) of ANT chips is provided by the relevant semiconductor supplier and their regional distributors. For nRF51422-specific help, please contact Nordic Semiconductor.

2 Software Setup Procedure

The following ANT, Nordic and third-party software components must be installed to begin development with the N5 Starter Kit:

Table 2-1. ANT Software Components

ANT Components		
	Component	**Source**
1.	ANT USB Interface Board Driver	www.thisisant.com/developer/resources/downloads (ANT USB Interface Board Driver (Windows))
	Description: Windows drivers for the ANT USB Interface Board.	
2.	ANTWareII	www.thisisant.com/developer/resources/downloads (ANTWareII)
	Description: PC Utility tool used to evaluate and debug ANT designs and applications (using the ANT USB Interface Board). Requires .NET Framework 3.5.	
3	OTA Updater	www.thisisant.com/developer/resources/downloads (OTA Updater)
	Description: PC tool used to perform over-the-air application, bootloader and SoftDevice image updates on an N5 module running the ANT Bootloader/DFU example code. Requires .NET Framework 4.0.	
4	ObservANT	www.thisisant.com/developer/resources/downloads (ObservANT)
	Description: PC tool used to display debug information transmitted over an ANT channel. Requires .NET Framework 4.0.	

Table 2-2. Third-Party Software Components

Third-Party Components		
	Component	**Source**
1.	Keil MDK ARM Development Kit v5 (Evaluation License sufficient for SDK reference designs)	https://www.keil.com/download/product (MDK-ARM v5)
	Description: Development environment specifically designed for microcontroller applications that lets you develop using the nRF51 SDK application and example files.	
2.	SEGGER J-Link Programmer Software	http://www.segger.com/jlink-software.html (J-Link software & documentation pack)
	Description: Software package required to use the J-Link programmer included in the development kit. (Contains drivers and files required to debug directly from the Keil Development Kit.)	

Table 2-3. Nordic Semiconductor Software Components

Nordic Semiconductor Components		
	Component	**Source**
1.	nRF51 SDK 9.0.0	https://developer.nordicsemi.com/nRF51_SDK/nRF51_SDK_v9.x.x/ (nRF514_SDK_9.0.0)
	Description: Software Development Kit that provides source code of examples and libraries forming the base of your application development. The nRF51 SDK includes: • Example code • ANT profile examples • *Bluetooth* profile examples • Drivers • Libraries • nrfjprog For more information, see the documentation packaged with the nRF51 SDK.	
2.	nRFgo Studio	https://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRFgo-Studio (nRFgo Studio-Win 32/64)
	Description: Tool to program and configure devices. It supports the programming of nRF51 SoftDevices, applications, and bootloaders. The different programming modes are available on individual tabs in the nRF51 programming module. Studio is used for the following: • Setting the supply voltage • Erasing flash memory • Programming a SoftDevice • Programming an application • Programming the bootloader For more information, see the help in nRFgo Studio.	
3.	nRF51422 Evaluation Kit User Guide	https://www.nordicsemi.com/eng/Products/ANT/nRF51422-Evaluation-Kit (nRF51-422-EK-UG-Keil)
	Description: User guide for the Nordic nRF51422 Evaluation Kit. It contains information for developing nRF51 applications with the Keil MDK ARM Development Kit.	
4.	S210 nRF51422 SoftDevice v5.0.0	https://www.nordicsemi.com/eng/Products/ANT/nRF51422 (S210-SD-V5)
	Description: Precompiled ANT protocol stack.	

Table 2-4. N5 Starter Kit Software Components

N5 Starter Kit Components		
	Component	**Source**
1.	N5 Starter Kit SDK Installer	www.thisisant.com/developer/resources/downloads (N5 Starter Kit SDK v 3.0)

	Description: N5 Starter Kit patch for the nRF51 SDK. The N5 Starter Kit SDK includes:
	• Binaries for ANT Network Processor Application (S210 and S310 SoftDevices)
	• Example Code
	  ○ ANT IO Demo (Refer to Section 5.1)
	  ○ ANT Message Types Demo (Refer to Section 5.2)
	  ○ ANT Background Scanning Demo (Refer to Section 5.3)
	  ○ ANT Relay Demo (Refer to Section 5.4)
	  ○ ANT Auto Shared Channel Demo (Refer to Section 5.5)
	  ○ ANT Bootloader/DFU Demo (Refer to Section 5.6)
	  ○ ANT Debug Demo (Refer to Section 5.7)
	  ○ ANT Scalable Channel Demo (Refer to Section 5.8)
	  ○ ANT Scalable Encrypted Channel Demo (Refer to Section 5.9)
	  ○ ANT Scan and Forward Demo (Refer to Section 5.10)
	  ○ ANT Asynchronous Controller using Continuous Scanning Mode (Refer to Section 5.11)
	• Header files to map nRF51422 IO pins to starter kit boards.
	Note that all of the example code provided in the N5 Starter Kit v3.0 is intended for use with the S210 SoftDevice version 5.0.0. The binaries for the ANT Network Processor Application are intended for use with the s210 and s310 SoftDevices versions 5.0.0 and 3.0.0, respectively.	

2.1 Installing the Software Tools

It is recommended the supporting Software Tools are installed in the order specified below to ensure functionality.

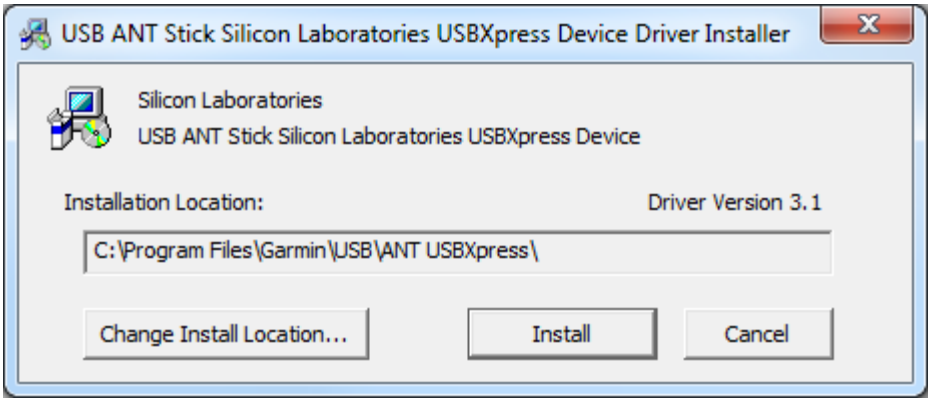
Table 2-5. Installation Order

Order	Software Component
1	ANT USB Interface Board Driver
2	ANTWareII
3	Keil MDK Arm Developer Kit
4	SEGGER J-Link Programmer Software
5	nRF51 SDK
6	nRFgo Studio
7	N5 Starter Kit SDK

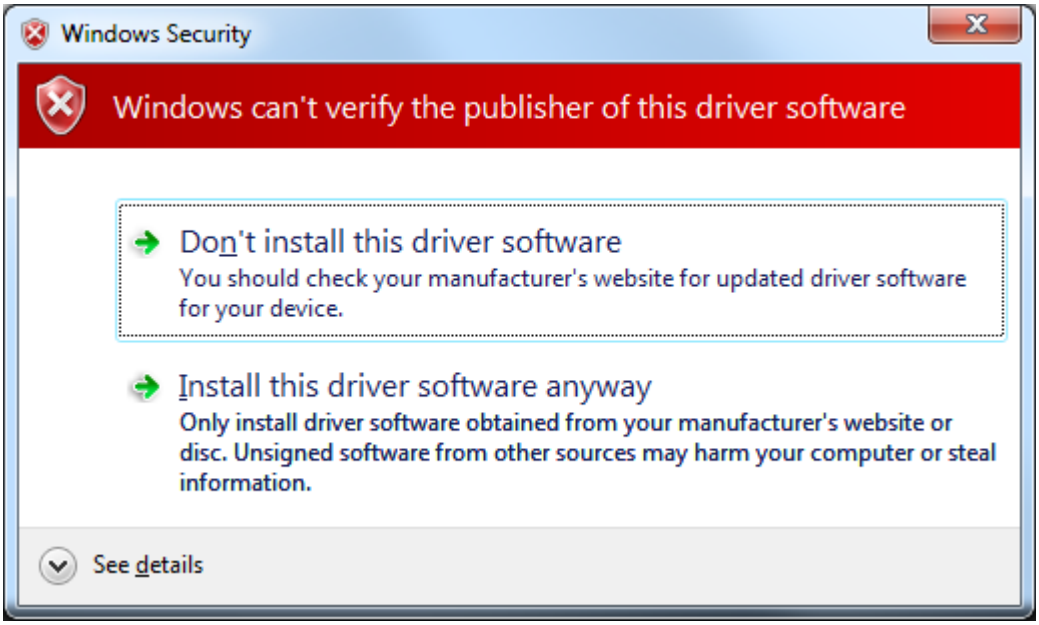
2.2 ANT USB Driver Installation

Download the 'ANT USB Interface Board Driver for Windows' package from http://www.thisisant.com/developer/resources/downloads and extract the entire contents onto your hard drive. Run the USBXpressInstaller.exe file contained in the folder.

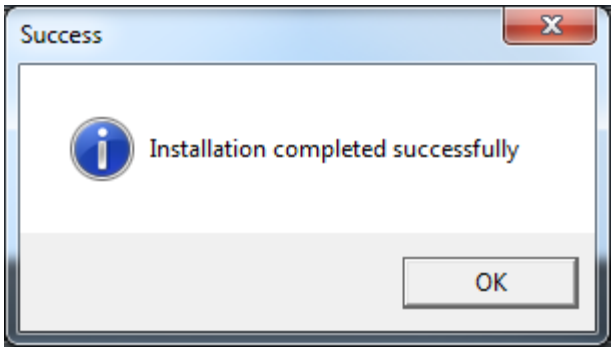
Install the drivers in your desired location.



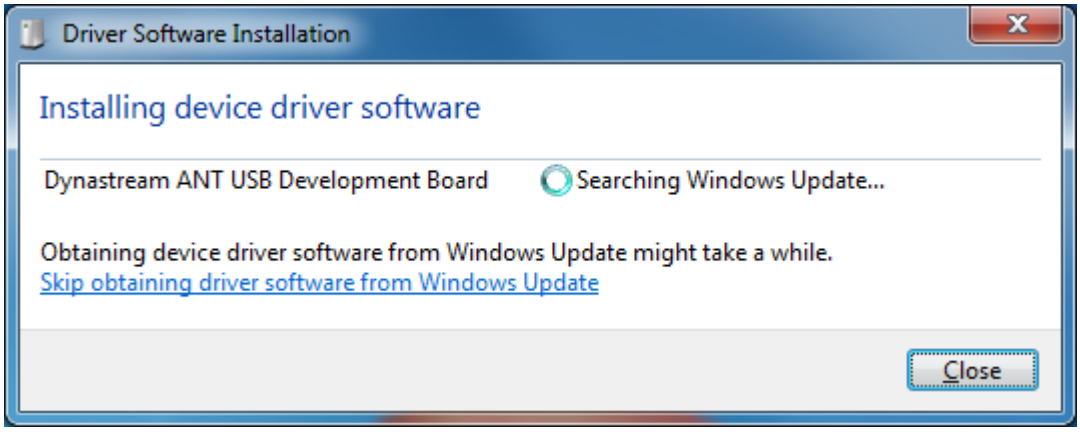
You may receive a warning message that indicates that Windows can't verify the publisher of the driver software. Click 'Install this driver software anyway' to continue.



A window will indicate if the drivers have installed correctly.



Connect an N5 module to the ANT Development Kit's USB interface board and insert into a USB port on your PC. The Driver Software Installation wizard should pop up and begin a search for drivers and indicate the USB device is 'Ready to Use' when it detects the installed drivers on the PC.



Note: The ANT USB Interface Board drivers are unsigned. Systems that require signed drivers for installation (e.g. Windows 8) are required to boot with driver signature enforcement disabled to complete the installation process.

2.3 Install ANTWareII

Download ANTWareII from the ANTWareII link on the www.thisisant.com/developer/resources/downloads page. Follow the steps in the installer to install ANTWareII on your computer. **Please note that the .NET Framework 3.5 must be installed on your PC in order for this application to run.**

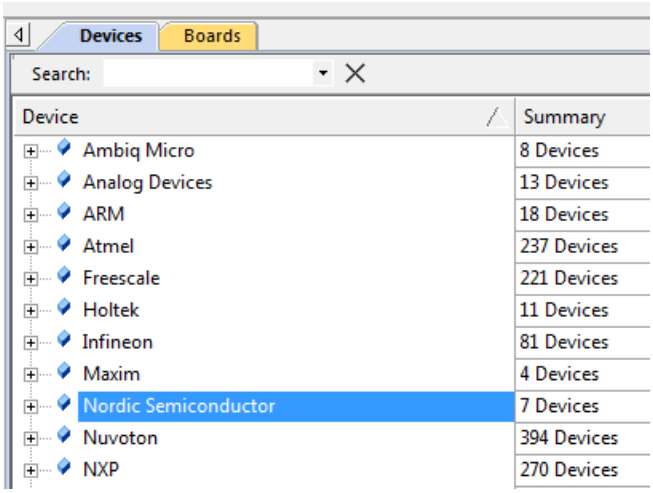
2.4 Install Keil MDK Arm Developer Kit

Download the Keil MDK Arm Developer Kit from the MDK-ARM v5 link on the https://www.keil.com/download/product/ page. Run the MDK installer. It is recommended that you use the default install locations.

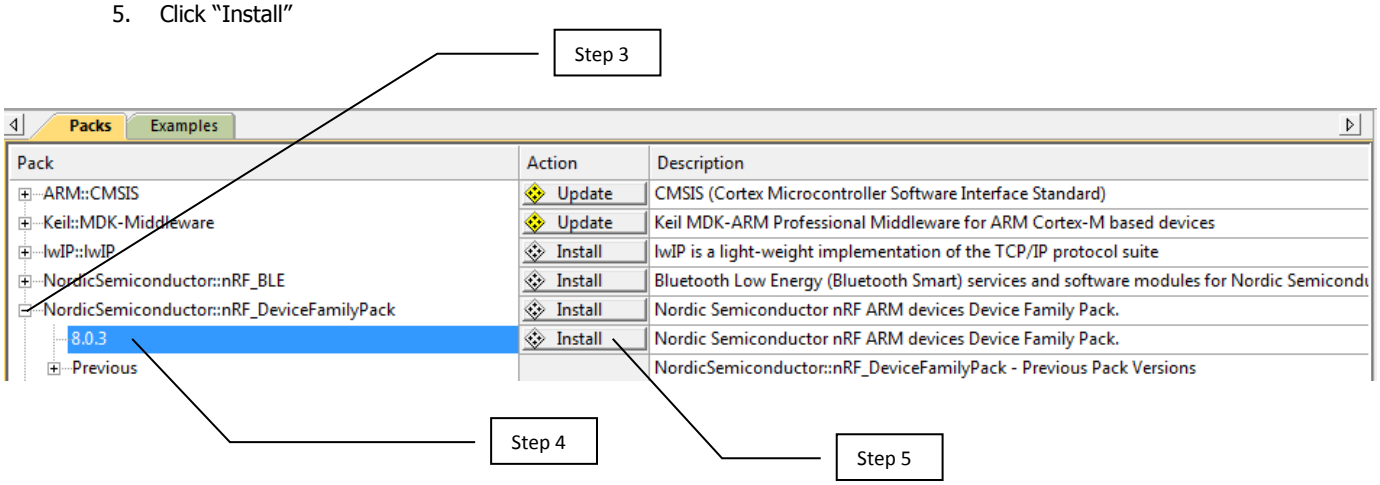
Continue to follow the steps in the installer. Upon completion of the installer, the Keil Pack Installer will automatically open.

Follow these steps to install the Nordic Semiconductor Device Family Pack:

1. Click "Packs" and select "Check For Updates"
2. Select "Nordic Semiconductor" within the "Devices" Tab.



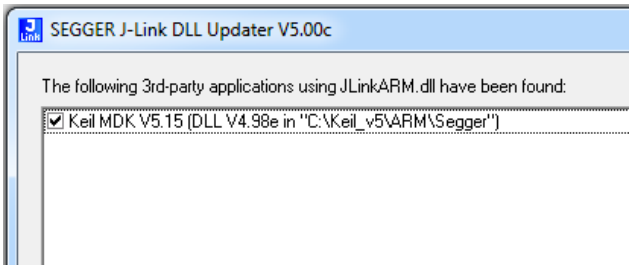
3. Expand "NordicSemiconductor::nRF_DeviceFamilyPack" within the "Packs" Tab.
4. Select the 8.0.3 Device Family Pack
5. Click "Install"



6. Exit the pack installer

2.5 Install SEGGER JLink Programmer Software

Download the Segger JLink Programmer from http://www.segger.com/jlink-software.html. Follow the steps in the installer to install the required Segger drivers on the computer. During the installation, any 3rd-party applications that use the JLinkARM.dll should be detected, select your version of Keil MDK.



2.6 Install Nordic nRF51 SDK

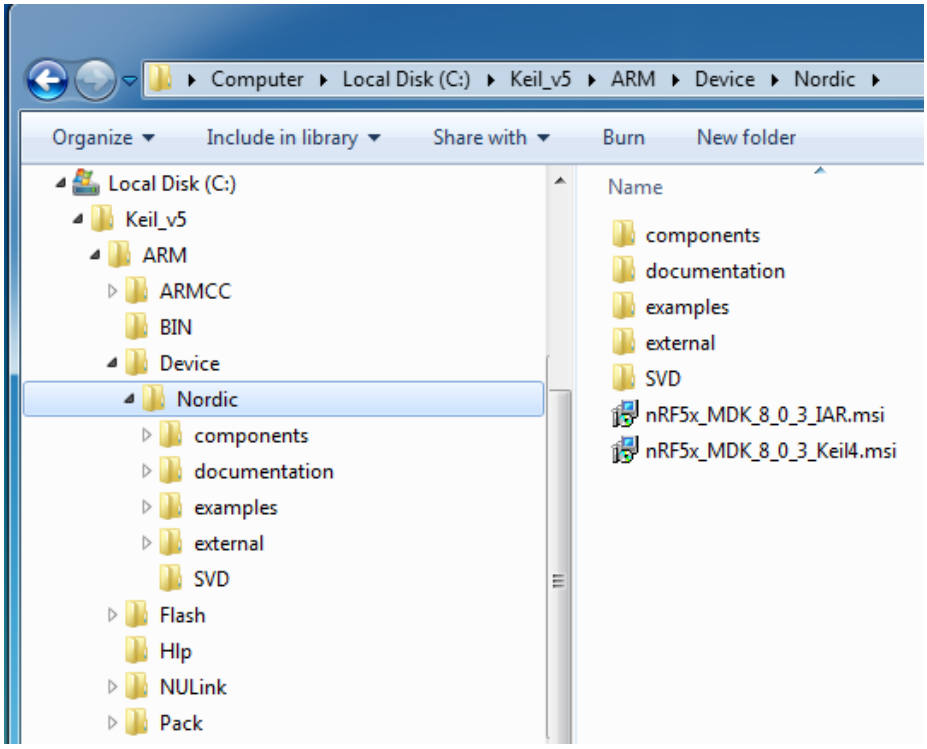
Download the Nordic nRF51 9.0.0 SDK zip file from https://developer.nordicsemi.com/nRF51_SDK/nRF51_SDK_v9.x.x.

Unzip the contents of the zip file in the following path, ensuring the directory structure in the zip file is preserved:

C:\Keil_v5\ARM\Device\Nordic

Note: This will require creating the Device and Nordic folders.

When finished you should have a directory structure like the image below:



Note: There is no need to run the nRFx_MDK_8_0_3_Keil4.msi or the nRFx_MDK_8_0_3_IAR.msi installer, these files were already installed with the Keil pack installer in Section 2.4

2.7 Install nRFgo Studio

Download the nRFGo Studio tool from https://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRFgo-Studio. Follow the steps in the installer to install nRFGo Studio on the computer.

2.8 Install N5 Starter Kit SDK

Download the zipped N5 Starter Kit SDK file from http://www.thisisant.com/developer/resources/downloads. It is important that the directory structure remain unchanged to avoid having to modify paths and project locations in the reference code.

1. Extract the entire contents of the N5 Starter Kit SDK onto your hard drive in a temporary location of your choice.
2. Move the entire folder n5_series from the extracted N5 Starter Kit SDK into the directory C:\Keil_v5\ARM\Device\Nordic\examples
3. Inside the bsp folder, move the files boards.h and n5_starterkit.h into the directory C:\Keil_v5\ARM\Device\Nordic\examples\bsp. Windows will detect a conflict since we are replacing the boards.h header file. When prompted, select "Move and Replace".

3 Programming the SoftDevice

The N5 modules that come in the N5 Starter Kit come preloaded with the SoftDevice s210 v3.0. The reference code provided in this package is intended for the SoftDevice s210 v5.0.0 or higher. To run the reference code, you must first program the appropriate SoftDevice onto the module by following the next steps:

1. Stack on the N5 module to be programmed onto the IO and Battery Boards, as shown in the figure below.

Figure 3-1. N5 Module stacked on IO and Battery Boards.

2. Insert CR2032 Coin Cell battery into the Battery Board.
3. Attach the JLink Segger Ribbon Cable to the programming header on the N5 module as shown in the figure below.

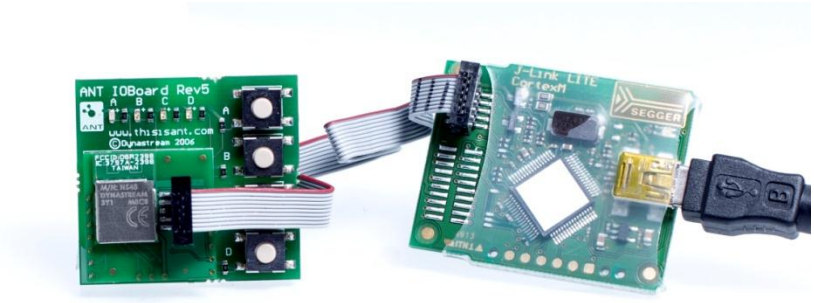
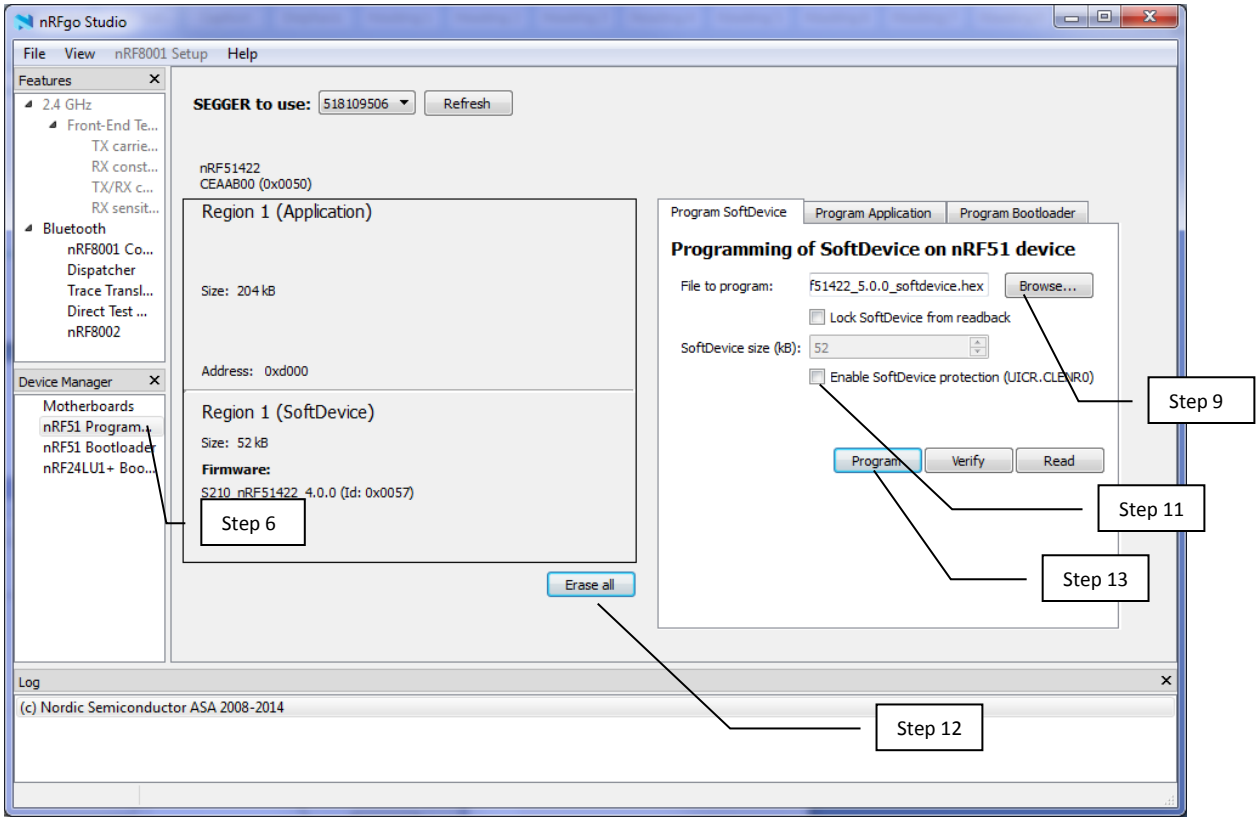


Figure 3-2. Stacked N5 Module with Segger JLink Programmer Ribbon Cable

4. Connect the JLink Programmer Segger to the computer using a USB to mini-USB cable.
5. Launch nRFgo studio.
6. Click on the "nRF51 Programming" option in the "Device Manager" panel on the left-hand side of the application.



7. Click on the "Program SoftDevice" tab on the right-hand side of the application.
8. Unzip the SoftDevice package (e.g. s210_nrf51422_5.0.0.zip).
9. Click the "Browse…" button and navigate to the directory where the hex file for the SoftDevice was unzipped to.
10. Open the SoftDevice hex file (e.g. s210_nrf51422_5.0.0.hex).
11. Disable the checkbox "Enable SoftDevice protection (UICR.CLENR0)" if you are planning on using the ANT Bootloader/DFU Demo.
12. Click the "Erase all" button at the bottom of the screen.
13. Click the "Program" button in the "Program SoftDevice" tab to program the SoftDevice onto the N5 Module.
14. The Log window at the bottom of the screen should read: "SoftDevice [SoftDevice hex file] programmed successfully".

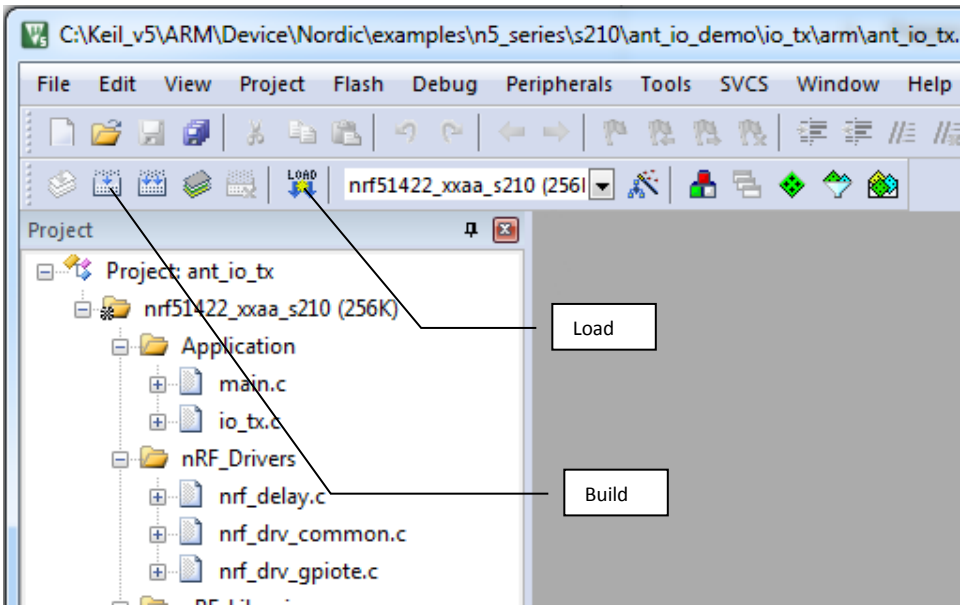
nRFGo Studio can also be used to program an application or a bootloader onto the N5 Module. Please refer to the "Help" > "Open Help" menu item in nRFGo for more details on the usage of this tool.

Important: It is not possible to program an N5 module while it is mounted to the USB interface board and connected to a PC.

4 Loading the Reference Code

In order to program one of the reference designs included in the N5 Starter Kit SDK onto a module:

1. Prepare the hardware and connect it to the computer as described in steps 1-4 of Section 3.
2. Locate the project. If the default nRF51 SDK and N5 Starter Kit SDK installation locations were used, all projects will be present under C:\Keil_v5\ARM\Device\Nordic\examples\n5_series\s210\
3. Open the project in Keil µVision by double clicking the .uvprojx file inside the arm directory of the project. For example, for the ant_io_demo (master), you would need to double click on the following file: C:\Keil_v5\ARM\Device\Nordic\examples\n5_series\s210\ant_io_demo\io_tx\arm\ant_io_tx.uvprojx
4. Click the "Build" icon to compile the project.
5. Click the "Load" icon to download the firmware onto the module and execute it.



Please refer to the "nRF51422 Evaluation Kit User Guide" document for more details on working with Keil.

In order to program the provided binaries (hex files) for the ANT Network Processor application using nRFgo Studio, refer to the instructions in Section 3, selecting the "Program Application" tab in step 7.

Important: When programming the provided binary for the ANT Network Processor application, the I/O board must **not** be connected to the module and battery board. The ANT Network Processor application includes bootloader activation support via a BOOT pin. Connecting the I/O board to the module that is programmed with the ANT Network Processor application will result in activation of the bootloader, and a 2 minute timeout will be required before being able to run the network processor application.

5 Reference Designs Overview

Most of the reference designs included in the N5 Starter Kit SDK can be tested using only the components included in the N5 Starter Kit, however, some of the examples illustrate extended topologies that benefit from the use of additional components. While describing the different components required for each demo, this document will use the following terminology:

N5 Development Board: consists of an N5 module stacked on an I/O and Battery board, as depicted in Figure 5-1.

Figure 5-1. N5 Development Board

ANT USB Stick: consists of an N5 module, loaded with ANT Network Processor firmware, mounted on an USB Interface board, as shown in Figure 5-2. The N5 module that comes stacked on the ANT USB interface board in the N5 Starter Kit is preloaded with ANT Network Processor firmware. ANT Network Processor Firmware provides emulation of a stand-alone ANT module with serial communication with external controllers. Alternatively, an ANTUSB-m or ANT USB2 stick can be used.

Figure 5-2. ANT USB Stick

5.1 ANT IO Demo

The ANT IO demo was designed to demonstrate the functionality of the LEDs and buttons of the I/O board in the N5 Starter Kit, as well as showcase basic bidirectional communication between two nodes using ANT.

The demo can be used with either one N5 development board and one ANT USB stick connected to the PC application ANTware (Figure 5-3) or two N5 development boards (Figure 5-4).

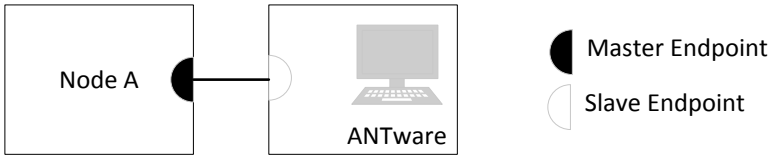


Figure 5-3. IO Demo Topology using an ANT USB stick

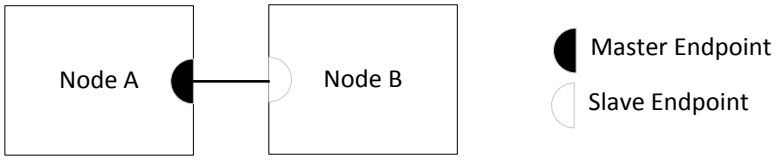


Figure 5-4. IO Demo Topology using two N5 development boards

5.1.1 Usage of the preloaded IO Tx Demo and Network Processor programs on the N5 module

The N5 Module that comes stacked on the I/O and Battery Boards comes preloaded with the example IO Tx demo. The N5 Module that comes stacked on the ANT USB interface board is preloaded with an ANT Network Processor firmware. Make sure the markings on the modules match the markings on the mounting boards. Follow the steps below to test the preloaded functionality of the modules:

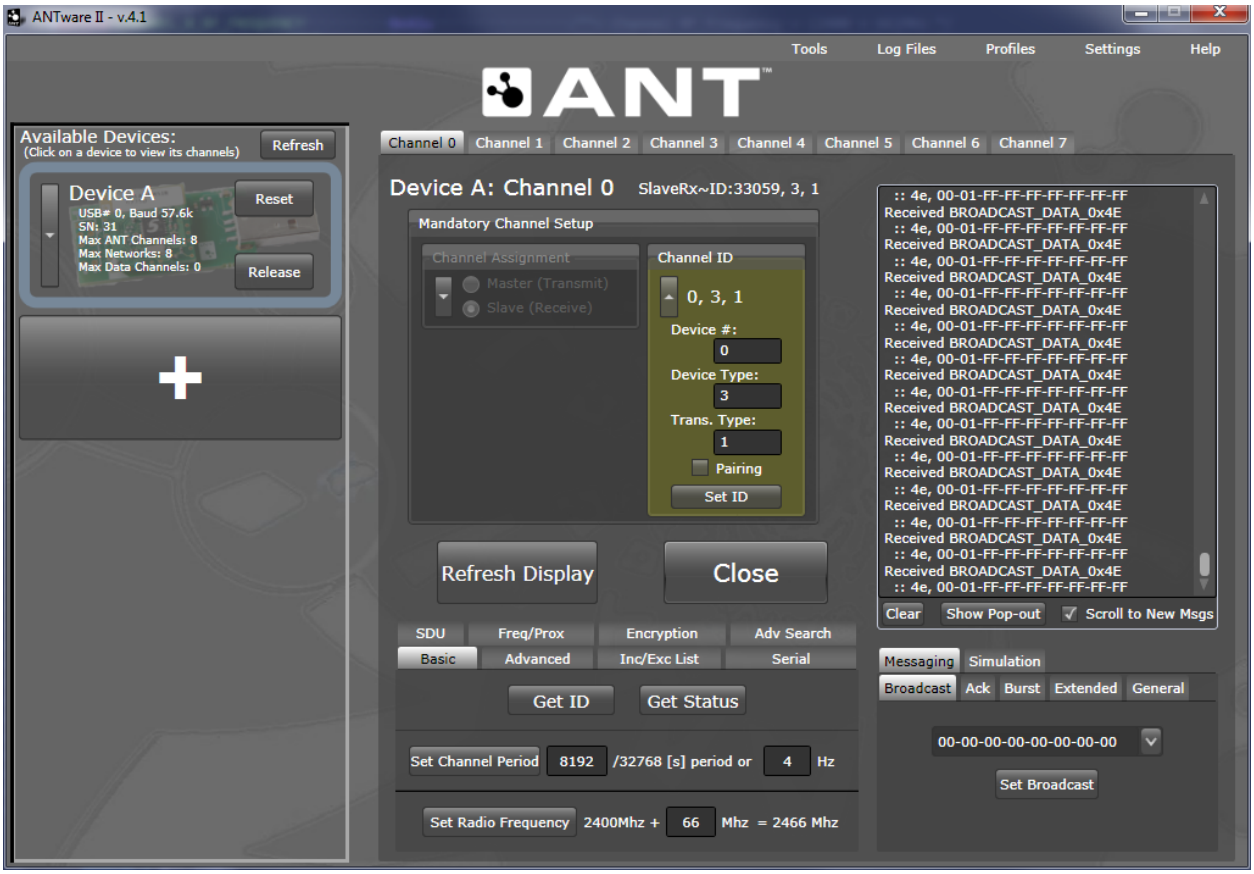
1. Insert a CR2032 coin cell battery into the N5 development board.
2. Connect an ANT USB stick into a USB port on your PC.
3. Open the ANTWare tool.
4. Configure Channel 0 as a Slave.
5. Set the Channel ID to the following values:

Table 5-1. IO Demo Channel ID

Parameter	Value
Device Number	0 (wild card)
Device Type	3
Transmission Type	1

6. Open the channel with default values for all the other channel configuration parameters to begin the search for the other N5 module.

7. Once the two modules have paired (indicated by receiving broadcasts in the ANTWare Log window), you can begin to test the Buttons and LEDs on the IO board.



8. Click on the "Simulation" tab on the right-hand side of the ANTWare application. Modify the "Respond With" field to either of these values:

Table 5-2. IO Demo LED Control Message Format

Tx Buffer Value	Description
01-00-00-00-00-00-00-FE	Turns on LED A on the IO Board
01-00-00-00-00-00-00-FD	Turns on LED B on the IO Board
01-00-00-00-00-00-00-FB	Turns on LED C on the IO Board
01-00-00-00-00-00-00-F7	Turns on LED D on the IO Board

9. Check the "Auto Send Response to Received Msgs" check box. The N5 module will now respond with a message of its own every time it receives an ANT broadcast message to control the LEDs on the IO board.

10. The buttons on the IO board will allow you to modify the contents of the broadcast messages being transmitted from the N5 module.

5.1.2 Usage of the IO Tx and Rx Demos

1. Insert a CR2032 coin cell battery into an N5 development board (Node A)
2. Load the reference code for the **ant_io_demo\io_tx** project onto the module, as described on Section 4.
3. Insert a CR2032 coin cell battery into a second N5 development board (Node B)
4. Load the reference code for the **ant_io_demo\io_rx** project onto the module, as described on Section 4.
5. Pressing the buttons on one of the nodes will control the state of the LEDs on the other node. For example, press button A on Node A; LED A on Node A will turn ON while the button remains pressed.

5.2 ANT Message Types Demo

The ANT Message Types reference demo demonstrates the three different ANT message types: Broadcast, Acknowledged, and Burst. The LEDs of the IO Board in this demo will give a visual presentation of the CPU sleep time and ANT active time using different ANT messages. The PC application ANTware is used to test the functionality of the demo.

This demo requires one N5 development board and one ANT USB stick.

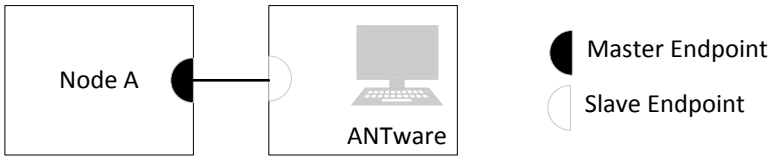


Figure 5-5. Message Types Demo Topology

5.2.1 Usage

1. Insert a CR2032 coin cell battery into the N5 development board.
2. Load the reference code for the **ant_message_types\message_types_master** project onto the module, as described on Section 4.
3. Plug in an ANT USB stick into the computer.
4. Open the ANTWare tool.
5. Configure Channel 0 as a Slave.
6. Set the Channel ID to the following values:

Table 5-3. Message Types Demo Channel ID

Parameter	Value
Device Number	0 (wild card)
Device Type	2
Transmission Type	1

7. Open the channel with default values for all the other channel configuration parameters to begin the search for Node A.
8. Once the two nodes have paired, the buttons on the N5 module with the IO board will allow you to switch between broadcast, acknowledged and burst messaging. Button A will switch the N5 module to broadcasting mode, Button B will switch it to acknowledgement mode and Button C will switch it to bursting mode. LED A will indicate when the CPU is sleeping, while LEDs B, C and D will indicate when a broadcast message, acknowledged message, or a burst transfer is being transmitted via ANT.

5.3 ANT Background Scanning Demo

The purpose of the ANT Background Scanning example is to show how to implement a background scanning channel in combination with a master channel on a single device, and to show how to use extended messages to obtain the RSSI (Received Signal Strength Indication) and channel ID information from received packets. It does not utilize any buttons or LED's.

Background scanning channels are a powerful feature in ANT because they allow for asynchronous topologies while still allowing other channels to run independently on the same device. In this example, the RSSI and channel ID of messages received on the background scanning channel are copied to the data payload of the master channel, effectively mirroring these parameters back to the transmitting devices. For more details, please consult the "ANT Message Protocol and Usage" document as well as the following application notes: "ANT AN ANT Channel Search and Background Scanning Channel" and "ANT AN RSSI Extended Information".

This example can work with any number of nodes (Figure 5-6) or it can work with a single development board in conjunction with an ANT USB stick and the ANTware PC application (Figure 5-7). The usage instructions in this document describe the second setup. In this example ANTware will be configured to interface to a single node. A slave channel will be configured in ANTware to track the master channel of the node. A master channel will be opened up in ANTware to match the channel parameters of the background scanning channel.

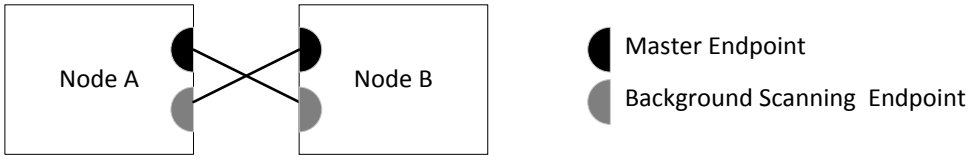


Figure 5-6. Background Scanning Demo Topology with 2 Nodes

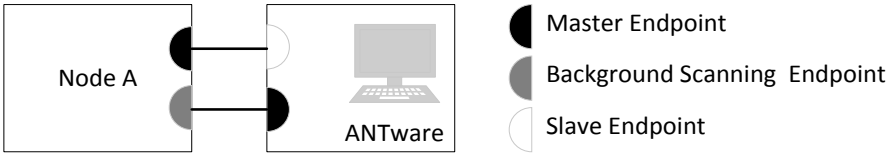


Figure 5-7. Background Scanning Demo with ANTware

5.3.1 Operation

After start up or reset each ANT node will open up two channels - one background scanning and one master channel. Note that the background scanning channel is configured similar to a regular slave channel with a flag set to denote its special function. As a background scanning channel is effectively a searching channel that never synchronizes to any master, it is important to set the search timeout appropriately. In this example the search timeout is set to infinity.

Table 5-4. Background Scanning Demo Channel Parameters

Parameter	Master Channel	Background Scanning Channel
Channel Type	Master (0x10)	Slave(0x00)
Extended Assignment	N/A	EXT_PARAM_ALWAYS_SEARCH (0x01)
Network	Public	Public
Radio Frequency	2477MHz	2477MHz
Channel Period	16Hz (2048)	N/A
Device Number	Serial Number	Wildcard (0)
Device Type	1	1
Transmission Type	5	5
Search Timeout (High Priority)	N/A	No HP Search (0)
Search Timeout (Low Priority)	N/A	Infinite (255)

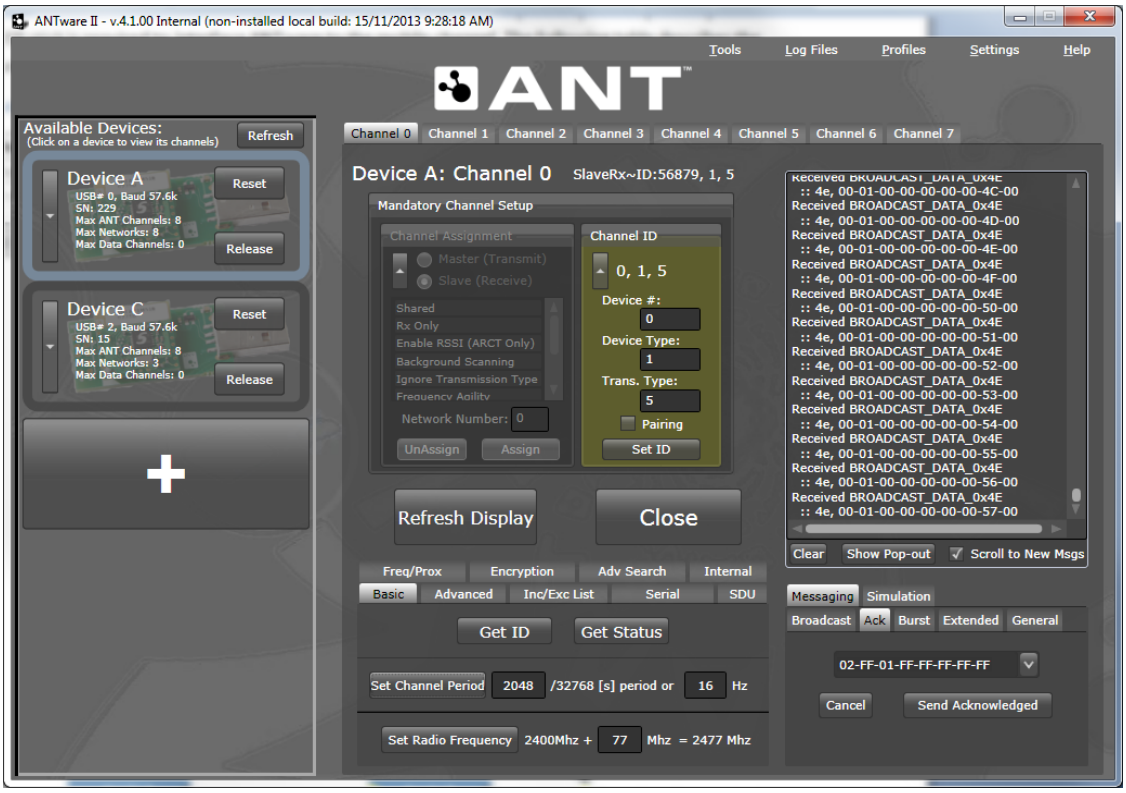
The master channel on each device sends a single payload of data indicating the signal strength and channel ID of the last message that was received on the background scanning channel. The format of this message is detailed below.

Table 5-5. Message Format for Default Master Message

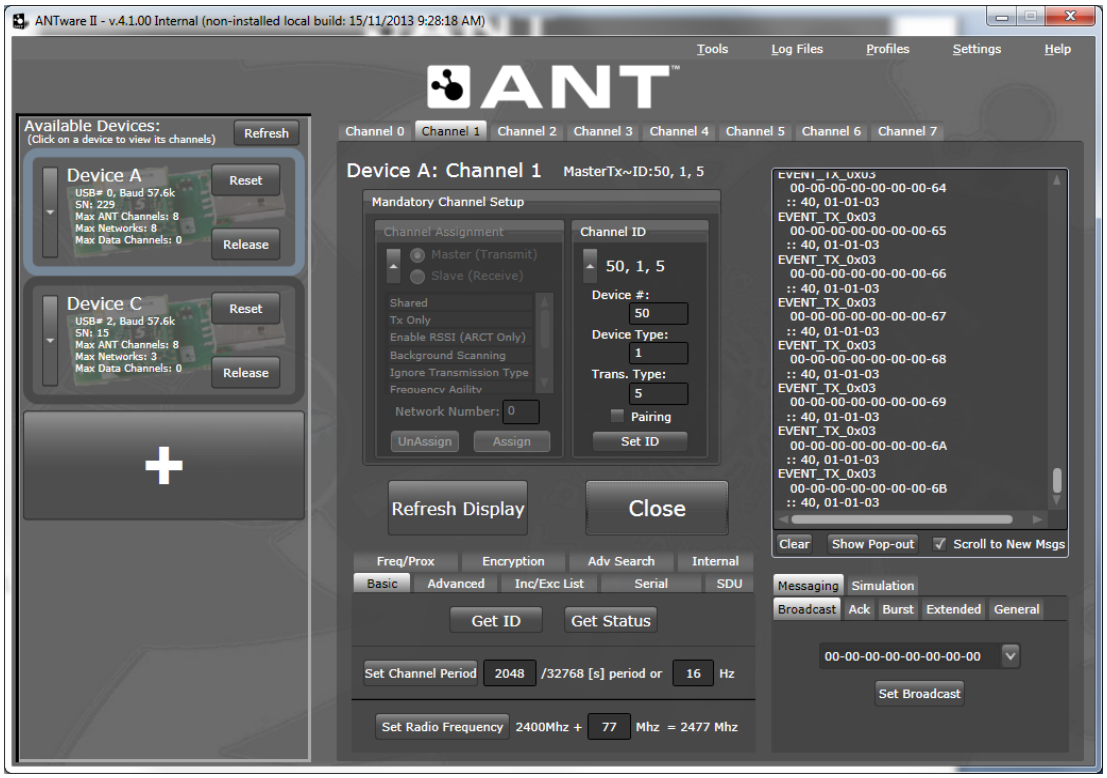
Byte	Description
0	Page Number = 1 (ANT_BEACON_PAGE)
1	RSSI (signed dBm value corresponding to Device Number below)
2	Device number of last message received on background scanning channel (little endian). Corresponds to the RSSI value above
3	
4-5	Reserved
6	Counter which increases with each channel period
7	Number of messages received on background scanning channel

5.3.2 Usage

1. Insert a CR2032 coin cell battery into the N5 development board.
2. Load the reference code for the **ant_background_scanning** project onto the module, as described on Section 4.
3. Plug in an ANT USB stick into the computer.
4. Open the ANTWare tool.
5. Configure Channel 0 as a Slave.
6. Configure the Device Type, Transmission Type, Channel Period and Radio Frequency as specified in the "Master Channel" column of Table 5-4. Set the Device Number to 0 (wild card).

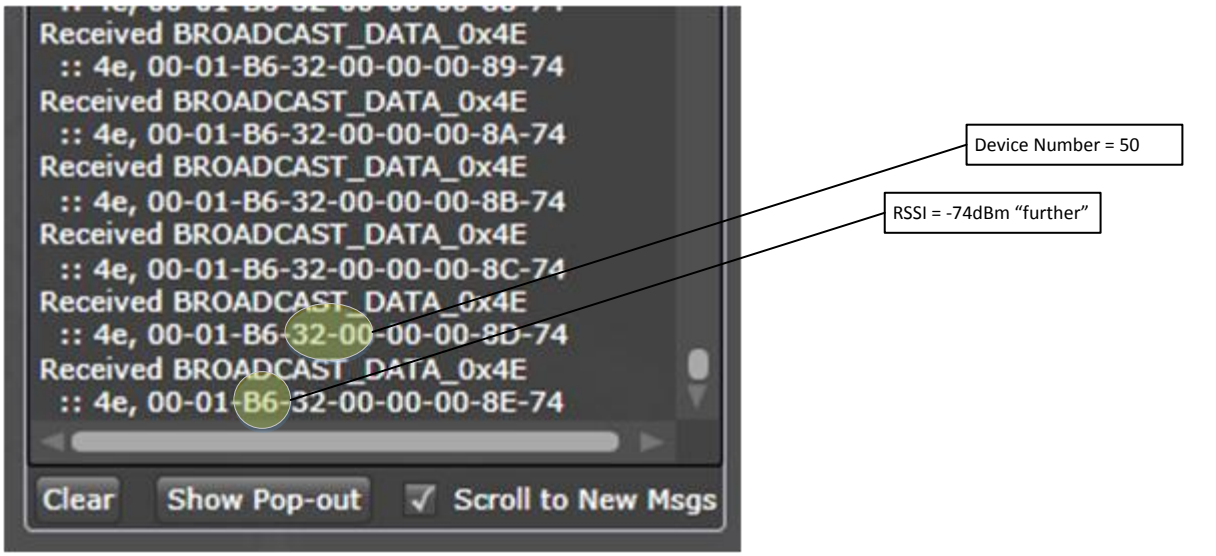


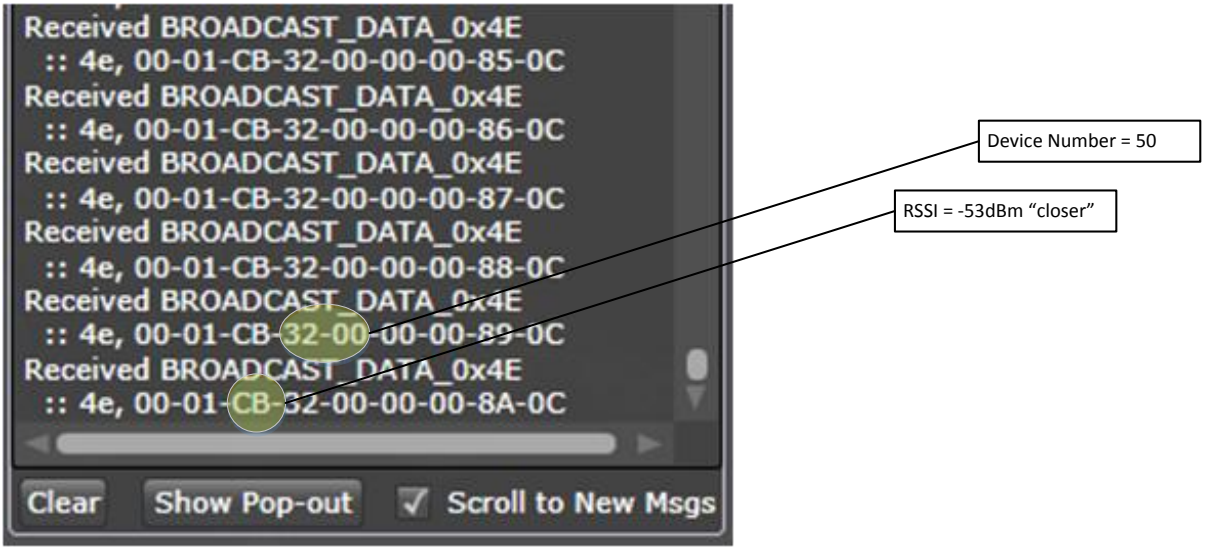
7. Configure Channel 1 in ANTware to broadcast messages to the background scanning channel running on Node A. To do this, configure Channel 1 as a Master, and configure the Device Type, Transmission Type, Channel Period and Radio Frequency as specified in column 'Master Channel' of Table 5-4. The device number can be set to any



value except 0. Below the device number is set to 50. The payload transmitted by this channel is not relevant.

8. Observe the message payload received on Channel 0 on ANTware. Note that the channel ID reported will equal 50 and the corresponding RSSI value will change as the N5 development board is moved closer and further away from the ANT USB stick.





Note that any number of channels can be configured in ANTware to transmit to the background scanning channel on Node A. Make sure to use a different device number for each of the channels. The background scanning channel will receive messages from these asynchronously and report back to the slave channel configured in ANTware.

5.4 ANT Relay Demo

The ANT Relay Demo implements a simple relay network utilizing ANT's multi-endpoint and proximity pairing capabilities. Nodes can be connected in a daisy chain with no limitation as to the number of devices connected. Each node is connected to the previous one using proximity pairing. The state of an LED is relayed among all of the nodes in the network, i.e., if the state of the LED on one node changes, the state of the LED's on all nodes will also change. The state of the LED can be changed by pressing a button on any of the boards, or by connecting to any of the nodes wirelessly using the ANTware PC application. This demo can be extended to fixed topologies that require range extension through relay and report status or accept commands. This network configuration is appropriate for coin cell operated devices and applications that require a mobile interface.

To run this demo at least two N5 development boards are required. Optionally, an ANT USB stick is required to interface ANTware to the mobile channel.

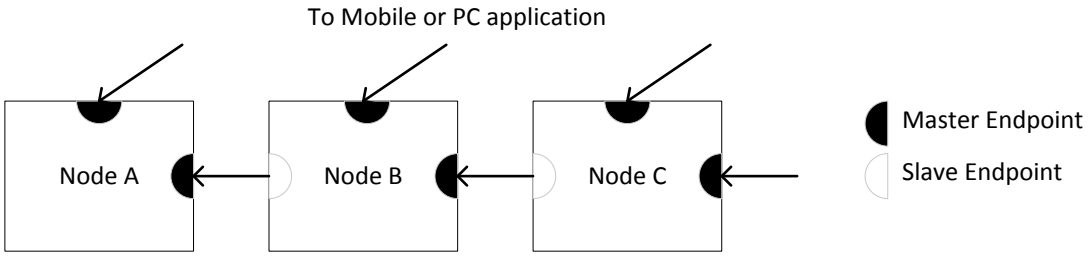


Figure 5-8. ANT Relay Demo Topology

5.4.1 Operation

After start up or reset each ANT node will open up two master channels: one for the purpose of the relay network and the other to allow for connections to a mobile or PC application (for example ANTware).

The table below describes the channel parameters for each of these master channels.

Table 5-6. Relay Demo Channel Parameters

Parameter	Mobile Channel	Relay Channel
Channel Type	Master (0x10)	Master (0x10)
Network	Public	Public
Radio Frequency	2477MHz	2472MHz
Channel Period	4Hz (8192)	4Hz(8192)
Device Number	Serial #	Serial #
Device Type	2	1
Transmission Type	5	5

The user can pair one board to another by pressing a button on the board they wish to pair, or by sending the pairing command over the mobile channel (refer to Table 5-9). Proximity pairing is used (with a bin setting = 1) so boards must be brought very close to each other (<10cm) in order to pair. Once a connection to an adjacent node is made, the state of the LED on the board is relayed among all of the boards. The relay channel is 100% bi-directional, with the slave endpoint sending a status message back to the master for every message that it receives. The following 8 byte message is used to relay information between the devices.

Table 5-7. Relay Status Message

Byte	Description
0	Page Number = 1 (ANT_RELAY_MAIN_PAGE)
1	LED Status (1 = ON, 0 = OFF)
2	Counter which increases every time the state of an LED changes. If the counter changes, the node will change the state of its LED to reflect the change. The format of this 4 bytes unsigned value is little endian.
3	
4	
5	
6	Reserved = 0xFF
7	State of Slave Channel (0 = unassigned, 1 = assigned, 2 = searching, 3 = tracking)

If the state of the LED changes a counter will increase and all boards will update the state of their own LED to reflect the new state. The state of the LED can be changed by pressing a button on any of the boards, or by sending the appropriate command over the mobile channel.

The status message sent on the mobile channel to the PC or mobile application is almost identical as that sent by the relay channels. This message is detailed below.

Table 5-8. Mobile Channel Status Message

Byte	Description
0	Page Number = 1 (ANT_MOBILE_MAIN_PAGE)
1	LED Status (1 = ON, 0 = OFF)
2-6	Reserved = 0xFF
7	State of Slave Channel (0 = unassigned, 1 = assigned, 2 = searching, 3 = tracking)

The PC or mobile application can send a message to any device it is connected to and affect the LED or pairing state. The following table describes the format of the command used to emulate the buttons on the board.

Table 5-9. Mobile Channel Command Page

Byte	Description
0	Page Number = 2 (ANT_MOBILE_COMMAND_PAGE)
1	Reserved = 0xFF
2	Command (1 = Pairing, 2 = LED on, 3 = LED off)
3-7	Reserved = 0xFF

The command page may be sent using broadcast or acknowledged messages. Where possible it is recommended to use acknowledged messages. This message type will inform whether the command was successful or not.

If a slave channel loses connection with the master it is connected to for longer than 30s it will close its channel and re-initialize the search proximity bin setting. A connection can be re-initialized by the user by pressing the appropriate button or sending the correct command over the mobile channel.

5.4.2 Usage

The following table describes the function of the buttons and LEDs.

Table 5-10. Function of Button and LEDs in Relay Demo

Hardware	Function
Button A	Toggle state of LED C
Button B	Start Pairing mode
LED C	Toggled by button A
LED D	Off by default. Turns on when Slave Endpoint receives a message.

To test the example code, perform the following steps:

1. Insert a CR2032 coin cell battery into the N5 development board.
2. Load the reference code for the **ant_relay_demo** project onto the module, as described on Section 4.
3. Repeat steps 1-2 for as many nodes as desired.
4. After power up, each of the boards will configure a master channel transmitting every channel period, but they will not be connected to each other. Pressing button A on each board will affect the state of LED C, but only on the board where the button is pressed. LED D will be off, indicating that no messages have been received on the slave endpoint.
5. To pair two nodes, press button B on Node B. This will open the slave endpoint in Node B, which will begin to search for other devices within its proximity pairing zone.
6. Bring Node A and Node B close to each other (<10 cm) to pair. Once the slave endpoint on Node B begins to receive data from Node A, it will turn LED D on.
7. Once the two nodes are connected, they will continue communicating without the requirement of being in close proximity of each other. Pressing button A on either of the boards will toggle the state of the LED C on both of the boards. There will be slight latency as the message percolates across the relay. Subsequent boards (for example Node C) can be added by repeating the procedure and pairing to Node B and so on.

Optional steps to use ANTware to interact with the nodes using the mobile channel:

8. Plug in an ANT USB stick into the computer.
9. Open the ANTWare tool.
10. Configure Channel 0 as a Slave.
11. Configure the Device Type, Transmission Type, Channel Period and Radio Frequency as specified in the "Mobile Channel" column of Table 5-6. Set the Device Number to 0 (wild card) to connect to any of the boards.
12. Turn the LED on using the command described in Table 5-9. Select the "Ack" tab, set the message payload to 02-FF-02-FF-FF-FF-FF-FF, and click "Send Acknowledged".
13. Turn the LED off using the command described in Table 5-9. Select the "Ack" tab, set the message payload to 02-FF-03-FF-FF-FF-FF-FF, and click "Send Acknowledged".
14. Pairing can be initiated by sending the command described in Table 5-9. Select the "Ack" tab, set the message payload to 02-FF-01-FF-FF-FF-FF-FF, and click "Send Acknowledged". This is equivalent to pressing button B on the particular node ANTware is connected to.

5.5 ANT Auto Shared Channel Demo

The ANT Auto Shared Channel Demo implements a polling network between a master device and multiple slave devices using an ANT shared channel. This configuration is suitable for applications where a central node retrieves data periodically from a large number of sensors. For more information on shared channels, refer to the "ANT Message Protocol and Usage" document.

In order to allow on the fly addition/removal of slaves to the network, this demo implements a handshaking mechanism to dynamically assign addresses to the slave devices. For more details on the messaging involved in assigning addresses to the shared slaves, refer to the application note "AN Auto Shared Channel Master Example".

To run this demo at least two N5 development boards are required. Using single byte addressing, up to 253 N5 development boards acting as shared slaves can be added to the network.

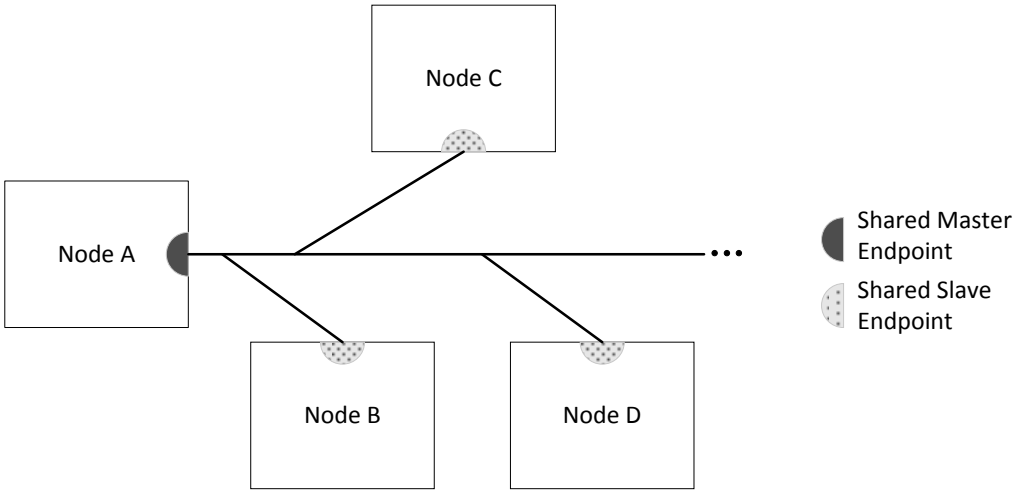


Figure 5-9. Auto Shared Channel Demo Topology

5.5.1 Usage

The following table describes the function of the buttons and LEDs on the auto shared master (Node A).

Table 5-11. Function of Button and LEDs on Auto Shared Master

Hardware	Function
Button A	Turn on LED A on itself and any connected slaves.
Button B	Turn off LED A on itself and any connected slaves.
LED A	Toggles on start up to indicate the application is running. Controlled by buttons A and B.
LED B	Toggles on ANT events
LED D	Off by default. Toggles for 200 ms when a slave joins or leaves the network. Toggles continuously on unrecoverable error conditions.

The following table describes the function of the buttons and LEDs on the auto shared slaves (Nodes B, C, D…).

Table 5-12. Function of Button and LEDs on Auto Shared Slave

Hardware	Function
LED A	Controlled by buttons A and B of the auto shared master.
LED C	Toggles when the node is polled by the auto shared master.
LED D	Indicates if the node is connected to the auto shared master. Connected = ON, Not Connected = OFF.

In order to test the example code:

1. Insert a CR2032 coin cell battery into an N5 development board (Node A).
2. Load the reference code for the **ant_shared_channel\ant_shared_channel_master** project onto the module, as described on Section 4.
3. Insert a CR2032 coin cell battery into another N5 development board (Node B).
4. Load the reference code for the **ant_shared_channel\ant_shared_channel_slave** project onto the module, as described on Section 4.
5. Repeat steps 3-4 for as many slave nodes as desired.
6. After power up, Node A will configure a shared master channel. The nodes programmed with the auto shared channel slave firmware (e.g., Node B, C, D…) will configure a shared slave channel and will begin to search for an auto shared master within its proximity pairing zone. Within 30 seconds of powering up a slave node (e.g. Node B), bring it onto close proximity (<10 cm) to Node A to pair. Once the slave node registers and obtains an address from Node A, it will turn LED D on.
7. Repeat step 6 to connect additional slave nodes to Node A.
8. Once the nodes are connected, they will continue communicating without the requirement of being in close proximity to Node A. Press buttons A or B on Node A to toggle the state of LED A on all nodes.

5.6 ANT Bootloader/DFU Demo

The ANT Bootloader/DFU example shows how Over-the-Air (OTA) updates for a SoftDevice, bootloader or application can be performed on the nRF51422, using ANT File Share (ANT-FS) technology.

The ANT Bootloader/DFU example in the N5 SDK v3.0.0 currently supports the following SoftDevices

- SoftDevice s210 v5.0.0 or later
- SoftDevice s310 v3.0.0 or later

The example code can be used to perform the following type of updates:

- Update the application code, with or without an existing application on the device.
- Update the bootloader, as long as the start address of the new bootloader matches that of the existing bootloader.
- Update the SoftDevice to a newer version of the same SoftDevice.
- Swap from SoftDevice s210 v5.0.0 or later to SoftDevice s310 v3.0.0 or later, and viceversa.
- Update the SoftDevice and the bootloader to newer versions in a single process.

If there is no existing application on the device, the device will start in the bootloader mode. In case an existing application is present, the device needs to be put explicitly into bootloader mode via a button press (refer to the Usage section), or by having the application request to enter bootloader mode (refer to the Passing information between Application and Bootloader section).

In case the device has an existing application when performing an update, the example code will attempt to preserve the existing application if possible, using a "dual-bank" update. Refer to Section 5.6.1 for more details.

Important: If the device has an existing application when updating the SoftDevice, the existing application will be erased. After updating the SoftDevice, make sure to install an application on the device.

The ANT Bootloader/DFU example currently supports updating one image at a time, except by updating both the bootloader and SoftDevice, which can be performed in a single session. If updating both the SoftDevice and the bootloader, it is highly recommended to perform this update in a single process, including both SoftDevice and bootloader images in the file uploaded to the device, to ensure that the versions of the bootloader and SoftDevice running on the device are known to be compatible. For more details on this, refer to the Transport Mechanism: ANT-FS section, as well as to the application note Over the Air Firmware Updates Using ANT-FS.

Important: Before updating the SoftDevice, make sure to carefully review the release notes and the migration notes for the new SoftDevice, and test the application and bootloader code to ensure their compatibility with the new SoftDevice. Migration between particular SoftDevices may require special steps. Any special requirements will be outlined in the migration document. These documents are typically released together in the same package as the SoftDevice.

5.6.1 Memory Layout

The ANT Bootloader/DFU example reserves the flash region between 0x3B800 – 0x40000 for the bootloader. The region between 0x0000 – 0x1000 is reserved for the Master Boot Record. The SoftDevice occupies flash region starting at address 0x1000, with size depending on the specific SoftDevice (s210/s310). The remaining space is divided into two memory banks of equal size, BANK 0 (starting after the SoftDevice) and BANK 1 (ending before the bootloader). If an application is present in the device, it is placed in the space right after the SoftDevice, and depending on its size, may occupy part or the entirety of BANK 0, and none, part or the entirety of BANK 1. This is illustrated in Figure 5-10.

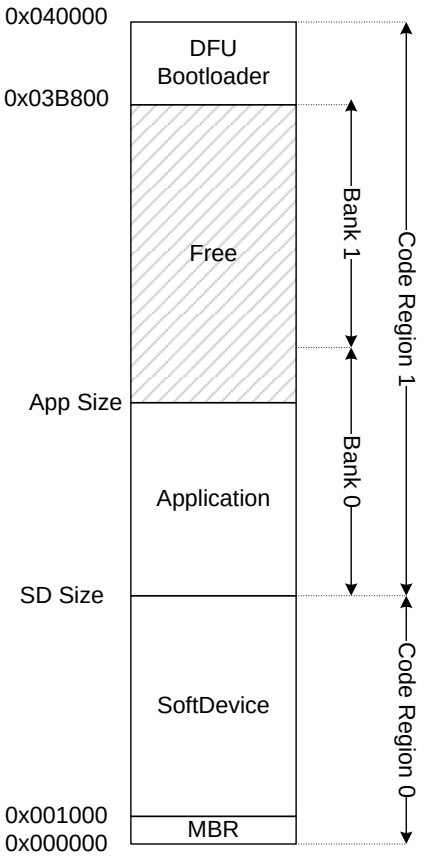


Figure 5-10. Memory Layout

Table 5-13 lists the memory ranges when using the SoftDevices s210 v4.0.0/4.0.1 and s310 v2.0.0/2.0.1. Thus, bank size when using a s210 SoftDevice is 0x17400, and when using a s310 SoftDevice is 0xF400.

Table 5-13. Memory ranges for S210/S310 SoftDevices

Usage	Memory Range (s210 v4.0.0/4.0.1)	Memory Range (s310 v2.0.0/2.0.1)
DFU Bootloader and data	0x3B800 – 0x40000	0x3B800 – 0x40000
Application Code (Bank 0), Free/Swap (Bank 1)	0x0D000 – 0x3B800	0x1D000 – 0x3B800
SoftDevice	0x01000 – 0x0D000	0x01000 – 0x1D000
Master Boot Record (MBR)	0x00000 – 0x01000	0x00000 – 0x01000

The example code attempts to preserve an existing application in the event of failures during the update process whenever possible; this is referred to as a dual bank update. This is achieved by storing the new image exclusively on BANK 1 if

possible, and swapping it to its destination location on activation. Use of BANK 0 while receiving a new image will result in not preserving the existing application; this is referred to as a single bank update.

In the example code, selection between dual and single bank mode is performed automatically depending on the size of the total image transferred to the device, current application size, and type of update requested. Dual bank updates are used by default, except under the following conditions:

- The application current size (as reported by the application using the app_size parameter) exceeds the bank size.
- The application current size (as reported by the application using the app_size parameter) is unknown.
- The size of the image received (as reported in the header of the SUF file) exceeds the bank size.
- A SoftDevice update was requested.

5.6.2 Transport Mechanism: ANT-FS

ANT-FS is a session based transport mechanism designed to securely and automatically transfer data files between two ANT enabled devices. A detailed description of ANT-FS is available in the ANT-FS Technical Specification. The specific details of the message exchange used for the update process as defined by ANT-FS are outlined in the application note ANT AN Over the Air Firmware Updates Using ANT-FS, including the format of files exchanged.

The ANT Bootloader/DFU example implements an ANT-FS client. A PC implementation of an ANT-FS host customized for performing OTA updates out of the box is provided in the OTA Updater tool. This tool automates all of the interactions between host and client to query device information and transfer new images to the device.

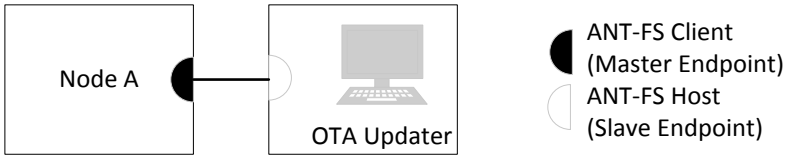


Figure 5-11. ANT Bootloader/DFU Topology

To establish communication, channel parameters must match in the host and client devices. The OTA Updater tool is preconfigured with the correct channel parameters required to interface with the ANT Bootloader/DFU example. The default channel parameters used in the example code and the OTA Updater PC tool are listed in Table 5-14.

Table 5-14. ANT Bootloader/DFU default channel parameters

Parameter	Value
Channel Type	Master
Radio Frequency	2450 MHz
Network Key	ANT-FS Managed Network Key
Device Type	16
Transmission Type	5
Device Number	Least significant 2 bytes of the ANTFS_CLIENT_SERIAL_NUMBER
Channel Period	8192 (4Hz)

Important: The ANTFS_NETWORK_KEY compile time configuration option must be set to the ANT-FS network key. The ANT-FS network key can be obtained at http://www.thisisant.com/developer/ant-plus/ant-plus-basics/network-keys.

5.6.3 Device Identification

The following compile time configuration options are available to customize how the device identifies itself:

- ANTFS_CLIENT_MANUF_ID: This 2-byte value identifies the manufacturer of the device. Manufacturer ID values are managed by ANT+. The current list of manufacturer ID values can be found in the FIT.xls profile (available within the FIT SDK at www.thisisant.com). New manufacturers are required to be members of the ANT+ Alliance in order to be added to this list; please contact the ANT+ Alliance at antalliance@thisisant.com for details. The value 255 (0x00FF) has been reserved as a development ID and may be used by manufacturers that have not yet been assigned a value. The ANT Bootloader/DFU example uses the development ID by default.
- ANTFS_CLIENT_DEV_TYPE: This 2-byte value is managed by each manufacturer and can be used to provide information regarding the type of device (e.g. model number). The example code uses value 1 by default.
- ANTFS_CLIENT_SERIAL_NUMBER: The ANT Bootloader/DFU example sets this value to the 4 least significant bytes of the chip device identifier in FICR.
- ANTFS_CLIENT_NAME: UTF-8 string that can be used to provide a human readable descriptor of the device. The example code sets this value to "ANTFS OTA Update". The string length can be customized with ANTFS_CLIENT_FRIENDLY_NAME_MAX; this length should not exceed 256. It is highly encouraged to customize this string, especially if using the development Manufacturer ID.
- OTA_INFO_HARDWARE_VERSION: This value is used to report the hardware version of the device in the OTA Update Information file. The value is managed by the manufacturer, and is set to 0 by default.
- OTA_INFO_REGION_PRODUCT_ID: Manufacturer specific 1-byte field that can be used to provide additional information regarding the type of device to be updated. For example, this field can be used to identify region specific versions of a product, or different SKUs based on the same hardware. This field is included in the OTA Update Information file.

5.6.4 Authentication

The example code supports passthru authentication only. For more details on ANT-FS authentication types, refer to the ANT-FS Technical Specification. Pairing/passkey authentication are not supported in the current version.

5.6.5 Over the Air Update Information File

The ANT Bootloader/DFU example demonstrates the use of an OTA Update Information file to provide information about the current application, bootloader and SoftDevice version that are installed on the device. The version string for the SoftDevice is populated by the bootloader, queried from the SoftDevice through sd_ant_version_get(). The version string for the bootloader is hard coded, and can be configured in version.c. The version string for the application version must be configured from the application, using ant_boot_settings_api.

5.6.6 Passing information between Application and Bootloader

The example code reserves a memory block located towards the end of the flash memory page to allow passing parameters between application and bootloader, even across power cycles. The parameters are stored in the ant_boot_settings_t structure, and include:

- app_version: Application version string (UTF-8), as reported by the OTA Update Information file. The maximum length of this string is 16; any unused characters must be set to the null character.
- app_size: This value can be used to indicate the size of the current application. The size is used to determine whether the current application can be preserved in the event of an update failure. If unknown, set to 0xFFFFFFFF; the bootloader will not attempt to preserve the existing application in this case.
- param_flags: The parameter flags is a 32-bit bitfield which contains control and status flags.
- Table 5-15 describes the format of this bitfield.

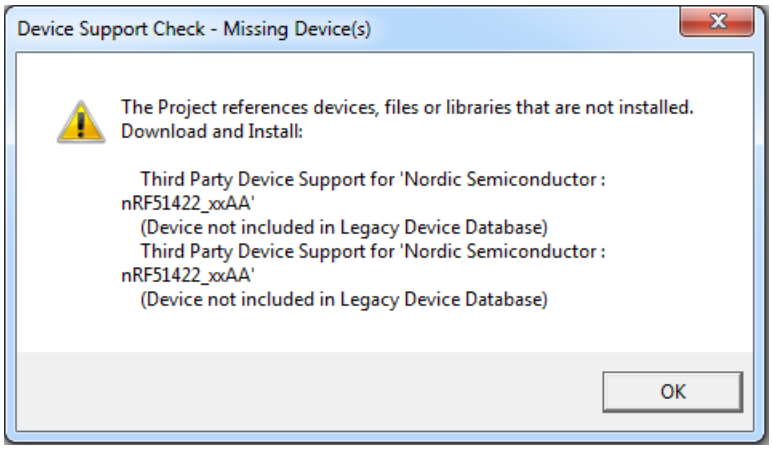
Table 5-15. Parameter flags

Bit(s)	Field	Value	Description
32-3	Reserved	All bits set to 1	N/A
2..1	ENTER_BOOT	00 = Bypass Bootloader Mode Done (Reserved for internal bootloader use) 01 = Reserved 10 = Enter Bootloader Mode 11 = Bypass Bootloader Mode Init (Default after a flash erase)	This field is used internally by the bootloader to determine whether bootloader mode will be bypassed upon startup. This field can be used by the application to indicate that the device should start in bootloader mode on reset, by setting it to value 0x10. The bootloader will automatically set this field to value 0x00 once it has run – if running in bootloader mode is desired again, the application needs to explicitly set this field to 0x10 again. **Important:** Applications should not set this field to any value other than 0x10.
0	PARAM_VALID	0 = Valid 1 = Invalid	This flag is used to indicate that the memory block reserved for the ant_boot_settings is populated with valid values and can be interpreted by the bootloader.

The ant_boot_settings_api is provided to facilitate the implementation of interaction between the bootloader and application. Refer to the ant_ota_tester project for an example of usage of the ant_boot_settings_api.

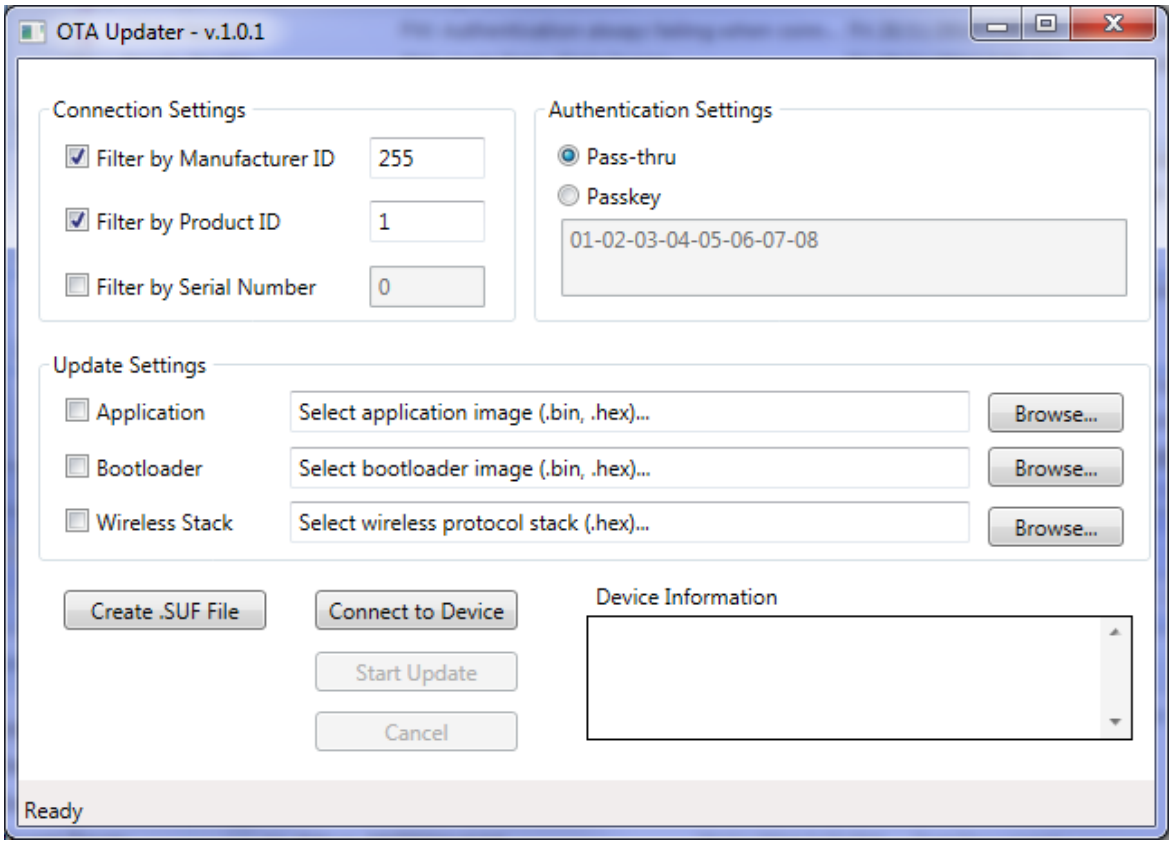
5.6.7 Usage

1. Edit antfs.c to make sure that ANTFS_NETWORK_KEY is set to the ANT-FS network key.
2. Insert a CR2032 coin cell battery into the N5 development board. In order to ensure adequate power is supplied during flash operations, use a new coin cell battery or power the module using a fixed 3.0 V power supply.
3. Load the reference code for the **ant_device_firmware_update/bootloader** project onto the module, as described on Section 4. Ignore the missing devices alert message.



4. If no application is present, the bootloader will run automatically. If a valid application is present on the device, the application will run; to manually start the bootloader, press button D and toggle the reset button.
5. Connect an ANT USB stick to your PC.

6. Start the OTA Updater application. The status bar at the bottom should indicate "Ready".



7. Configure the Connection Settings. If filtering by Manufacturer ID/Product ID, these parameters should match exactly those configured in the bootloader.

8. Configure the Authentication Settings. The bootloader uses pass-thru authentication by default. Passkey authentication is not supported in this version.

9. Browse for the image to update, and make sure the corresponding checkbox is checked. Both binary and hex formats are supported. Note that the OTA Updater application does not validate whether the selected image corresponds to an application, bootloader or wireless stack, so it is imperative to carefully select the type of image, otherwise, the update process will not work correctly.

10. To test the process, you can build the **ant_device_firmware_update/ant_ota_tester** project (without loading it onto the module) and use the resulting hex file as the application image to update. Ignore the missing devices alert message.

11. Click on "Connect to Device". Once the bootloader is discovered, the Device Information panel will show information about the current device, e.g., manufacturer ID, product ID, serial number and versions of images currently installed on it.

12. Click on "Start Update". The update process will begin.

13. When the update is complete, the status bar will indicate "Image upload complete. Image will be activated, do not power down device".

Optional steps if using the ant_ota_tester application to test the update process:

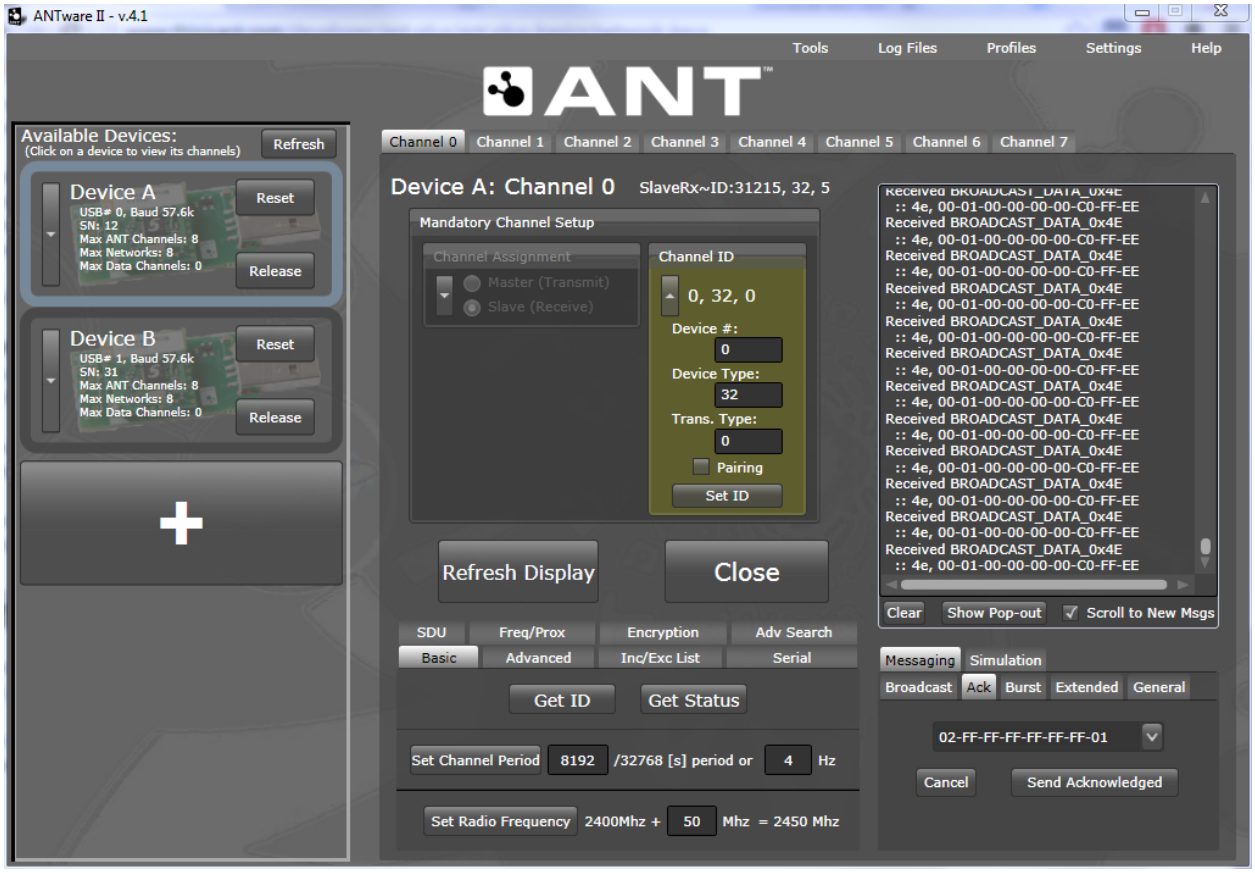
1. Close the OTA Updater application.

2. Make sure an ANT USB stick is connected to the PC
3. Start ANTware
4. Configure Channel 0 as a slave
5. Configure the rest of the channel parameters as follows:

Table 5-16. OTA Tester Channel Parameters

Parameter	Value
Network Key	Public
Device Number	0 (wild card)
Device Type	32
Transmission Type	0 (wild card)
Radio Frequency	2450 MHz
Channel Period	8192 (4Hz)

6. Click Auto-Open
7. The USB stick should start receiving broadcast data as in the following screenshot.



8. To start the bootloader again, select the "Ack" tab, and then set the payload to 02-FF-FF-FF-FF-FF-FF-01
9. Click "Send Acknowledged".
10. The device will start bootloader mode, and ANTware will lose communication with it.
11. Once in bootloader mode, you can repeat steps 6-13 to perform another update. Make sure to close ANTware to make the ANT USB stick available to the OTA Updater tool. If no update session is initiated within two minutes of starting the bootloader, and an application is present on the device, the application will run.

5.7 ANT Debug Demo

The purpose of the ANT Debug Demo is to show how to utilize debug messages. Debug messages are specially formatted messages usually sent over a separate ANT channel which convey information about the device which may be useful in debugging. Debug messages are received by an application such as ObservANT which can decode, display and log the information. Refer to the ANT Debug reference code and the "ObservANT User's Manual" for more information about using debug messages.

This demo uses:

- 1 N5 Development Board to run the reference code
- 1 ANT USB stick connected to the PC application ObservANT to view the debug messages
- A second ANT USB stick connected to the PC application ANTware, or a second N5 Development Board running the IO Rx Demo

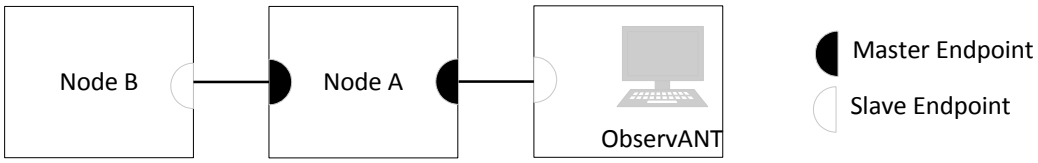


Figure 5-12. ANT Debug Demo Topology

The ANT Debug Demo is essentially just the IO Tx Demo with an additional channel for debug messages.

5.7.1 Usage

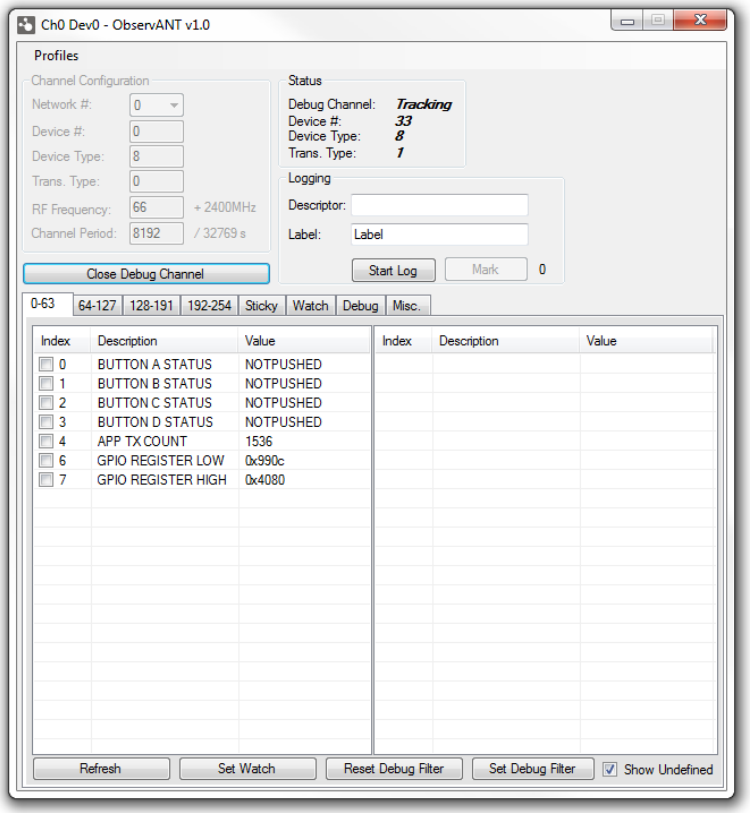
1. Insert a CR2032 coin cell battery into an N5 development board (Node A).
2. Load the reference code for the **ant_debug** project onto the module, as described on Section 4.
3. Connect an ANT USB stick into a USB port on your PC.
4. Copy the config.txt file included in the ant_debug project directory to the same directory where ObservANT is installed. This file is used by ObservANT to format the received debug fields.
5. Open the PC application ObservANT
6. Open a channel on ObservANT using the following Channel ID:

Table 5-17. Debug Demo Channel Parameters

Parameter	Value
Device Number	0 (wild card)
Device Type	8
Transmission Type	1

Use default values for all the other channel configuration parameters.

7. Observe the debug messages being received by ObservANT. Pressing buttons on the IO board will have an effect on the debug values.



8. Try setting a filter using ObservANT to change which debug fields are transmitted. This can be done by checking the boxes next to desired fields and clicking the Set Filter button.

9. Try sending a custom debug command by sending an acknowledged message from ObservANT to the device. Use the "Send Acknowledged" box under the "Ack" tab. Sending any message with the first byte set to 0x42 will signal the device to turn on all the LEDs and any other message will turn them all off.

10. Optional: Set up a second N5 Development Board with the Rx IO demo (Node B) or use ANTware as described in the IO Demo section to interact with the application channel.

5.8 ANT Scalable Channel Demo

The ANT Scalable Channels demo demonstrates how to set up and enable multiple ANT channels using the s210 v5.0.0 SoftDevice. Refer to the release notes for the s210 v5.0.0 and s310 v3.0.0 SoftDevices for more details. The ANT scalable channel application will open 15 channels by default.

The demo can be used with two N5 development boards (Figure 5-13) or one N5 development board and an N5 module, loaded with the ANT Network Processor firmware in the N5 SDK 3.0, mounted on an USB Interface board attached to a PC running ANTwareII (Figure 5-14).

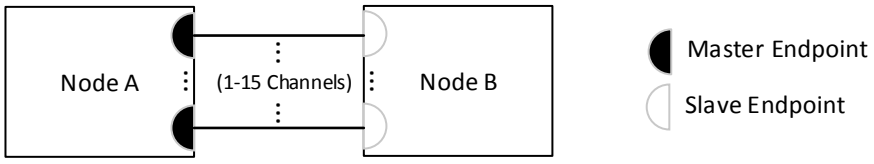


Figure 5-13. ANT Scalable Demo Topology using two N5 development boards

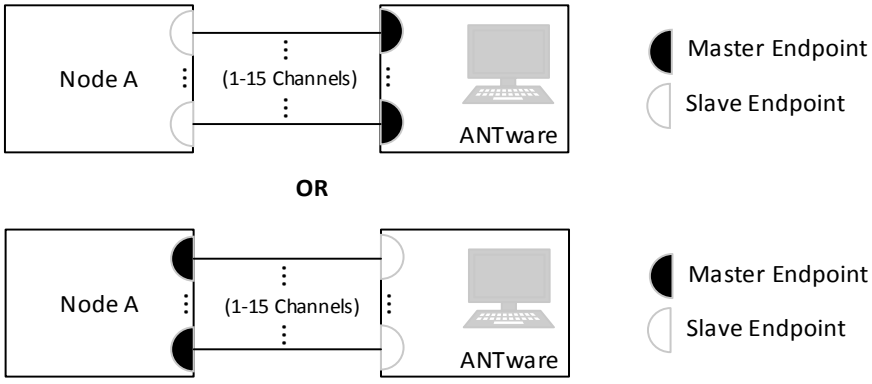


Figure 5-14. ANT Scalable Demo Topology using an ANT USB stick

On the transmitting (master) device, the pattern of LEDs indicates the number of open transmitting broadcast channels. On the receiving (slave) device, the pattern of LEDs will indicate the number of tracking channels. For instance, if 4 channels are tracking, the LED pattern will be 0100 (see photo below).

5.8.1 Usage with two N5 development boards

1. Insert a CR2032 coin cell battery into an N5 development board (Node A).
2. Load the reference code for the **ant_scalable\tx** project onto the module, as described on Section 4.
3. Insert a CR2032 coin cell battery into a second N5 development board (Node B).
4. Load the reference code for the **ant_scalable\rx** project onto the module, as described on Section 4.
5. On the transmitting device (Node A), the pattern of LEDs indicates the number of open transmitting broadcast channels.
6. On the receiving device (Node B), the pattern of LEDs will indicate the number of tracking channels.

5.8.2 Usage with ANTwareII

Note: To track up to 15 channels on ANTwareII, you must use an N5 module, loaded with the ANT Network Processor firmware from the N5 SDK 3.0, mounted on an USB Interface board, see Updating the Network Processor on the N5 module

1. Insert a CR2032 coin cell battery into an N5 development board (Node A).
2. Load the reference code for the **ant_scalable\rx** project onto the module, as described on Section 4.
3. Open the ANTware tool.
4. Configure Channels 0-14 as a Master.
5. Set the Channel ID of each channel to the following values:

Table 5-18. Scalable Demo Channel ID

Parameter	Value
Device Number	Channel number + 1
Device Type	2
Transmission Type	1

6. Open each channel with default values for all the other channel configuration parameters to begin transmission.
7. On the receiving device (Node A), the pattern of LEDs will indicate the number of tracking channels.

Note: This example can be run with Node A as the master, and ANTware as the slave. This is done by following the steps above, with these modifications:

Step 2: Load the reference code for the **ant_scalable\tx** project onto the module, as described on Section 4.

Step 4: Configure Channels 0-14 as a Slave.

Step 6: Open each channel with default values for all the other channel configuration parameters to begin searching.

Step 7: On the transmitting device (Node A), the pattern of LEDs will indicate the number of open transmitting channels.

Step 8: Once the two modes have paired, the received messages can be viewed on each of the 15 channels in ANTware.

5.9 ANT Scalable Encrypted Channel Demo

The ANT Scalable Encrypted Channels demo demonstrates how to set up and use up to 15 encrypted ANT channels on an the N5 development board.

The demo can be used with two N5 development boards (Figure 5-15) or one N5 development board and an N5 module, loaded with the ANT Network Processor firmware in the N5 SDK 3.0, mounted on an USB Interface board attached to a PC running ANTwareII (Figure 5-16).

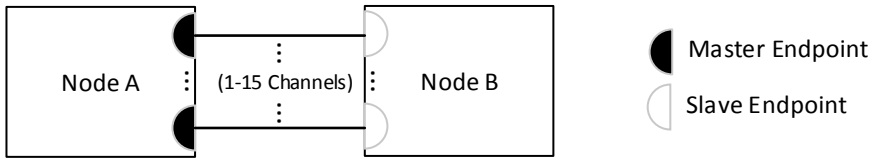


Figure 5-15. ANT Scalable Encrypted Demo Topology using two N5 development boards

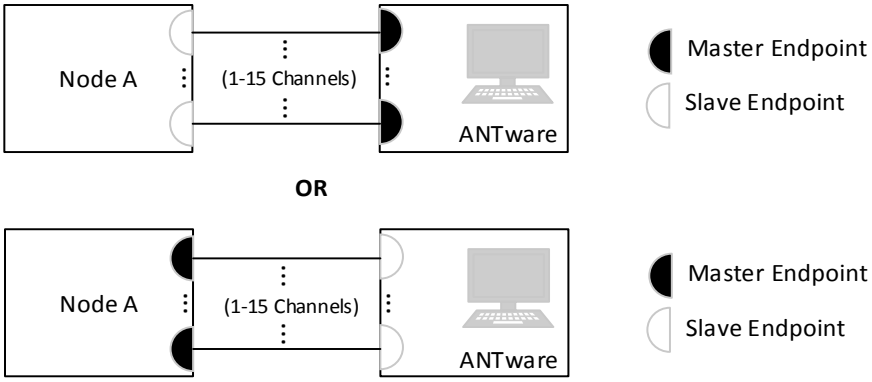


Figure 5-16. ANT Scalable Encrypted Demo Topology using an ANT USB stick

On the transmitting (master) device, the pattern of LEDs indicates the number of open transmitting encrypted channels. On the receiving (slave) device, the pattern of LEDs will indicate the number of channels which have been decrypted. For instance, if 4 channels are tracking, the LED pattern will be 0100 (see photo below).

5.9.1 Usage with two N5 development boards

1. Insert a CR2032 coin cell battery into an N5 development board (Node A)
2. Load the reference code for the **ant_scalable_encrypted\tx** project onto the module, as described in Section 4
3. Insert a CR2032 coin cell battery into a second N5 development board (Node B)
4. Load the reference code for the **ant_scalable_encrypted\rx** project onto the module, as described in Section 4
5. On the transmitting device (Node A), the pattern of LEDs indicates the number of open transmitting encrypted channels.
6. On the receiving device (Node B), the pattern of LEDs will indicate the number of channels which have been decrypted.

5.9.2 Usage with ANTwareII

Note: To track up to 15 channels on ANTwareII, you must use an N5 module, loaded with the ANT Network Processor firmware from the N5 SDK v3.0, mounted on an USB Interface board, see Updating the Network Processor on the N5 module

1. Insert a CR2032 coin cell battery into an N5 development board (Node A).
2. Load the reference code for the **ant_scalable_encrypted\tx** project onto the module, as described in Section 4.
3. Open the ANTware tool.
4. Open the device panel.
5. Enable advanced burst.

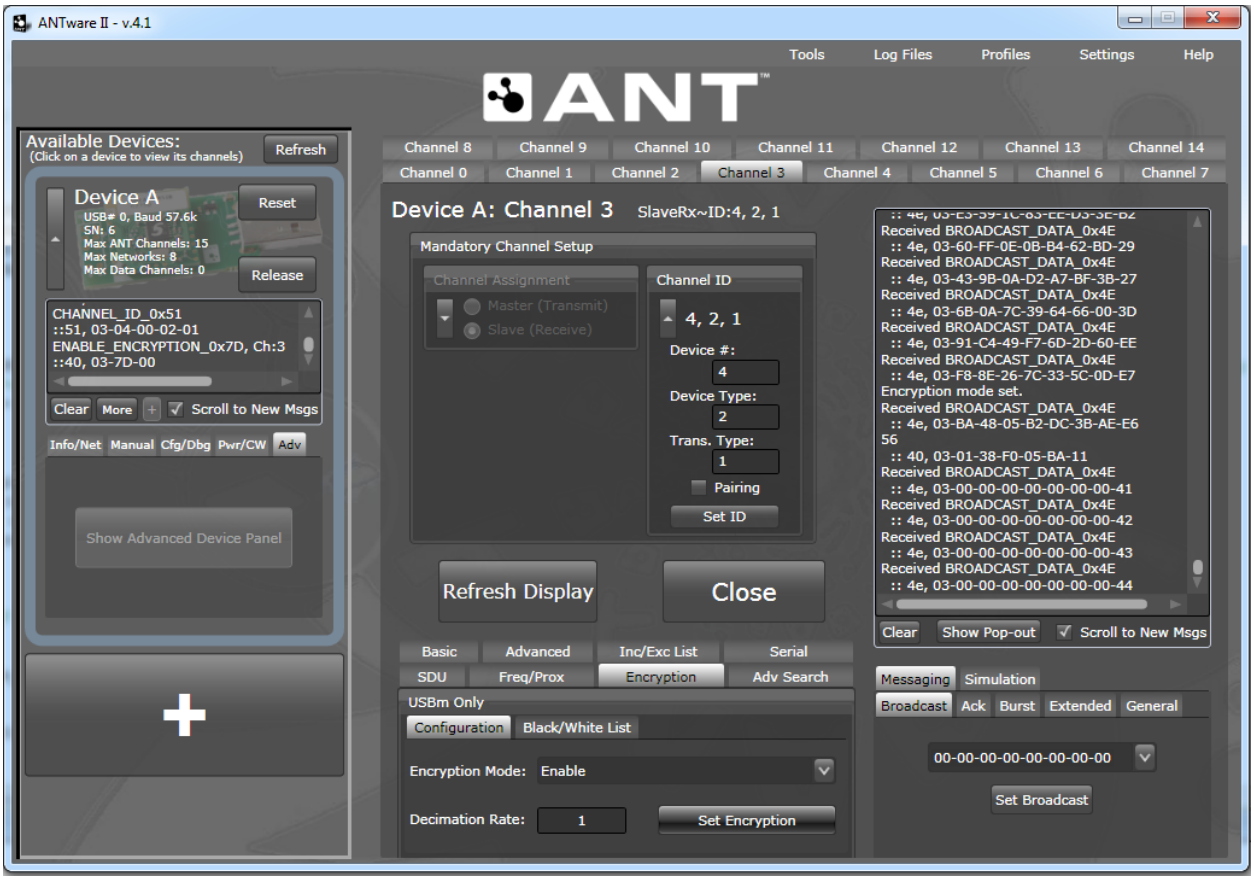
6. Set the Encryption Key to 03-01-04-01-05-09-02-06-05-03-05-08-09-07-09-03.

7. Configure Channels 0-14 as a Slave.
8. Set the Channel ID of each channel to the following values:

Table 5-19. Scalable Encrypted Demo Channel ID

Parameter	Value
Device Number	Channel number + 1
Device Type	2
Transmission Type	1

9. Open each channel with default values for all the other channel configuration parameters to begin transmission.
10. Notice how each channel is receiving garbled broadcast data.
11. While the channel is open, enable the encryption for each of the channels.



12. Notice how after encryption was enabled on each channel, the broadcast messages are decrypted, with the last byte of the payload increasing in every message.

5.10 ANT Scan and Forward Demo

The Scan and Forward Demo demonstrates how to construct a multi hop adhoc network where messages can be relayed across the network to extend the range. This demo uses a background scanning channel and an ANT master channel to listen for and repeat a sequence of messages. Each time a node sends an update, it is received and repeated by other nodes in range, allowing the message to reach nodes placed at the far end of the network.

The demo can be used with two or more N5 development boards (Figure 5-17)

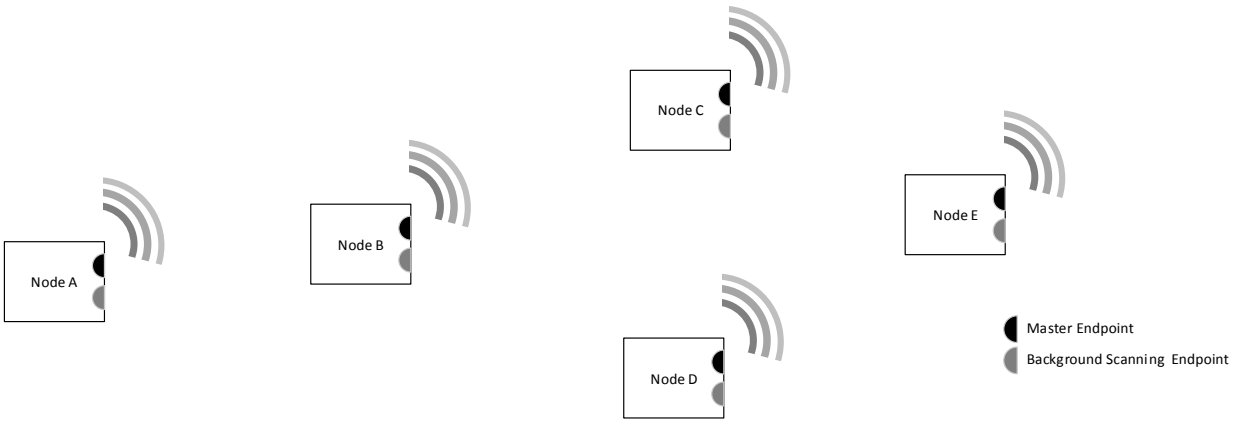


Figure 5-17. ANT Scan and Forward Demo Topology

5.10.1 Usage

1. Insert a CR2032 coin cell battery into X number of nodes N5 development boards (Node A – Node X)
2. Load the reference code for the **ant_scan_and_forward** project onto the modules, as described in Section 4.
3. Scatter the nodes in the desired placement, making sure that each node is within the range of at least one other node.
4. Press the buttons on the any of the nodes to interact with the network

Table 5-20. Function of Scan and Forward Buttons

Button	Command
A	Current Node – LED on
B	Current Node – LED off
C	All Nodes – LED on
D	All Nodes – LED off

5.11 ANT Asynchronous Controller using Continuous Scanning Mode Demo

The Asynchronous Controller using Continuous Scanning Mode Demo demonstrates how to use continuous scanning to asynchronously send a message to the nearest device. In Figure 5-18 the controller (Node A) is in range to receive messages from nodes C, D and E. The controller uses continuous scanning mode to identify the node with the strongest RSSI and sends back a command message to that node in the reverse direction. For demonstration purposes, the asynchronous controller demo is intended to be used in conjunction with the Scan and Forward Demo from Section 5.10, but can be adapted to other control applications.

The demo can be used with three or more N5 development boards or one N5 development board and one ANT USB stick attached to a PC running ANTwareII.

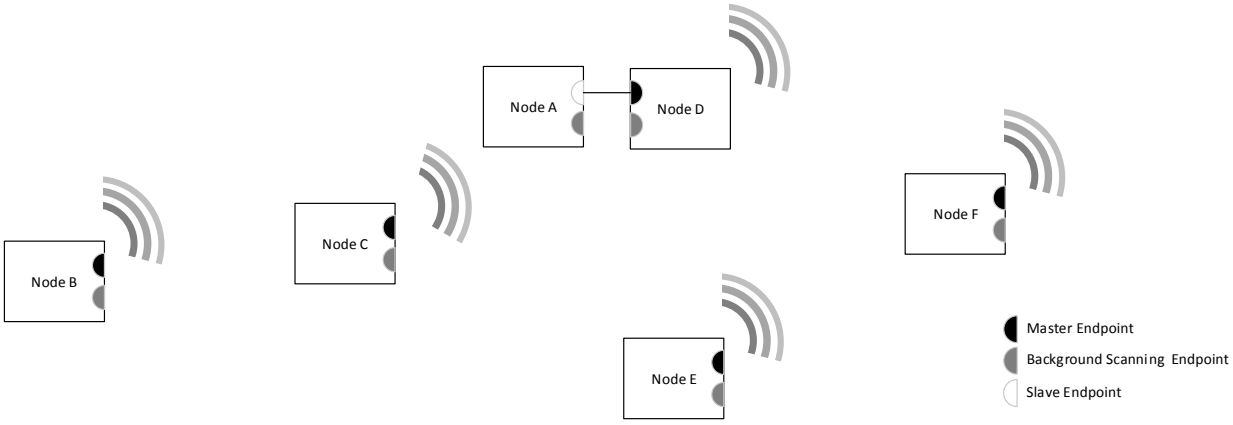


Figure 5-18. ANT Asynchronous Controller Demo Topology using N5 development boards

Table 5-21. Function of LEDs on Controller

Hardware	Function
LED A	Turns on when continuous scanning is active
LED B	Toggles when DEVICE_STATUS_PAGE is received on continuous scanning channel
LED C	Turns on if no Nodes were found sending a DEVICE_STATUS_PAGE after continuous scanning

5.11.1 Usage with Scan and Forward Demo

1. Insert a CR2032 coin cell battery into an N5 development board (Node A)
2. Load the reference code for the **ant_async_controller** project onto the module, as described in Section 4
3. Insert a CR2032 coin cell battery into X number N5 development boards (Node B – Node X)
4. Load the reference code for the **ant_scan_and_forward** project onto the modules, as described in Section 4
5. Separate the nodes in a configuration of your choice. Distances greater than 2 meters provides the best experience.

6. On the controller device (Node A), when a button is pressed, the corresponding command is send into the network of nodes

Table 5-22. Function of Asynchronous Controller Buttons

Button	Command
A	Closest Node – LED on
B	Closest Node – LED off
C	All Nodes – LED on
D	All Nodes – LED off

7. On the network nodes (Node A – Node X), the LEDs will turn on or off based on the command received from the controller

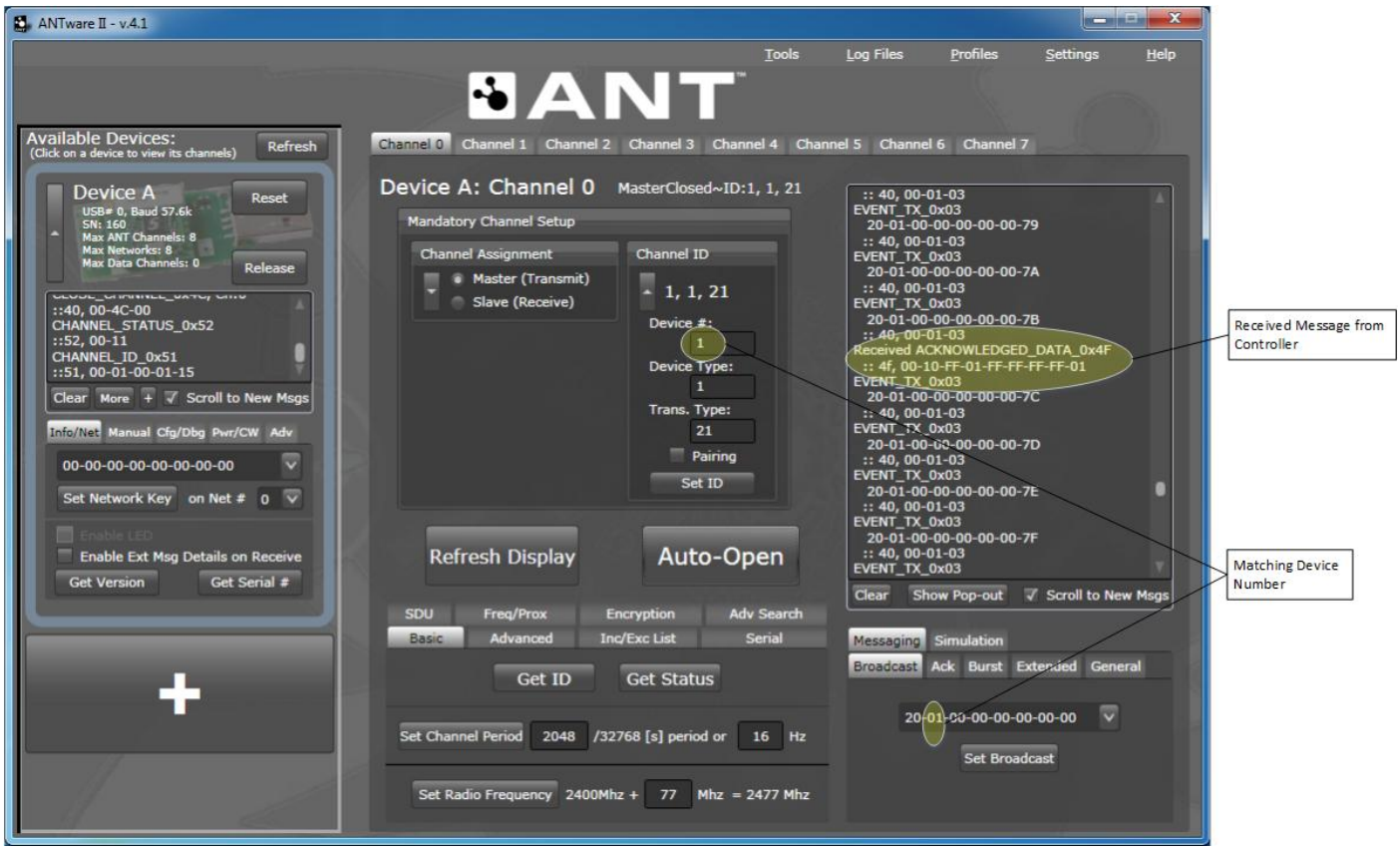
5.11.2 Usage with ANTwareII

1. Insert a CR2032 coin cell battery into an N5 development board (Node A).
2. Load the reference code for the **ant_async_controller** project onto the module, as described in Section 4.
3. Insert one ANT USB stick into your computer.
4. Open the ANTware tool.
5. Configure Channel 0 as a master.
6. Set the Channel Parameters to the following values.

Table 5-23. Asynchronous Controller Demo Channel Parameters

Parameter	Value
Device Number	1
Device Type	1
Transmission Type	21
Channel Period	16Hz (2048)
Radio Frequency	2477MHz

7. Set the broadcast message to 20-01-00-00-00-00-00-00 (The second byte of the broadcast message matches the device number)
8. Open the channel.
9. Pressing a button on the Controller (Node A) will send a command which can be seen in the ANTware message window as an acknowledged message.


ANTware II - v.4.1
Tools
Log Files
Profiles
Settings
Help
ANT
Available Devices:
(Click on a device to view its channels)
Refresh
Device A
USB# 0, Baud 57.6k
SN: 160
Max ANT Channels: 8
Max Networks: 8
Max Data Channels: 0
Reset
Release
::40, 00-4C-00
CHANNEL_STATUS_0x52
::52, 00-11
CHANNEL_ID_0x51
::51, 00-01-00-01-15
Clear
More
Scroll to New Msgs
Info/Net
Manual
Cfg/Dbg
Pwr/CW
Adv
00-00-00-00-00-00-00-00
Set Network Key
on Net #
0
Enable LED
Enable Ext Msg Details on Receive
Get Version
Get Serial #
Channel 0
Channel 1
Channel 2
Channel 3
Channel 4
Channel 5
Channel 6
Channel 7
Device A: Channel 0
MasterClosed~ID:1, 1, 21
Mandatory Channel Setup
Channel Assignment
Master (Transmit)
Slave (Receive)
Channel ID
1, 1, 21
Device #:
1
Device Type:
1
Trans. Type:
21
Pairing
Set ID
Refresh Display
Auto-Open
SDU
Freq/Prox
Encryption
Adv Search
Basic
Advanced
Inc/Exc List
Serial
Get ID
Get Status
Set Channel Period
2048
/32768 [s] period or
16
Hz
Set Radio Frequency
2400Mhz +
77
Mhz = 2477 Mhz
:: 40, 00-01-03
EVENT_TX_0x03
20-01-00-00-00-00-00-79
:: 40, 00-01-03
EVENT_TX_0x03
20-01-00-00-00-00-00-7A
:: 40, 00-01-03
EVENT_TX_0x03
20-01-00-00-00-00-00-7B
:: 40, 00-01-03
Received ACKNOWLEDGED_DATA_0x4F
:: 4f, 00-10-FF-01-FF-FF-FF-FF-01
EVENT_TX_0x03
20-01-00-00-00-00-00-7C
:: 40, 00-01-03
EVENT_TX_0x03
20-01-00-00-00-00-00-7D
:: 40, 00-01-03
EVENT_TX_0x03
20-01-00-00-00-00-00-7E
:: 40, 00-01-03
EVENT_TX_0x03
20-01-00-00-00-00-00-7F
:: 40, 00-01-03
EVENT_TX_0x03
Clear
Show Pop-out
Scroll to New Msgs
Messaging
Simulation
Broadcast
Ack
Burst
Extended
General
20-01-00-00-00-00-00-00
Set Broadcast
Received Message from Controller
Matching Device Number

6 Changes when switching from N5 Starter Kit SDK 2.0 to 3.0

6.1 Migrating projects from N5 Starter Kit SDK 2.0 to 3.0

The N5 Starter Kit SDK v3.0 supports the SoftDevice s210 v5.0.0 that allows for up to 15 encrypted channels. A new SoftDevice API function was introduced to enable the configuration of multiple ANT channels; please refer to the release notes for the SoftDevice s210 v.5.0.0 for more details on this feature. Projects built using the N5 Starter Kit SDK 2.0 and using more than one channel must be updated. Refer to Section 5.8 for an example on how to enable multiple channels.

6.2 Updating the Network Processor on the N5 module

Follow these steps to updatean N5 module to the required version of the network processor:

1. Remove the N5 module from the ANT USB interface board.
2. Program the N5 module with SoftDevice v5.0.0 as described in Section 3, however do not use the I/O board as described. Instead connect the N5 module directly to the battery board.
3. After completing the last step (step 14) of Section 3
4. Click on the "Program Application" tab
5. Click the "Browse…" button and navigate to:

   C:\Keil_v5\ARM\Device\Nordic\examples\n5_series\s210\ ant_network_processor_s210
6. Open the Network Processor hex file (eg. ant_network_processor_s210.hex)
7. Click the "Program" button in the "Program Application" tab to program the network processor application onto the N5 Module
8. Remove the N5 module from the battery board.
9. Stack the N5 module back onto the ANT USB interface board.

7 Running Nordic examples on the N5 Starter Kit

The Nordic nRF51 SDK provides a layer of abstraction from the board that easily allows the developer to write code that runs on multiple boards; eliminating the need to know about how the pins are assigned on a specific board. This layer of abstraction occurs within their Board Support Package (BSP). There are some limitations of portability as some development boards may offer different functionality such as more/less LEDs or buttons. If the example code is compatible to run on the N5 development board, the layer of abstraction allows us to easily modify an example from the Nordic SDK to run on the N5 development board.

7.1 Considerations when converting Nordic example to run on the N5 Starter Kit

Here are some general considerations that should be thought of before switching a Nordic example to run on the N5 starter kit. A detailed example of converting a Nordic project to run on the N5 Development Board is shown in Section 7.2.

- **Is the project compatible to run on the N5 development board?**

  Not all of the Nordic demo projects will run on the N5 development board. As an example, the N5 Starter Kit does not include the necessary peripherals to run the temperature example.

- **Selecting the correct project file**

  It is important to open the correct project file as Nordic has provided a variety of project configurations based on whether the nRF51 SDK was installed with/without Keil packs. When opening Keil projects developed by Nordic, do not open the µVision Multi-Project file located in the base directory of that specific example. Instead open the "pca10028" folder, and open the "arm5_no_packs" folder. Inside the "arm5_no_packs" folder you will find the correct µVision5 Project file.

- **Copy the Target Settings**

  Once the correct project has been opened, the first thing that should be done is to take note of ALL the Target information for the specific project. If the targeted device is changed, these settings will be reset and must be re-entered to have proper functionality of the example. (Step 6 & 9 in updating blinky example)

- **Updating the Targeted Device**

  Once the Target information has been recorded, the targeted device should be changed to "nRF51422_xxAA". The N5 modules included in the Starter Kit use the first revision of the nRF51422 silicon chip and to ensure proper functionality the project should be adjusted to target that version. (Step 7 & 8 in updating blinky example).

- **Updating the Target Information**

  After changing the Device you must reconfigure the Target settings to what they were before the Device was changed. (Step 9 in updating blinky example).

- **Updating the Flash Tool**

  Once changing the targeted device you will have to reconfigure the SEGGER J-Link programmer. (Step 13-19 in updating blinky example).

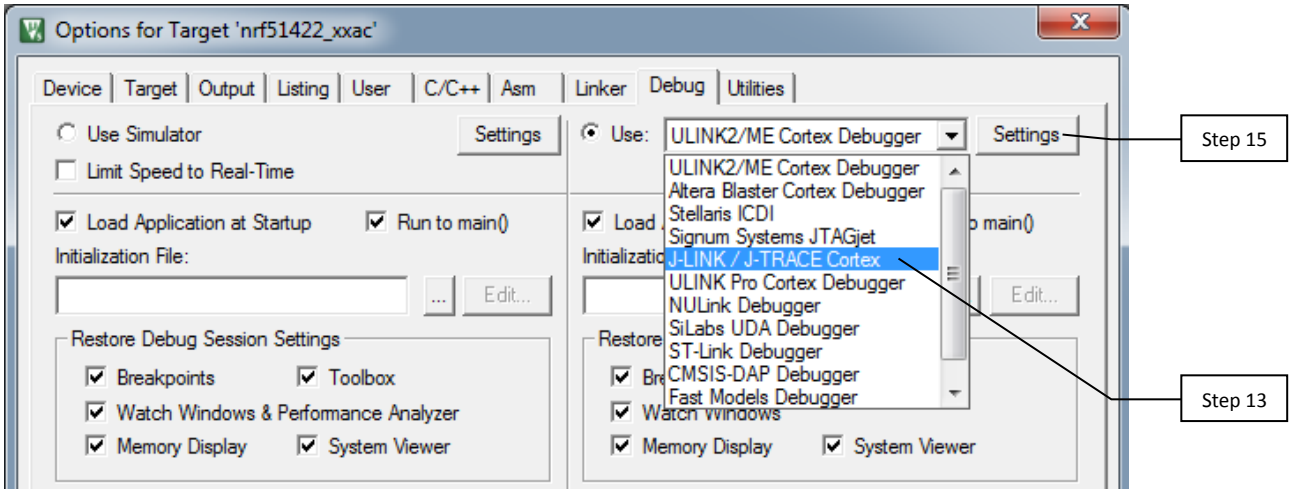
- **Ensure the correct SoftDevice is present**

  It is vital to ensure that the correct SoftDevice is programmed to the N5 module (using nRFgo studio in Section 3) before the application is flashed onto the N5 module. For example, all of the Reference Design Demos (Section 5) outlined in this document must have the SoftDevice s210 version 5.0.0 present to run properly. In Section 7.2 we were required to erase the SoftDevice off the N5 module to properly run the blinky example.

7.2 Updating the Nordic blinky example to run on the N5 Starter Kit:

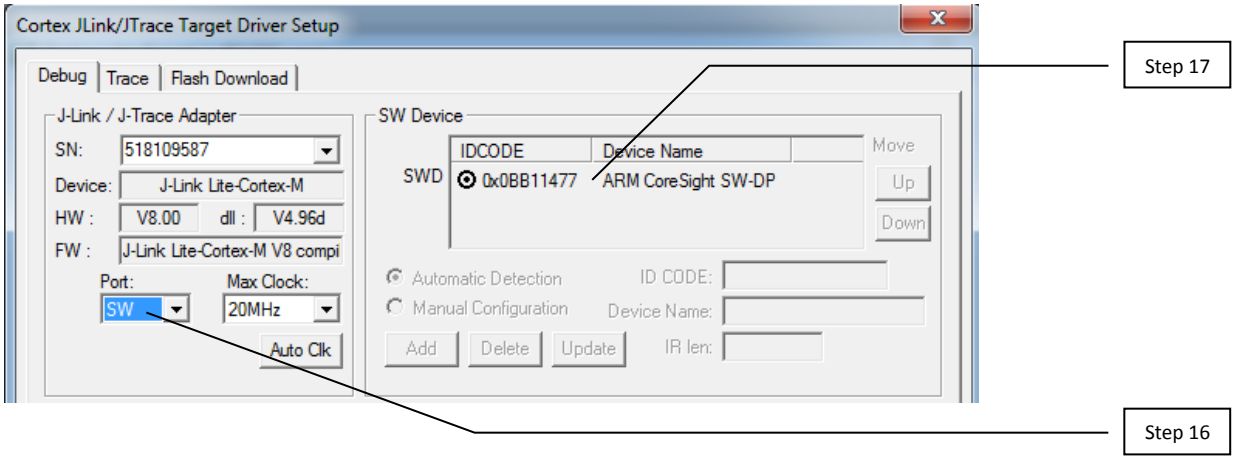
Note: Proceeding with these instructions will erase the SoftDevice from the N5 Development board. This will require you to reprogram the SoftDevice as described in Section 3 afterwards to properly run the Reference Design Demos (Section 5) outlined in this document.

1. Prepare the hardware and connect it to the computer as described in **steps 1-6** of Section 3.
2. Skip steps 7-11, and complete **step 12**.
3. Skip steps 13-14, and quit nRFgo studio.
4. Locate the project. If the default locations were used during installation you will need to double click on the following file: C:\Keil_v5\ARM\Device\Nordic\examples\peripheral\blinky\pca10028\blank\arm5_no_packs\blinky_blank_pca10028.uvprojx
5. Right click on the "nrf51422_xxac" folder in the "Project Window" and select "Options for target 'nrf51422_xxac"
6. Click on the "Target" tab and record ALL the settings on this window. You will need to re-enter these settings after they reset when the device is changed.
7. Click on the "Device" tab.
8. Select Nordic Semiconductor -> nRF51 Series -> **nRF51422_xxAA**.
9. Switch back to the "Target" Tab and re-enter the settings you recorded from the previous device.
10. Click on the "C/C++" tab
11. Change "BOARD_PCA10028" to "BOARD_N5DK1" in the "Define:" text box.
12. Click on the "Debug" tab.
13. Select "J-LINK / J-TRACE Cortex" from the dropdown menu.



14. The N5 Development Board should be powered on and connected to the SEGGER J-Link before completing Step 15.
15. Click on "Settings" located to the right side of the drop down menu.
16. In the "J-Link / J-Trace Adapter" section set "Port:" to SW.

17. Check that the SW device was detected.



18. Click on the "Flash Download" Tab.
19. Check to see if there is a Programming Algorithm.
    a. If there is no Programming Algorithm listed, click "Add"
    b. Select nRF51xxx
    c. Click "Add"
20. Click "OK".
21. Click the "Rebuild" icon.
22. Click the "Load" icon.

8 Appendix 1 – A Note from Segger

A SEGGER J-Link Lite Cortex-M is included in this kit. The J-Link Lite is a very small form factor debug probe which is software compatible to the widely acknowledged J-Link line. This device has a JTAG clock of up to 2 MHz. It supports SWD and -SWO. The J-Link Lite is only delivered and supported as part of a starter kit, which includes an evaluation board. It is not sold separately. It may only be used with the evaluation board it came with. The SEGGER public forum is available at: http://forum.segger.com/.

When you have completed your use of the J-Link Lite while working with this starter kit and are ready to move to a full debug probe and/or production flash programmer, we are confident you will find one of the following SEGGER solutions a perfect fit.

J-Link PRO

Debug Probe with USB and Ethernet interface
Includes all software enhancement modules

J-Link ULTRA+

Ultra Fast Debug Probe
Includes all software enhancement modules

J-Link PLUS

Debug Probe
Includes all software enhancement modules

J-Link BASE

Debug Probe

J-Link EDU

Educational Use Debug Probe

SEGGER also offers a full featured RTOS and middleware offering (File System, USB Stack, TCP/IP Stack, and Graphics Package).

Additional information may be found at: www.segger.com and: www.segger.com/debug-probes.html

Flasher ARM

Production Flash Programmer

Flasher Portable

Portable Flash Programmer