Using the DVK-BL652 and Nordic SDK v14.0.0 with Eclipse & GCC

BL652

Application Note *v1.0*

INTRODUCTION

This application note is intended to help developers who want to use BL652 module to develop applications using freely-available tools and the Nordic SDK. Step-by-step instructions follow for setting up Eclipse, the ARM GCC compiler and Nordic's nRF5x SDK. The ble_app_hrs example is used to demonstrate how to build, flash, and debug SDK applications on the BL652 development kit.

REQUIREMENTS

The following are required to complete this step-by-step guide: -

- Windows 7 machine
- DVK-BL652 Development kit
- USB A to micro USB cable (provided with DVK-BL652)

Note: The following sections contain a step-by-step guide on obtaining and installing the other requirements listed below. However, please note that the examples were completed and tested using the following:

- Nordic SDK v14.0.0
- nRFX-Command-Line-Tools v9.7.1
- Segger J-Link V6.20b
- Eclipse IDE for C/C++ Developers – Version: Juno Service Release 2 – Build: 20130225-0426
- GNU ARM Toolchain v5.4.1
- GNU Make Utility v3.81
- CoreUtils for Windows

Although unlikely, any deviation from the above setup can cause issues in the build. For more information regarding this, you can visit the Nordic tutorials on using SDK examples at:

https://devzone.nordicsemi.com/tutorials/7/development-with-gcc-and-eclipse/

DEVELOPMENT ENVIRONMENT SETUP

Nordic SDK

The Nordic Software Development Kit offers a rich development environment and examples for the BL652 module. It can easily be used to develop applications for the BL652 when development using freely available

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

software and C language is preferred. The SDK offers a wide selection of drivers, libraries, and examples for the module and its peripherals.

To use Nordic's SDK for BL652 development, complete the following steps:

1. Download the Nordic SDK zip file from https://www.nordicsemi.com/eng/Products/Bluetooth-low-energy/nRF5-SDK. The file is located in the **DOWNLOADS** tab.
2. Once downloaded, extract and place the SDK in a suitable location on your machine, e.g.: D:\Work\nRF5_SDK (Figure 1).

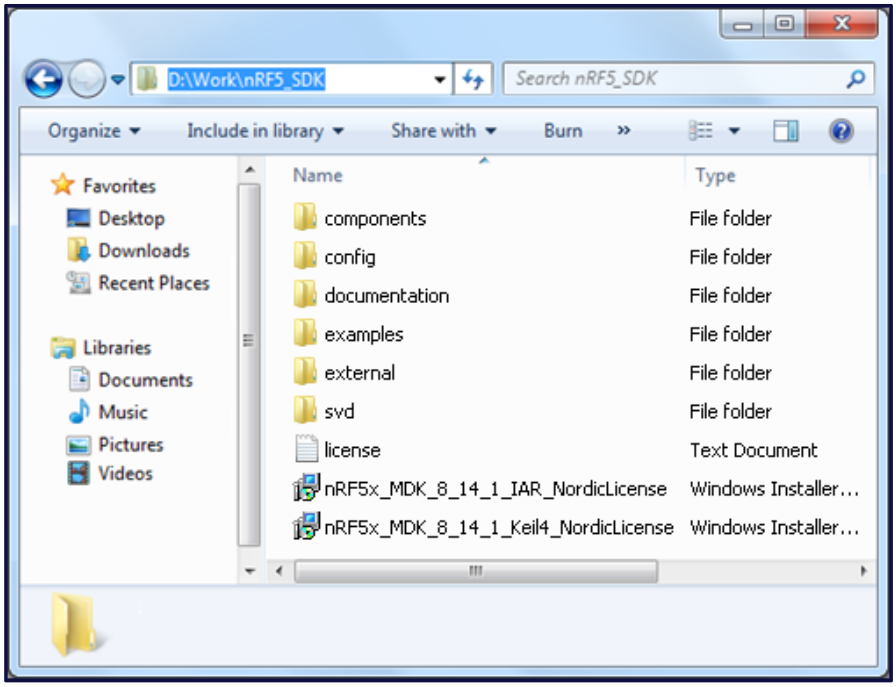

Figure 1: nRF5 SDK Folder Structure

By default, the BL652 uses the nRF52's on-chip RC oscillator as its clock source, as opposed to the nRF52 development kits, which use an external crystal. The difference in oscillator source should therefore be reflected in the SDK. Open and edit the nRF5_SDK\components\softdevice\common\nrf_sdh.c which is originally as follows:

```
168
169     nrf_clock_lf_cfg_t const clock_lf_cfg =
170     {
171         .source        = NRF_SDH_CLOCK_LF_SRC,
172         .rc_ctiv       = NRF_SDH_CLOCK_LF_RC_CTIV,
173         .rc_temp_ctiv  = NRF_SDH_CLOCK_LF_RC_TEMP_CTIV,
174     #ifdef S132
175         .accuracy      = NRF_SDH_CLOCK_LF_XTAL_ACCURACY
176     #else
177         .xtal_accuracy = NRF_SDH_CLOCK_LF_XTAL_ACCURACY
178     #endif
179     };
180
181     #ifdef ANT_LICENSE_KEY
182         ret_code = sd_softdevice_enable(&clock_lf_cfg, app_error_fault_handler, ANT_LICENSE_KEY);
183     #else
184         ret_code = sd_softdevice_enable(&clock_lf_cfg, app_error_fault_handler);
185     #endif
```

Change the oscillator so that it uses the BL652 configuration as follows:

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

```
168
169     nrf_clock_lf_cfg_t const clock_lf_cfg =
170     {
171         .source        = 0,
172         .rc_ctiv       = 16,
173         .rc_temp_ctiv  = 2,
174     #ifdef S132
175         .accuracy      = NRF_SDH_CLOCK_LF_XTAL_ACCURACY
176     #else
177         .xtal_accuracy = NRF_SDH_CLOCK_LF_XTAL_ACCURACY
178     #endif
179     };
180
181     #ifdef ANT_LICENSE_KEY
182         ret_code = sd_softdevice_enable(&clock_lf_cfg, app_error_fault_handler, ANT_LICENSE_KEY);
183     #else
184         ret_code = sd_softdevice_enable(&clock_lf_cfg, app_error_fault_handler);
185     #endif
```

nRFX-Command-Line-Tools

The nrfjprog is a command line tool used for erasing and flashing the BL652 with applications. The tool is bundled as part of the nRF5x-Command-Line-Tools. You can install and setup the command line tools as follows:

1. Navigate to https://www.nordicsemi.com/eng/Products/Bluetooth-low-energy/nRF52832
2. Select the **DOWNLOADS** tab.
3. Download **nRF5x-Command-Line-Tools-Win32**.
4. Once downloaded, launch the executable. Proceed through the setup.

Note: During the Setup process, you are prompted to install SEGGER J-Link. This is required for the command line tools to work, so proceed by accepting and installing it.

---

5. Once downloaded, the **nrfjprog** executable can be found at C:\Program Files (x86)\Nordic Semiconductor\nrf5x\bin (Figure 2).

Note: For 32-bit Windows installations, the location is:
C:\Program Files\Nordic Semiconductor\nrf5x\bin.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

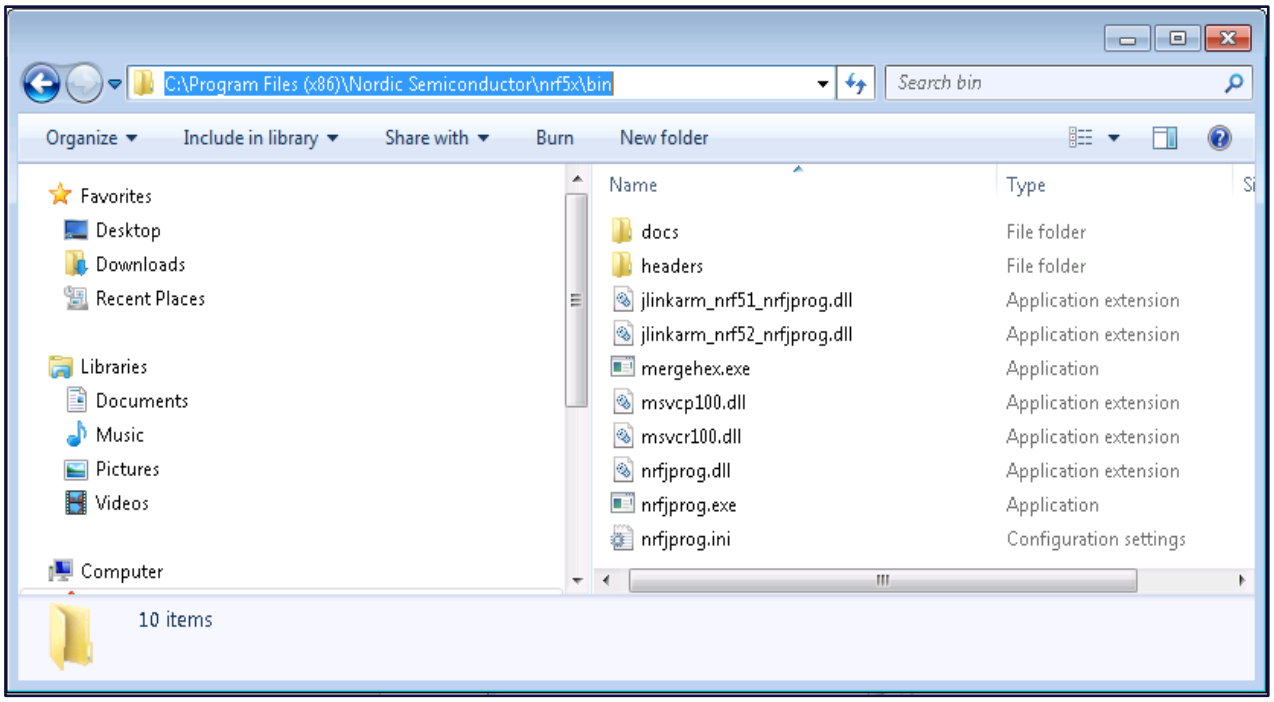

Figure 2: nrfjprog executable location

Eclipse

Eclipse is a universal, highly customizable IDE, C code browser, and an efficient editor. It is a free software based on Java. Please note that it is important to download the package that comes with the **CDT 'C/C++ development tools' plugin**. Usually no installation is usually required for Eclipse. Once downloaded (and unzipped), eclipse.exe can be launched straight away.

Eclipse Initial Setup

The following steps are required to setup Eclipse appropriately:

1. If not already installed, download and install the Java Runtime Environment from http://www.java.com

   Note: For 64-bit systems, please ensure that the bit-version of java matches that of eclipse, i.e. both should be 32-bit or 64-bit.

2. Download the Eclipse IDE for C/C++ from http://www.eclipse.org/downloads/packages/eclipse-ide-cc-developers/junosr2 (the download link should appear on the right-hand-side of the web page)

   Note: There are many different packages and types of Eclipse and each has its unique features and goes through a different installation process. While all packages should work similarly, the installation of the juno package is highly recommended, as it is used for the examples in this guide.

3. Extract the zip file into a folder in the same download location.
4. Move the folder to a known location, e.g.:
   D:\Work\Eclipse (Figure 3).

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

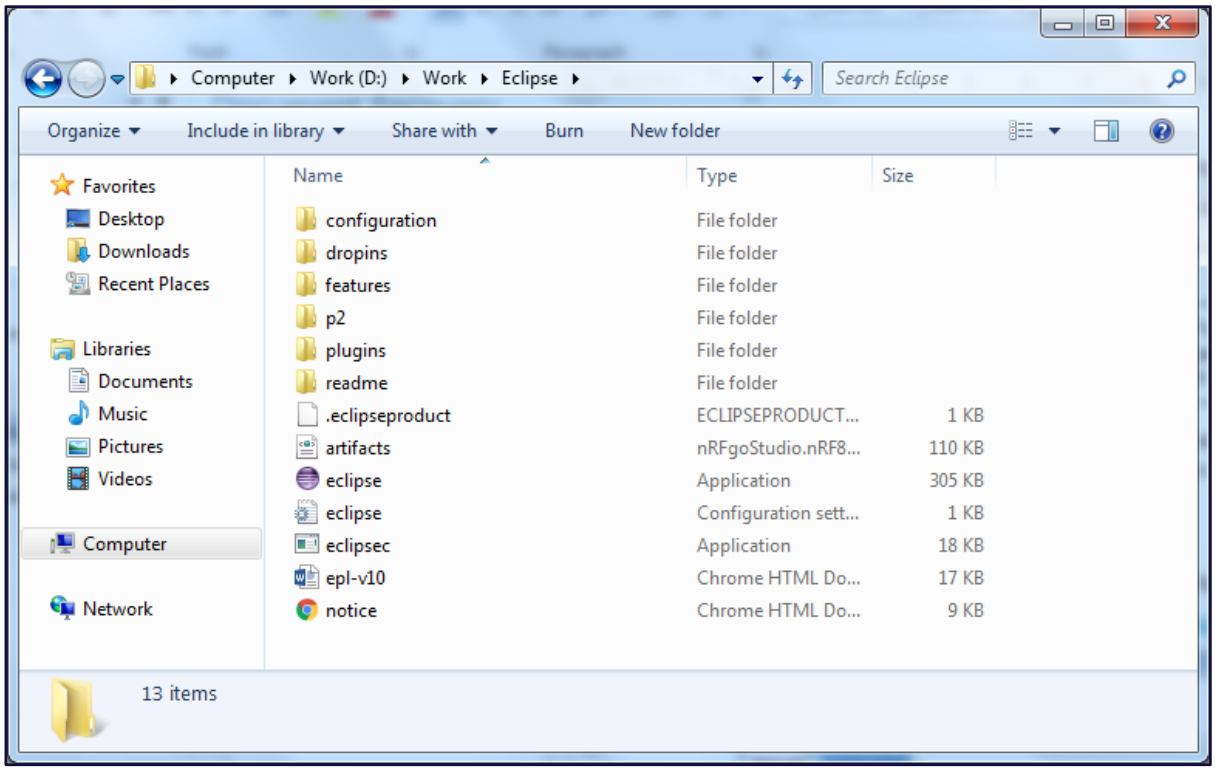

Figure 3: Eclipse executable location

5. Once moved, launch the Eclipse executable (Application). Use the default **Workspace** when prompted (Figure 4).

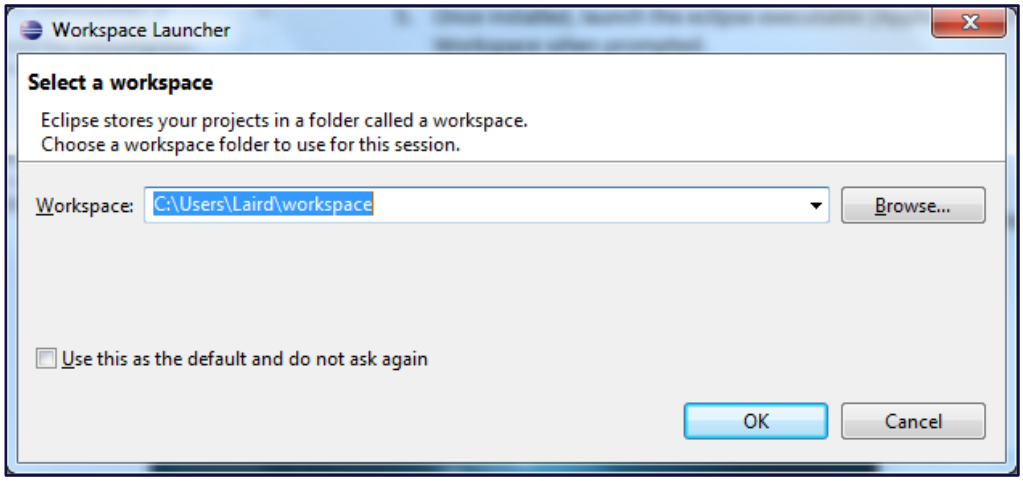

Figure 4: Eclipse default workspace

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

Eclipse Debugging Tools

To install the Eclipse hardware GDB tools, complete the following steps:

1. From Eclipse, click **Help** > **Install New Software** (Figure 5).

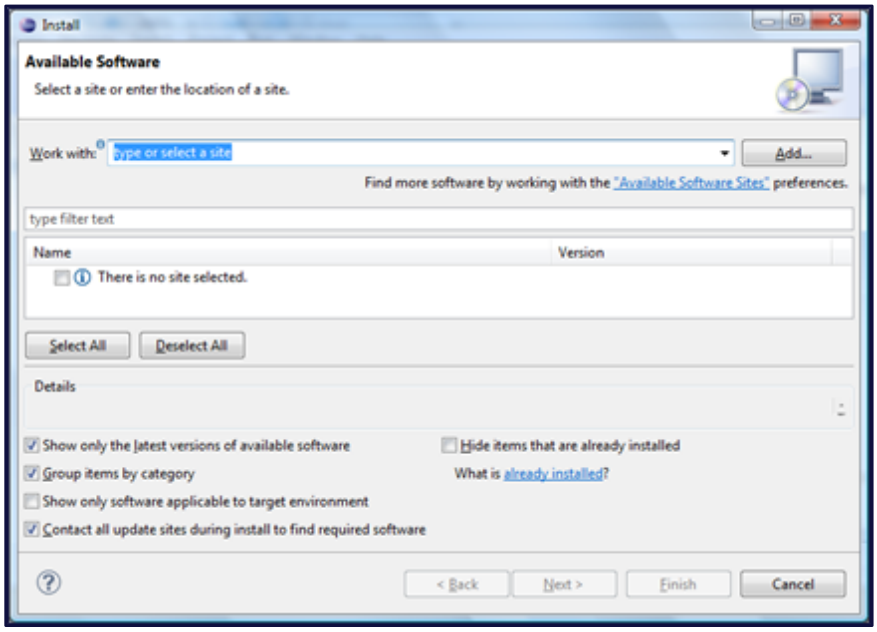

Figure 5: New Software installation window

2. Add the following URL to the repository list (Work with): http://download.eclipse.org/tools/cdt/releases/juno. The main items window will populate with new software packages from this repository.
3. Check CDT Main Features and GDB Hardware Debugging (Figure 6).

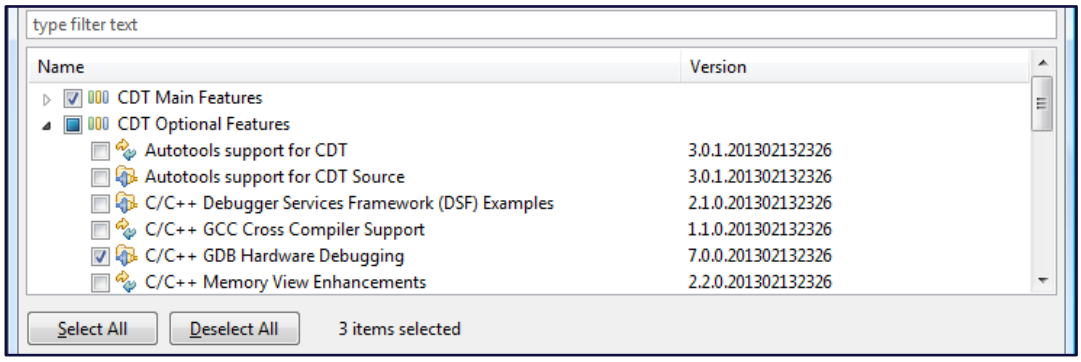

Figure 6: New software selected

4. Click **Next**.
5. Proceed through the installation by clicking **Next** in the windows that follow (Figure 7).

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

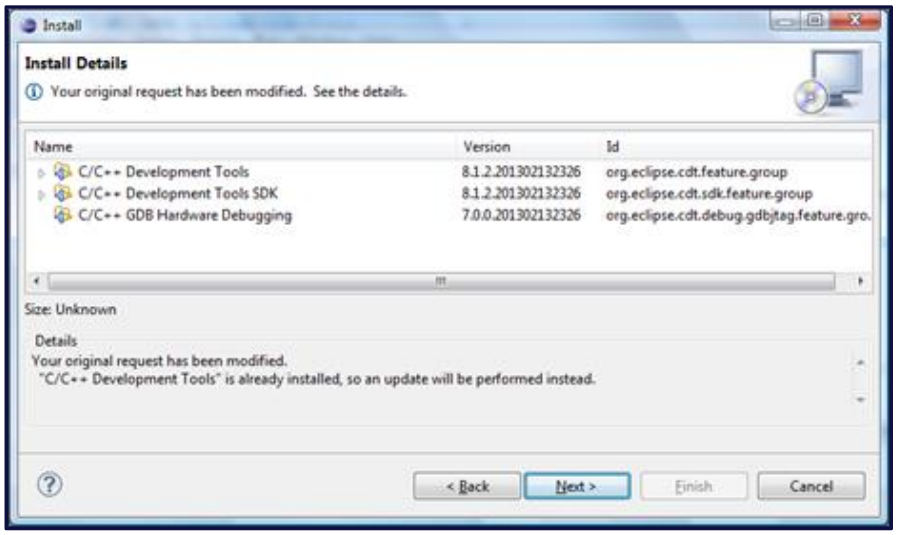

Figure 7: Completing installation

6. Once the installation is complete, you might need to restart Eclipse for the changes to take effect.

Eclipse Embedded System Register View

To install the embedded system register view, complete the following steps:

1. From Eclipse, click **Help > Install New Software**.
2. Add the following URL to the repository list (Work with): http://embsysregview.sourceforge.net/update. The main items window will populate with new software packages from this repository.

Note: If embsysregview does not appear in the window, your firewall may be preventing you from accessing the download page. Please check your firewall settings.

---

3. Select and install ***embsysregview*** (Figure 8).

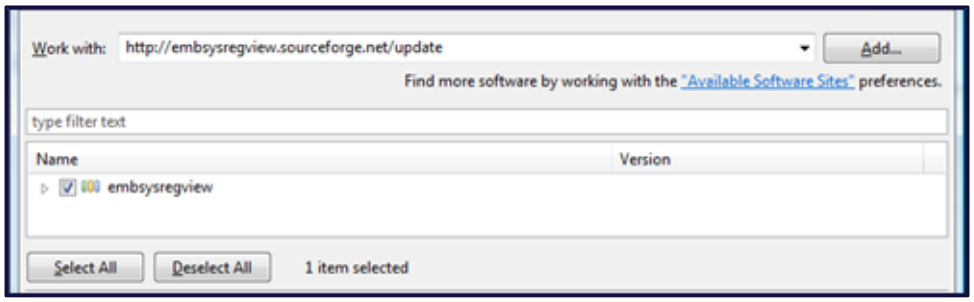

Figure 8: Installing embsysregview

4. Click **Next**.
5. Proceed through the installation.
6. Once the installation is complete, you might need to restart Eclipse for the changes to take effect.
7. To install the register view file for Nordic Semiconductor devices, copy file nrf52.svd as follows:
   - From: **/nRF5_SDK/svd**
   - To: **/eclipse/plugins/org.eclipse.cdt.embsysregview_<version>/data/SVD(CMSIS)/Nordic**

   You may need to create the Nordic folder if it does not already exist.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

GNU ARM Toolchain Setup

The GNU ARM toolchain contains tools for building and debugging the project, such as GCC and GDB. To setup the toolchain, complete the following steps:

1. Download the *Windows installer* file from the following link: https://launchpad.net/gcc-arm-embedded/+download.
2. Launch the executable.
3. Select the Destination folder to be a known location, e.g.: **D:\Work\Tools\ARM_GNU**

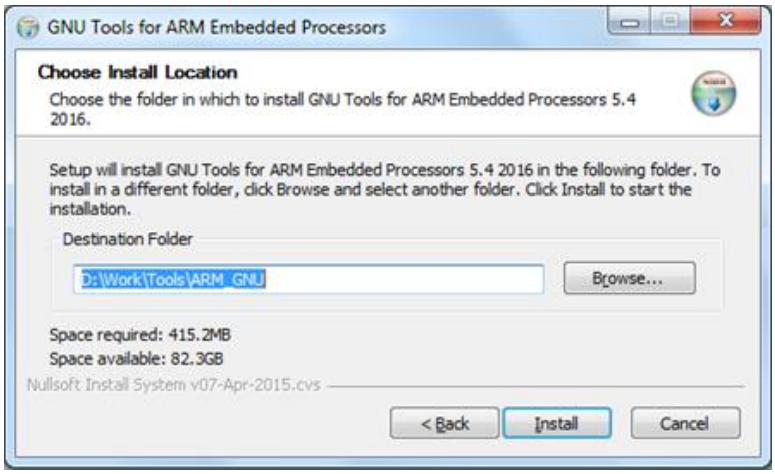

Figure 9: Installing GNU Tools for ARM Embedded Processors

4. Select **Install**.

To use the toolchain executables from any directory, their location needs to be added to the PATH environment variable. Add the executables to the PATH environment variable by doing the following:

1. Right click on **Computer** > **Properties**.
2. Select Advanced system settings.
3. Select **Environment Variables.**
4. In the **System variables** section, select **Path** and click **Edit**….
5. Add the location of the GNU ARM executables preceded by a semicolon (;), e.g.: **D:\Work\Tools\ARM_GNU\bin**
6. Open the command prompt (if already open cmd must be reopened) and confirm that the ARM toolchain is set up correctly by typing:

```
arm-none-eabi-gcc -version
```

The above command should return the version number of ARM GCC, if it doesn't then this means that the ARM toolchain location was not added to the PATH variable successfully (Figure 10).

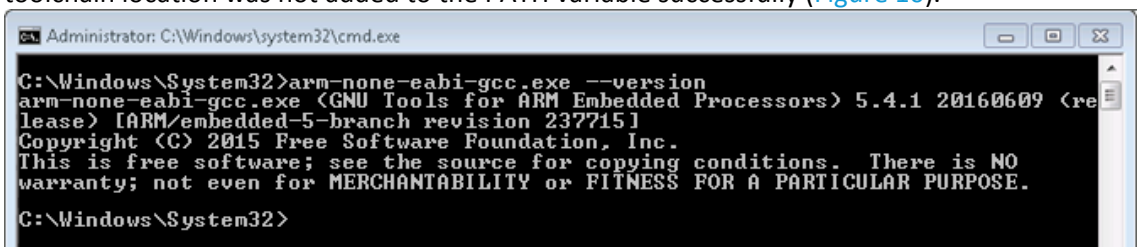

Figure 10: Using ARM GCC through the command prompt

The location and version of the ARM GNU toolchain executables also need to be added as a one-off setup to be used by the Nordic Makefiles. To do this, complete the following steps:

1. Open the file D:\Work\nRF5_SDK\components\toolchain\gcc\Makefile.windows
2. For **GNU_INSTALL_ROOT**, add the location of the 'bin' folder, e.g.: **D:\Work\Tools\ARM_GNU)**
3. For **GNU_VERSION**, add the version of the toolchain (obtained from Step 7 above), e.g. 5.4.1.

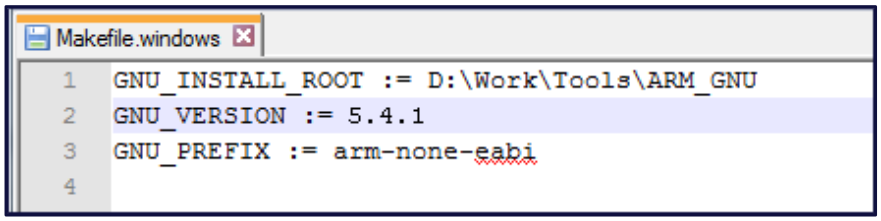

Figure 11: Adding ARM GCC to Makefile.windows

GNU Make Utility Setup

The GNU Make Utility allows the build of the project using a Makefile. This means that project settings in Eclipse will not affect the build process and that the project is self-contained through the Makefile. To setup the Make utility, complete the following steps:

1. Download the Windows installer file from the following link: https://sourceforge.net/projects/gnuwin32/files/make/
2. Launch the executable
3. Select the Destination Location to be a known folder, e.g.: **D:\Work\Tools\Make** (Figure 12)

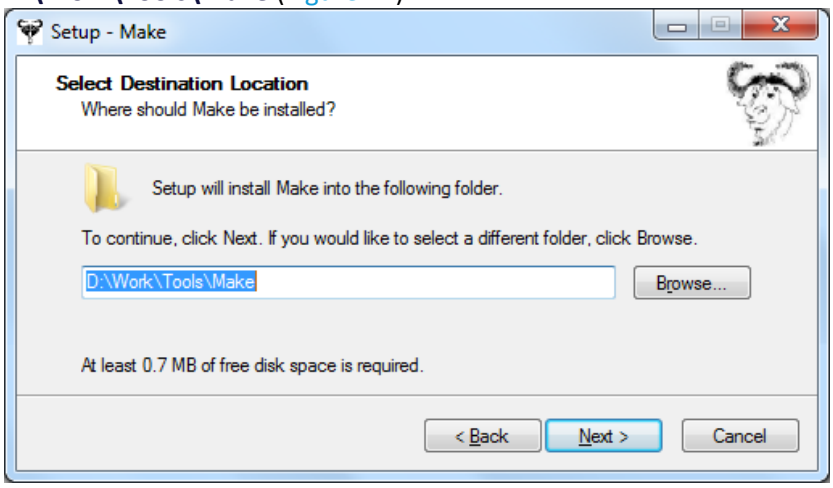

Figure 12: Installing Make utility

4. Proceed through the installation by clicking **Next** in the windows that follow, until the installation is complete.

Note: Some versions of MAKE prompt you to add the toolchains to the PATH variable. If this is the case, then select this option and skip to the next section.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

To use Make from any folder, add the GNU Make executable to the PATH environment variable by doing the following:

1. Right click on **Computer > Properties**.
2. Select **Advanced system settings**.
3. Select **Environment Variables**.
4. In the System variables section, select **Path** and click **Edit**…
5. Add the location of the GNU Make utility executable preceded by a semicolon (;), e.g.: D:\Work\Tools\GnuWin32\bin
6. Open the command prompt (if already open cmd has to be reopened) and confirm that Make is set up correctly by typing:

```
make --version
```

The above command should return the version number of make. If it doesn't, then this means that the make executable location was not added to the PATH variable successfully (Figure 13).

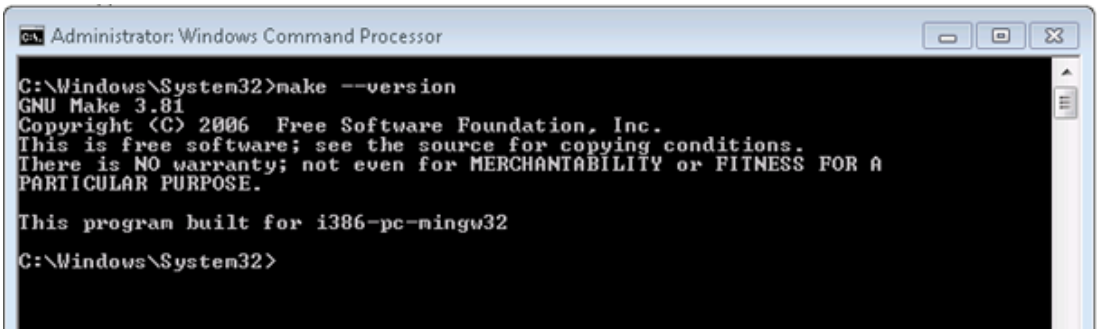

Figure 13: Using GNU Make through the command prompt

GNU CoreUtils for Windows

Due to the use of UNIX shell commands in the SDK makefiles, the GNU CoreUtils package must be installed for these commands to work. To do this, complete the following steps:

1. Download the CoreUtils Binaries zip file from http://gnuwin32.sourceforge.net/packages/coreutils.htm
2. Extract the files into a known folder, e.g.:
   D:\Work\Tools\CoreUtils

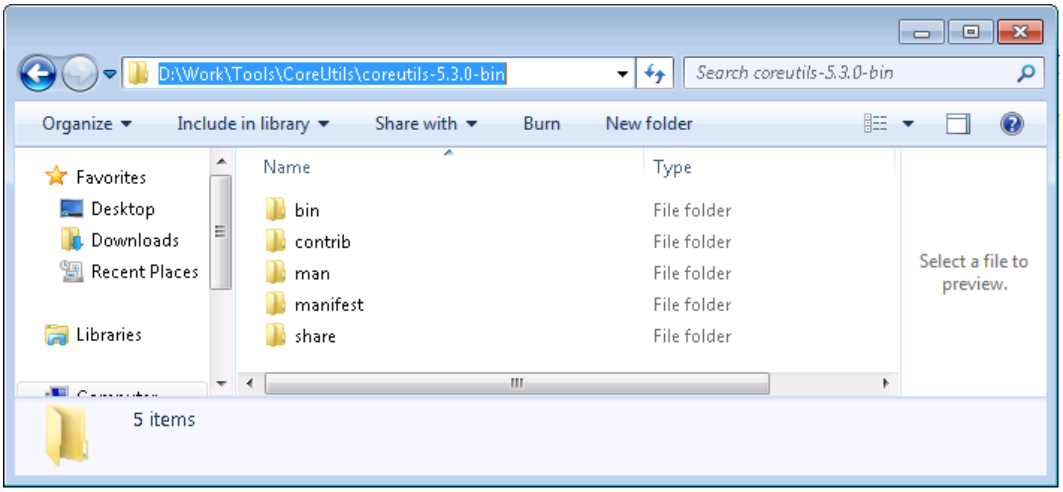

Figure 14: CoreUtils binaries location

To use CoreUtils from any folder, add the GNU CoreUtils binaries to the PATH environment variable by doing the following:

1. Right click on **Computer > Properties**.
2. Select **Advanced system settings**.
3. Select **Environment Variables**.
4. In the System variables section, select **Path** and click **Edit**…
5. Add the location of the GNU Make utility executable preceded by a semicolon (;), e.g.: **D:\Work\Tools\CoreUtils\coreutils-5.4.0-bin\bin**
6. Open the command prompt (if already open cmd has to be reopened) and confirm that the CoreUtils are set up correctly by typing:

```
rm --version
```

The above command should return the version number of rm. If it doesn't, then this means that the CoreUtils binaries location was not added to the PATH variable successfully (Figure 15).

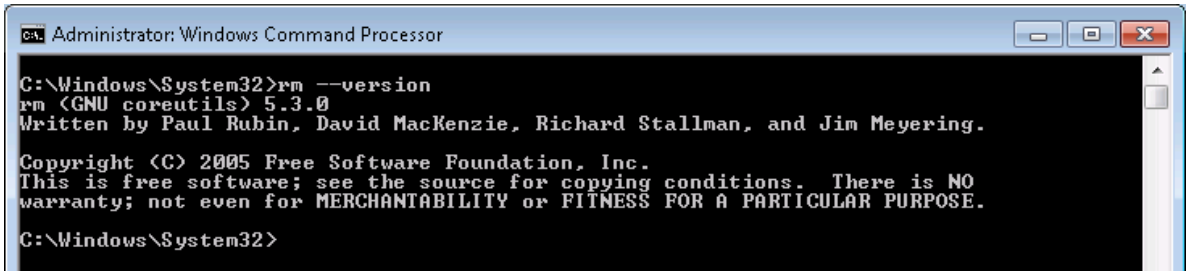

Figure 15: Using CoreUtils through the command prompt

Building and Debugging SDK Examples

Example of Building the HRS

In the next steps, we import and configure an existing HRS application. This sample HRS application comes as part of the nRF5 SDK.

1. Launch Eclipse.
2. Create a new project by selecting **File** > **New** > **C Project**.
3. In the **Project name** field, type *hrs*.
4. From the Project type section, select **Makefile project** > **Empty Project**. In the Toolchains section select – **Other Toolchain** –.
5. Click **Finish**. The project should now appear on the Project Explorer tab on the left (Figure 16).

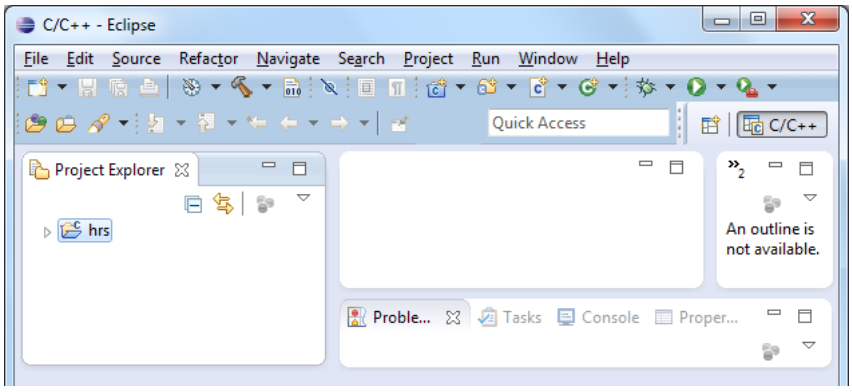

Figure 16: Creating project in Eclipse

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

6. Right click on the project (hrs) and select **New**->**Folder**.
7. Click the **Advanced** button at the bottom of the Window.
8. Select **Link to alternate location (Linked Folder)** and browse to the location of the hrs project, e.g.: **D:\Work\nRF5_SDK\examples\ble_peripheral\ble_app_hrs** (Figure 17).

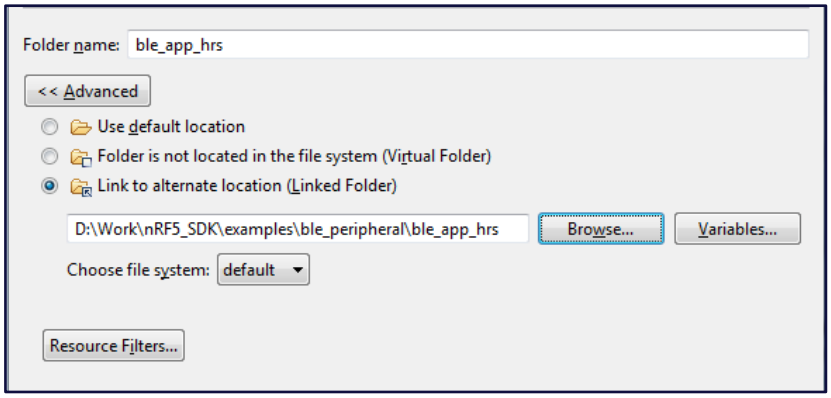

Figure 17: Adding files and folders to the project

9. The header and source files should now appear within the project (Figure 18).

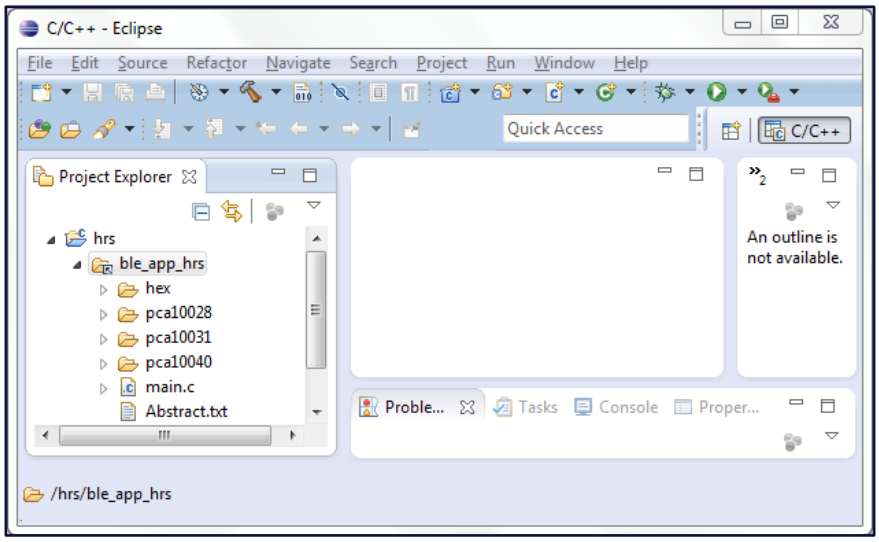

Figure 18: Project files and folders added in Project Explorer

10. Right click on the project (hrs) and select **Properties**. Select the **C/C++ Build** section.
11. Under the **Builder Settings** tab In the **Build directory** field, type the location of the Makefile that is used to build the project, e.g.:
D:\Work\nRF5_SDK\examples\ble_peripheral\ble_app_hrs\pca10040\s132\armgcc (Figure 19).

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

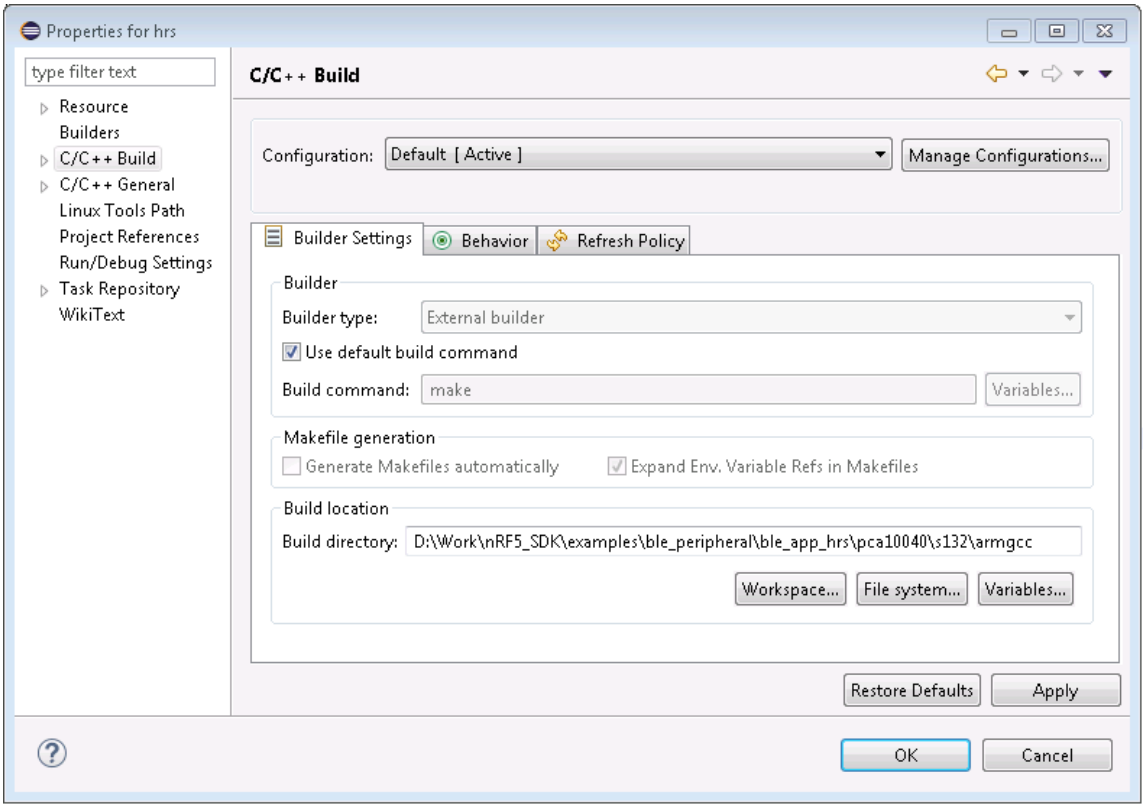

Figure 19: Adding Makefile location to project properties

12. Select **OK**.
13. You can now build the project by clicking **Project > Build All**. Alternatively, click the Build button or **CTRL+B**. Once the project is built, the hex and binary files are generated as shown in the figure below (Figure 20).

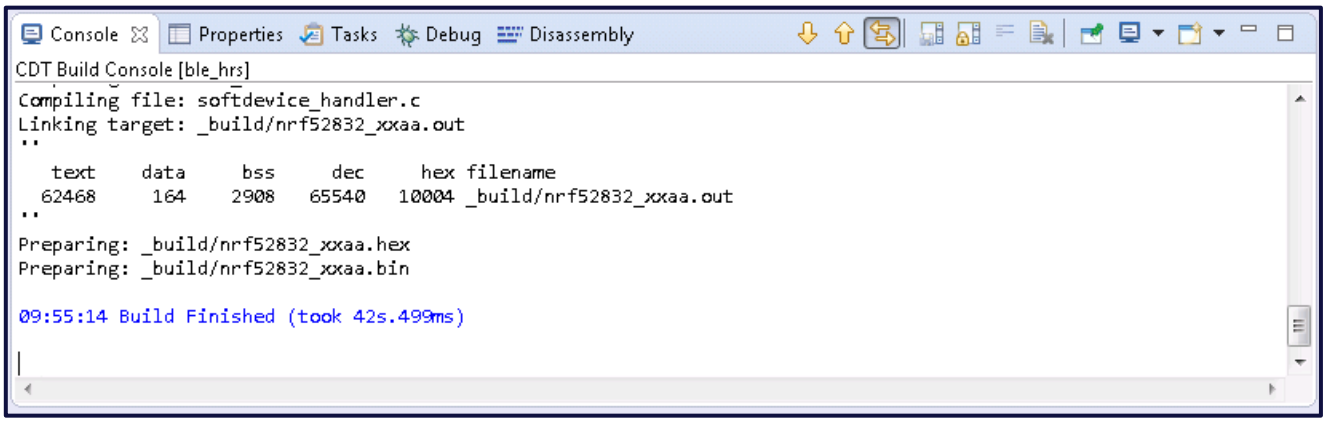

Figure 20: Building the HRS example

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

Flashing the HRS example

Connect the BL652 to the PC through USB2 port on the development kit (Figure 21).

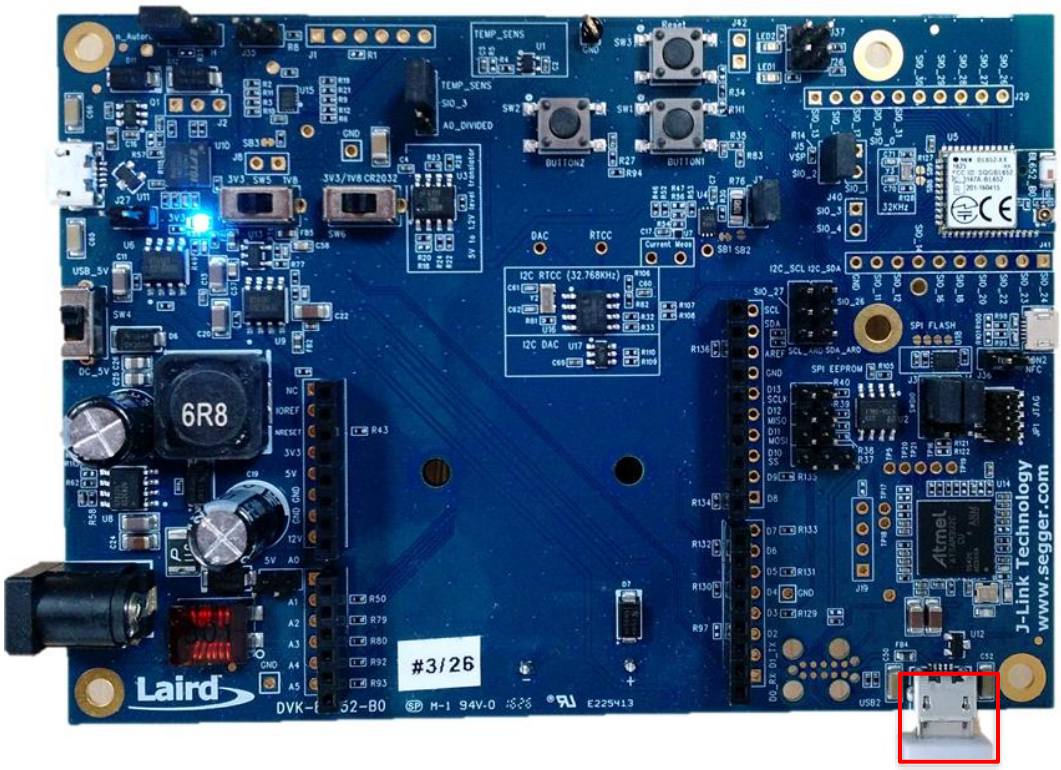

Figure 21: Connecting BL652 to PC through USB2 port

Flash downloading to the BL652 must be performed through the external toolchain configuration in Eclipse. For the hrs project, four external commands need to be added:

Erase

1. In Eclipse, select **Run > External Tools > External Tools Configuration**…
2. Right click on **Program** > **New**. In the Name field type *Erase*.
3. In the Location field, type the path of the latest version of nrfjprog.exe. This is usually located in the Nordic Semiconductor folder in Program Files e.g.:
   C:\Program Files (x86)\Nordic Semiconductor\nrf5x\bin\nrfjprog.exe
4. In the Arguments field, type *--eraseall –f nrf52*.
5. Switch to the **Build** tab and deselect **Build before launch**.
6. Click **Apply,** then click **Run** (Figure 22).

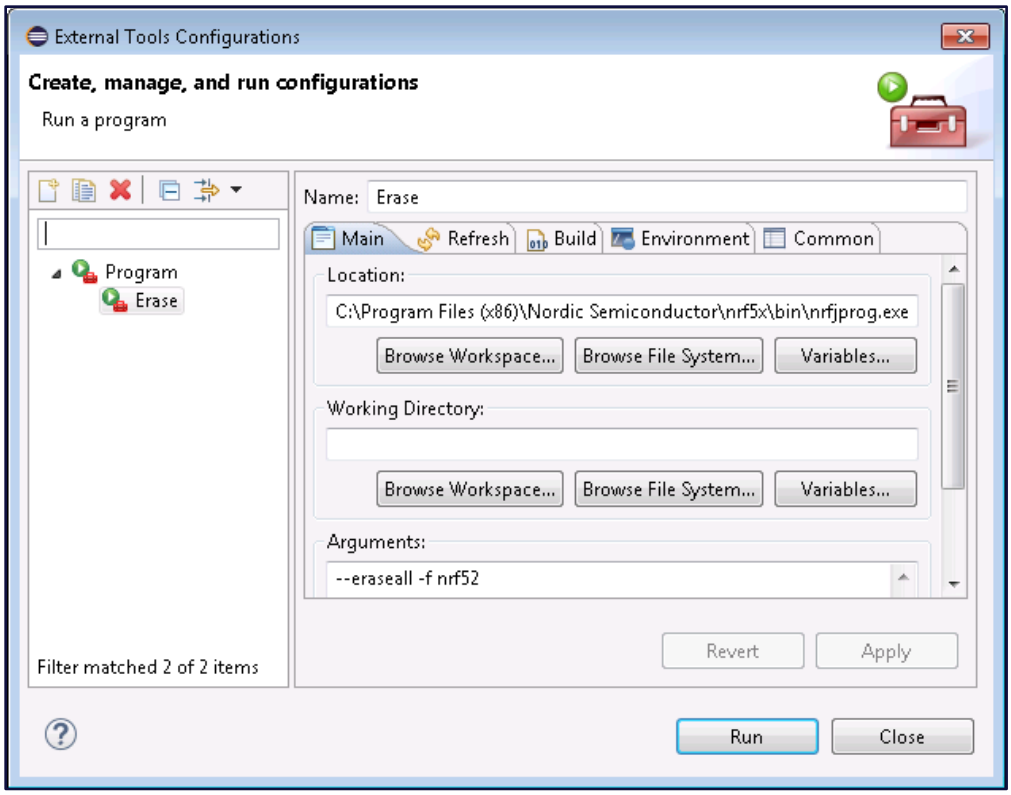

Figure 22: Adding Erase to External Tools Configurations

Flash Softdevice

1. Select **Run > External Tools > External Tools Configuration**…
2. Right click on **Program** > **New**. In the Name field type Flash_Softdevice.
3. In the **Location** field, type the path of the latest version of *nrfjprog.exe*. This is usually located in the Nordic Semiconductor folder in Program Files e.g.:
   C:\Program Files (x86)\Nordic Semiconductor\nrf5x\bin\nrfjprog.exe
4. In the Working Directory, enter the directory where the hex file of the softdevice is located. This will be in **D:\Work\nRF5_SDK\components\softdevice\s132\hex**
5. In the Arguments field, enter the --program command followed by the name of the hex file and the device family argument. e.g.:
   --program s132_nrf52_5.0.0_softdevice.hex -f nrf52
6. Switch to the **Build** tab and deselect **Build before launch**
7. Click **Apply,** then click **Run** (Figure 23).

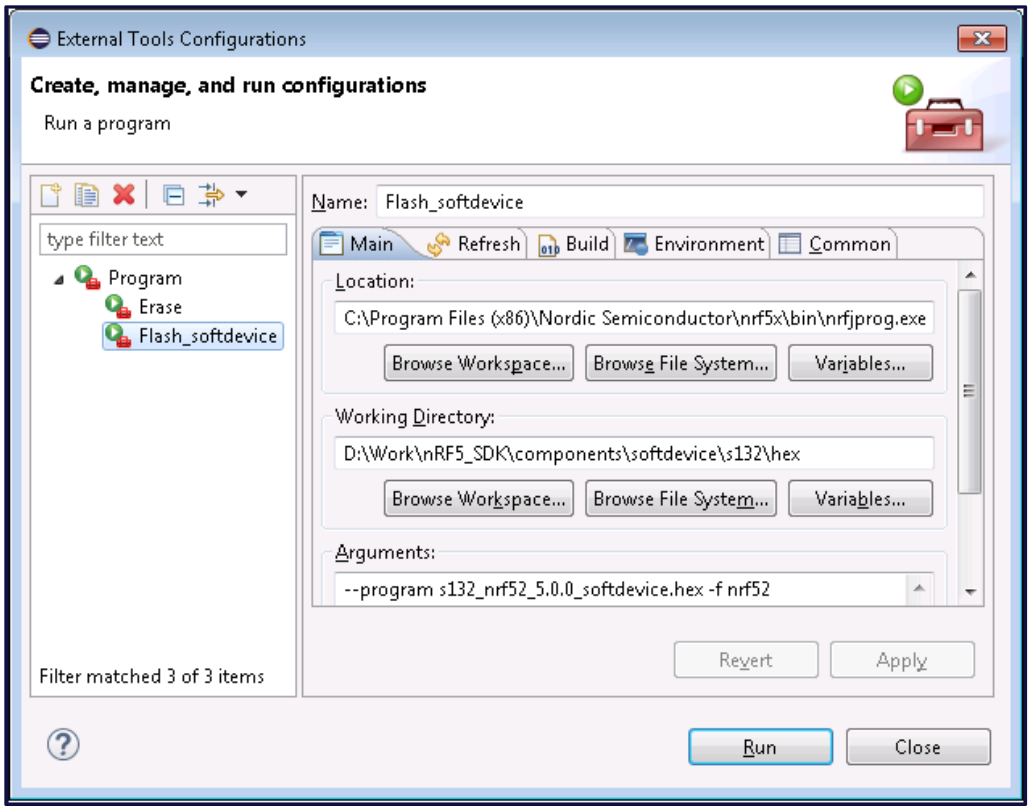

Figure 23: Adding Flash_softdevice to External Tools Configurations

Flash BL652

1. Select **Run > External Tools > External Tools Configuration**…
2. Right click on **Program** > **New**. In the Name field type *Flash_hrs*.
3. In the Location field, type the path of the latest version of nrfjprog.exe. This is usually located in the Nordic Semiconductor folder in Program Files e.g.:
   C:\Program Files (x86)\Nordic Semiconductor\nrf5x\bin\nrfjprog.exe
4. In the Working Directory, enter the name of where the generated *.hex file will be located, e.g.:
   D:\Work\nRF5_SDK\examples\ble_peripheral\ble_app_hrs\pca10040\s132\armgcc_build
5. In the Arguments field, type command used for programming the devkit. E.g.:
   --program nrf52832_xxaa.hex -f nrf52
6. Switch to the **Build** tab and deselect **Build before launch**.
7. Click **Apply** and then **Run.**

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

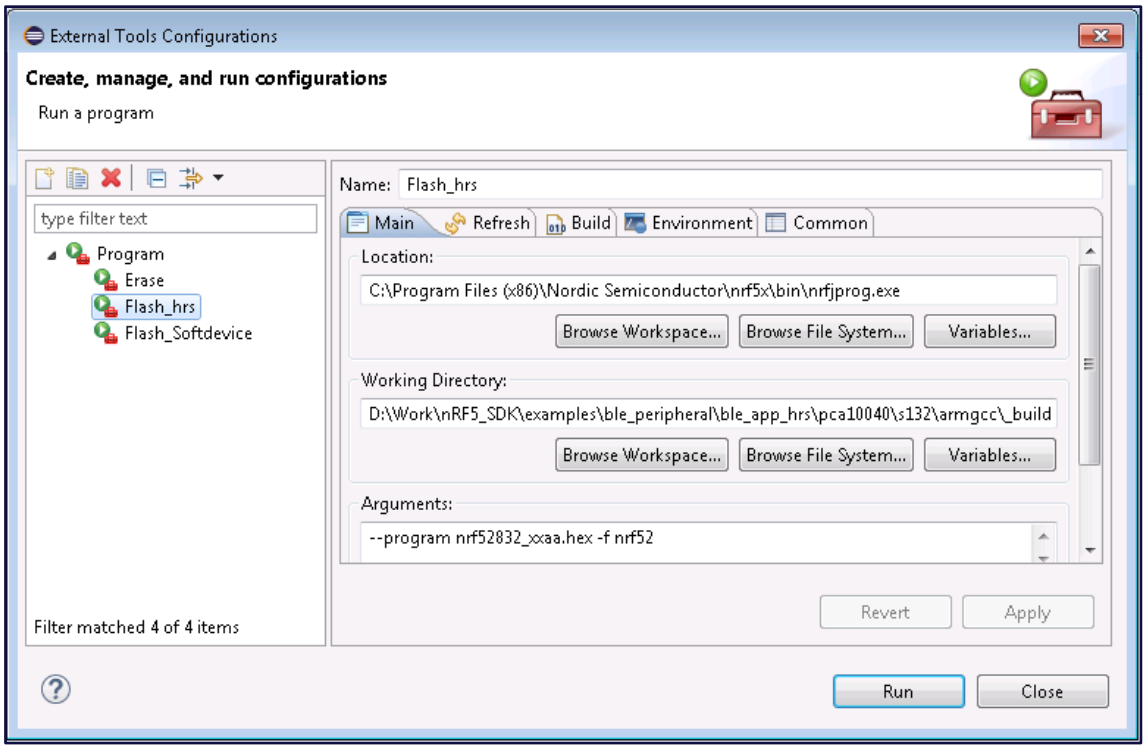

Figure 24: Adding Flash_hrs to External Tools Configurations

Reset

1. Select **Run > External Tools > External Tools Configuration**…
2. Right click on **Program** > **New**. In the Name field type *Reset*.
3. In the Location field, type the path of the latest version of nrfjprog.exe. This is usually located in the Nordic Semiconductor folder in Program Files e.g.:
   C:\Program Files (x86)\Nordic Semiconductor\nrf5x\bin\nrfjprog.exe
4. In the Arguments field, type - *-reset –f nrf52*
5. Switch to the **Build** tab and deselect **Build before launch**.
6. Click **Apply** and then click **Run** (Figure 25)**.**

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

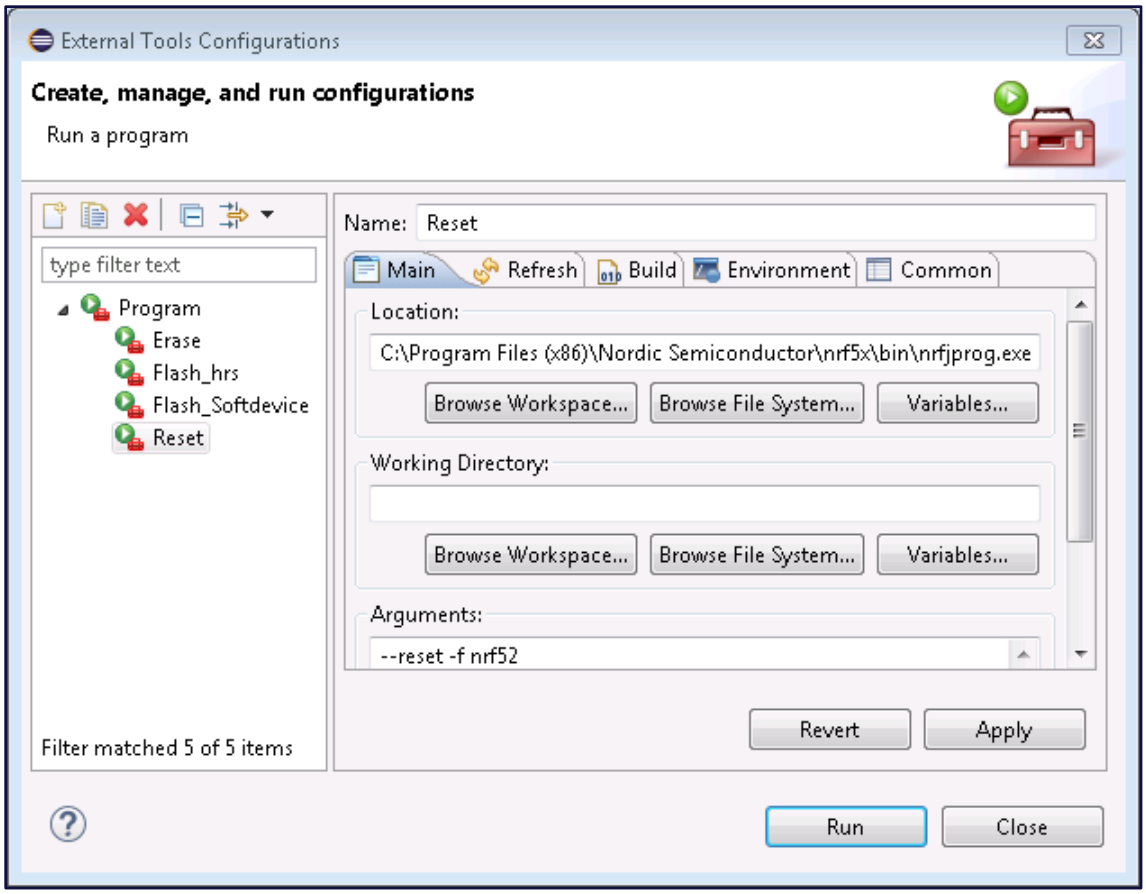

Figure 25: Adding Reset to External Tools Configurations

Once this is done and setup, you can now erase and flash the hrs app onto the module as follows:

1. **Run > External Tools > Erase**
2. **Run > External Tools > Flash_Softdevice**
3. **Run > External Tools > Flash_hrs**
4. **Run > External Tools > Reset**

Alternatively, the shortcut button for external tool configuration can be used.

Debugging the HRS example

Debugging an application on the BL652 allows you to see what is going on inside the program while it runs. This can help detect problems and issues, as well as show the values of the variables and the flow of the program at different stages. In order to debug the hrs application, complete the following steps: -

1. In Eclipse, select **Run** > **Debug Configurations**
2. Right click **GDB-SEGGER J-Link Debugging > New**

Note: If GDB SEGGER J-Link Debugging does not appear as one of the Debug Configurations options, then it has not been installed properly as part of nRF5x-Command-Line-Tools. You might need to install the J-Link Software from https://www.segger.com/downloads/jlink. Also ensure that the Eclipse Debugging Tools have been installed properly.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

3. In the **Name** field, specify the name of the debugging option, e.g. *hrs*.
4. In the **Project** field, specify the name of the project being debugged, e.g. *hrs.*
5. In the **C/C++ Application** field, provide the location of the *.out file. This is usually in the _build folder of the SDK example, e.g.:
   D:/Work/nRF5_SDK/examples\ble_peripheral/ble_app_hrs\pca10040/s132/armgcc/_build/nrf52832_xxaa.out
6. Select **Disable auto build** (Figure 26).

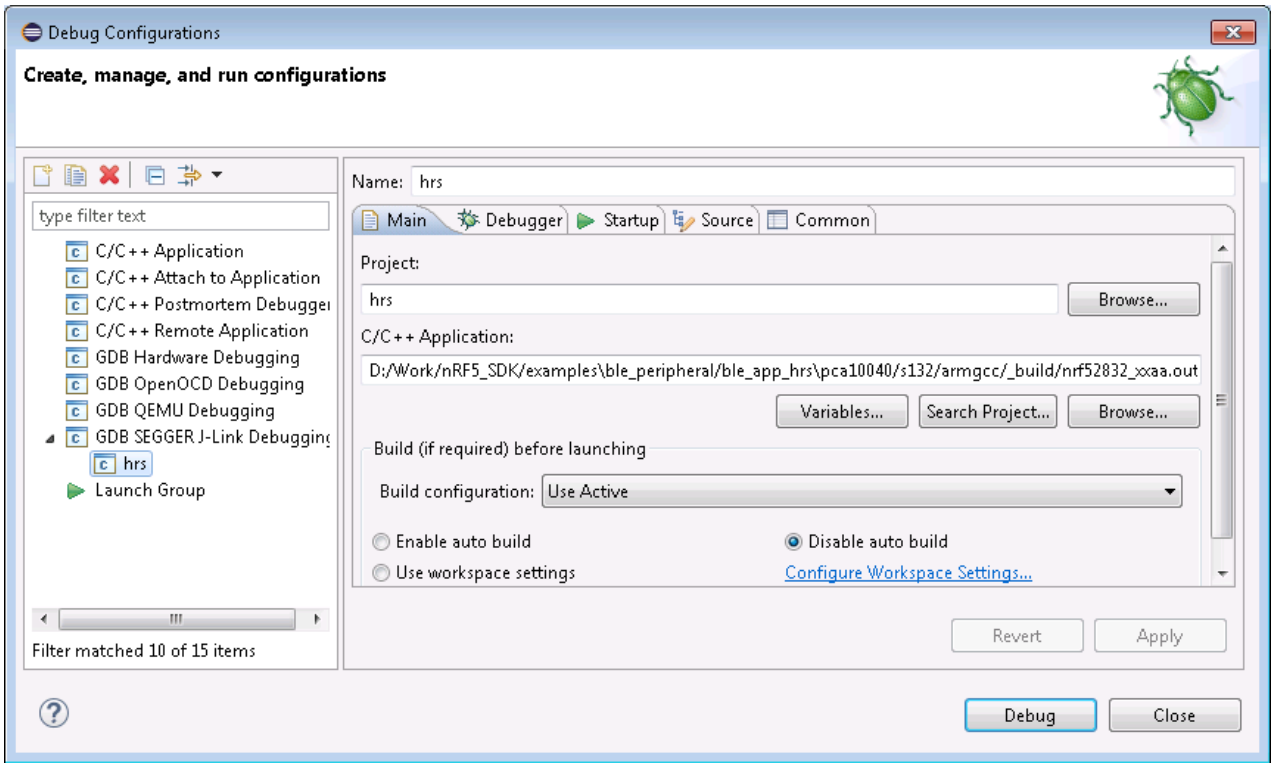

Figure 26: Debug Configurations for the hrs app

7. Switch to the **Debugger** tab.
8. In the **Device name**, enter Cortex-M4.
9. Scroll down until you see *GDB Client Setup*.
10. In the **Executable** field, enter the following: *arm-none-eabi-gdb.exe* (Figure 27).

Note: If there is more than one device connected, ensure that the BL652's Segger Serial Number is specified in the **USB serial** field in the Debugger tab.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

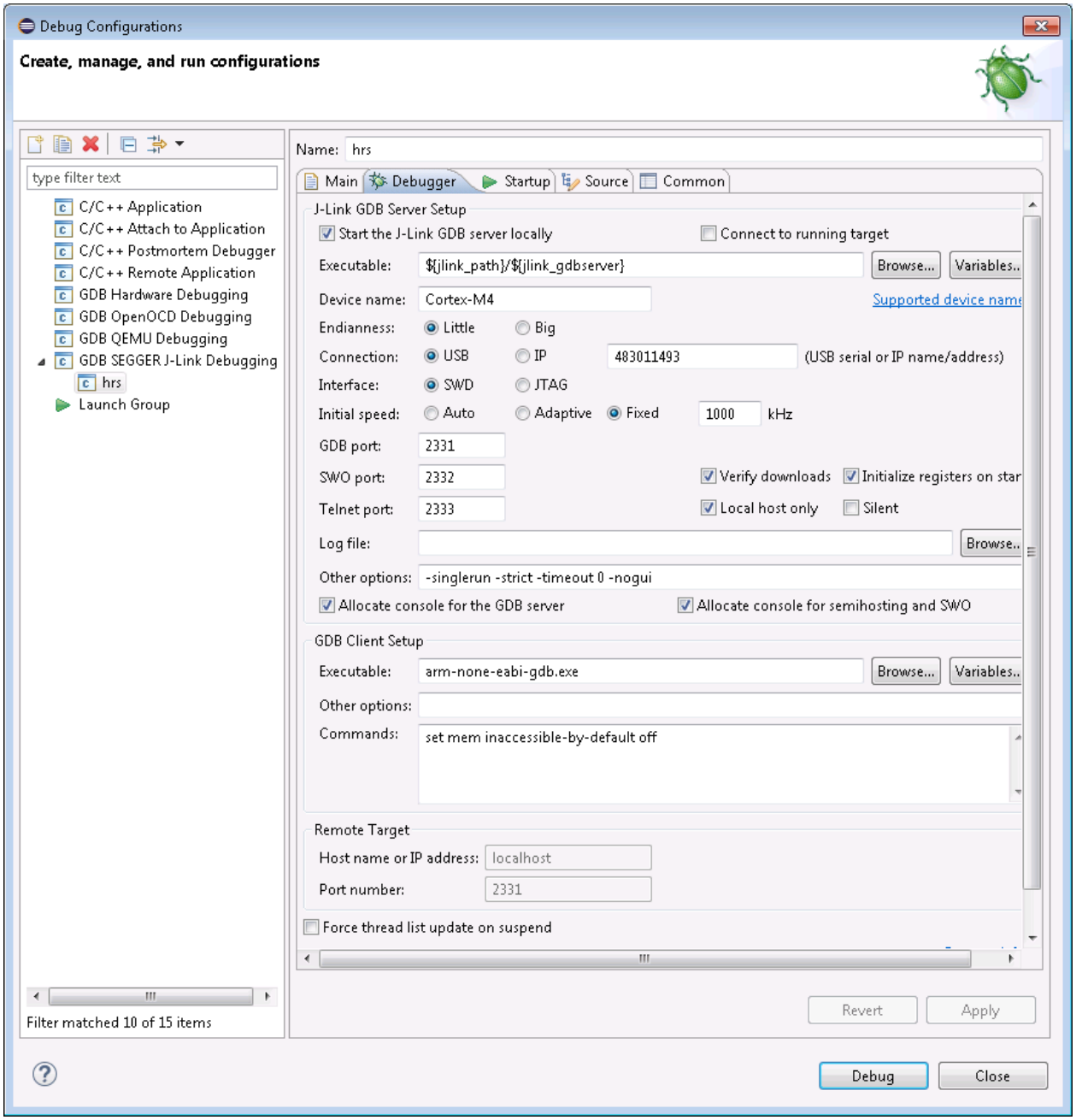

Figure 27: Debugger tab in the Debug Configurations for the hrs app

11. Click **Apply,** then click **Debug.**
12. You should now be in the debug view. You can step over, step into, and add breakpoints through the Run from the toolbar (Figure 28).

```
1018
1019 /**@brief Function for application main entry.
1020  */
1021 int main(void)
1022 {
1023     bool erase_bonds;
1024
1025     // Initialize.
1026     log_init();
1027     timers_init();
1028     buttons_leds_init(&erase_bonds);
1029     ble_stack_init();
1030     gap_params_init();
1031     gatt_init();
1032     advertising_init();
1033     services_init();
1034     sensor_simulator_init();
1035     conn_params_init();
1036     peer_manager_init();
1037
1038     // Start execution.
1039     NRF_LOG_INFO("Heart Rate Sensor example started.");
1040     application_timers_start();
1041
1042     advertising_start(erase_bonds);
1043
1044     // Enter main loop.
1045     for (;;)
1046     {
1047         if (NRF_LOG_PROCESS() == false)
1048         {
1049             power_manage();
1050         }
1051     }
1052 }
```

Figure 28: Debugging hrs application

REFERENCE

Further information relating to different utilities used in this app note can be found here:

- BL652 Product Page - http://www.lairdtech.com/products/bl652-ble-module
- Make - https://www.gnu.org/software/make/
- ARM Toolchain - https://launchpad.net/gcc-arm-embedded
- Eclipse - https://eclipse.org/org/
- Nordic Blog on using Eclipse and GCC - https://devzone.nordicsemi.com/blogs/18/development-with-eclipse-and-gcc/
- Nordic Tutorial on using Eclipse and GCC - https://devzone.nordicsemi.com/tutorials/7/development-with-gcc-and-eclipse/

REVISION HISTORY

Version	Date	Notes	Approver
1.0	16 Oct 2017	Separated from the SDK v11.0.0 document Updated for newer SDK (v14) and software	Youssif Saeed

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600