Application Note

Compiling DA14585 IoT Sensors Reference Application with Eclipse/GCC

AN-B-064

Abstract

The DA14585 IoT Sensors Reference Application can be compiled using the ARM GCC compiler and Dialog Semiconductor provides an example of Eclipse project to how to accomplish this. This document describes the required steps to download and compile this project.

Contents

Figures

1 Terms and Definitions

BLE	Bluetooth Low Energy
CDT	C/C++ Development Tooling
GAP	Generic Access Profile
GAPC	Generic Access Profile Controller
GAPM	Generic Access Profile Manager
GATT	Generic Attribute Profile
IDE	Integrated Development Environment
IoT	Internet of Things
MSK	Multi Sensor Development Kit
PC	Personal Computer
RAM	Random Access Memory
SOC	System on Chip

2 References

[1] UM-B-096, DA14585 IoT Multi Sensor Development Kit Software Reference Applications, User Manual, Dialog Semiconductor.

[2] UM-B-095, DA14585 IoT Multi Sensor Development Kit Hardware Design, User Manual, Dialog Semiconductor.

[3] UM-B-057, Smart Snippets^{TM} Studio User Guide, User Manual, Dialog Semiconductor.

3 Introduction

Dialog Semiconductor's IoT multi sensor development kit (MSK) [1], a compact development kit, offers 15 degrees of freedom accompanied by five software reference applications [2] and incorporates versatile sensor management with cloud applications. The IoT MSK is based on DA14585 Bluetooth Low Energy (BLE) system-on-chip (SoC) and a number of motion and environmental sensors.

The provided reference applications can be compiled with Keil µVision and ARMCC toolchain which is the recommended development environment. Many developers would like to use the free GCC compiler/linker and Dialog has provided an example Eclipse/GCC project port of the IoT Sensors Reference Application. For more details on this application refer to *section 5* of [1].

This document describes how to setup the environment, download, compile, and debug the IoT Sensors Reference Application using Eclipse/GCC. It is required that users should already be familiar with Eclipse CDT environment.

4 Installation

4.1 Prerequisites

- A Personal Computer running Windows 7 or Windows 10.
- SmartSnippets Studio v2.0.5.
- A DA14585 IoT MSK.
- A Communication Interface Board.

4.2 SmartSnippets™ Installation

Dialog's SmartSnippets™ Studio is a royalty-free software development platform for Smartbond™ devices. For detailed information on SmartSnippets™ Studio refer to [3].

SmartSnippets™ Studio contains:

- SmartSnippets™ IDE: Eclipse CDT based IDE with pre-configured plugins to provide the build/debug environment
- SmartSnippets™ Toolbox which covers all software development requirements, including:
  - programming and loading firmware into SRAM, OTP, and Flash
  - power profiling
- SmartSnippets™ documentation.

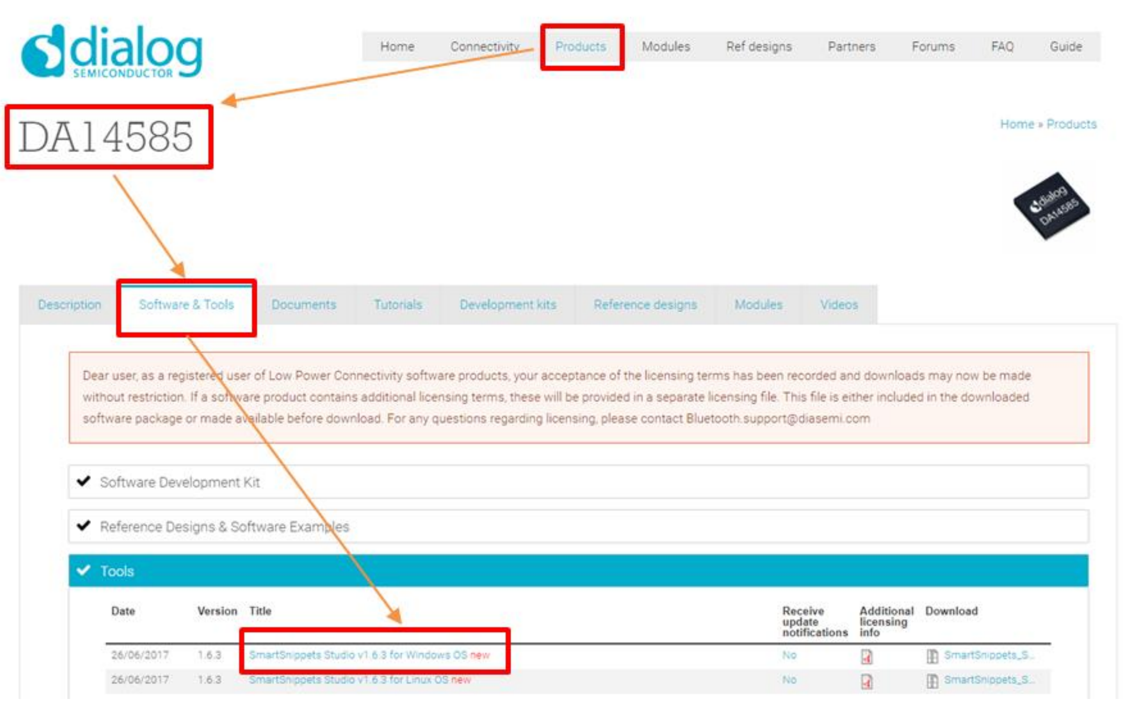
The SmartSnippets™ IDE is enabled by an on-board J-Link debugger from SEGGER. This offers standard debug capabilities such as single stepping, setting breakpoints, software download, and many more. For more details on the debugger capabilities, visit https://www.segger.com.

The installation procedure for SmartSnippets™ Studio is described in [3]. A summary of the steps is given here:

1. Navigate to Dialog's support portal https://support.dialog-semiconductor.com and choose **Connectivity** > **Products** > **DA14585** > **Software&Tools** > **Tools** (Figure 1) and download the latest 2.0.x version of SmartSnippets™ Studio.
2. Run the SmartSnippets™ Studio installer (.msi). Several required tools are automatically installed and others need to be manually downloaded and installed.

3. Select the already installed version of SEGGER J-Link GDB server or install the latest version (Figure 2) and click "**Next**".
4. Select the destination folder (Figure 3) for the SmartSnippets™ Studio and click"**Next**". The default installation location `C:\DiaSemi` is recommended.

The SmartSnippets™ Studio will be installed. Please Note that some software components during installation may require administrator privileges.



Dear user, as a registered user of Low Power Connectivity software products, your acceptance of the licensing terms has been recorded and downloads may now be made without restriction. If a software product contains additional licensing terms, these will be provided in a separate licensing file. This file is either included in the downloaded software package or made available before download. For any questions regarding licensing, please contact Bluetooth.support@diasemi.com

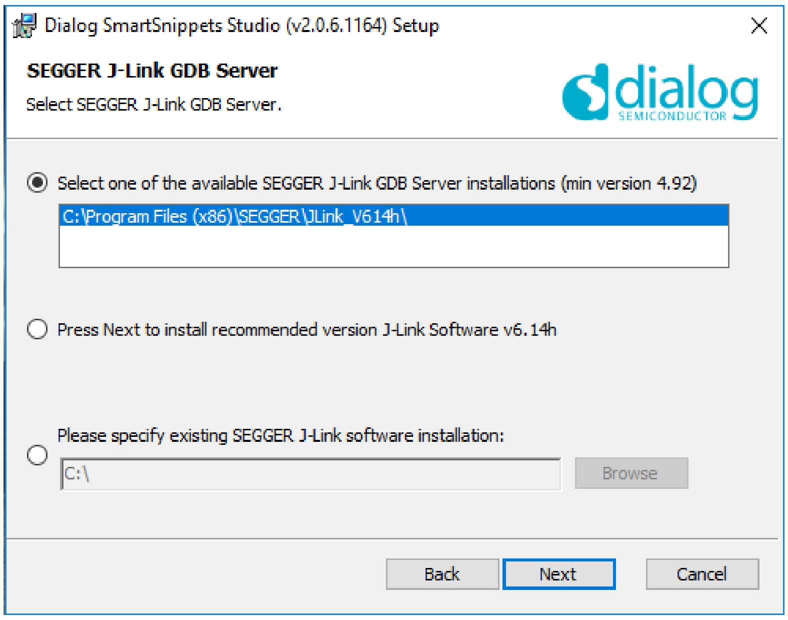
✓ Software Development Kit

✓ Reference Designs & Software Examples

✓ Tools

Date	Version	Title	Receive update notifications	Additional licensing info	Download
26/06/2017	1.6.3	SmartSnippets Studio v1.6.3 for Windows OS new	No		SmartSnippets_S...
26/06/2017	1.6.3	SmartSnippets Studio v1.6.3 for Linux OS new	No		SmartSnippets_S...

Figure 1: Download SmartSnippets^{TM} Studio



SEGGER J-Link GDB Server

Select SEGGER J-Link GDB Server.

[✓ Select one of the available SEGGER J-Link GDB Server installations (min version 4.92)]

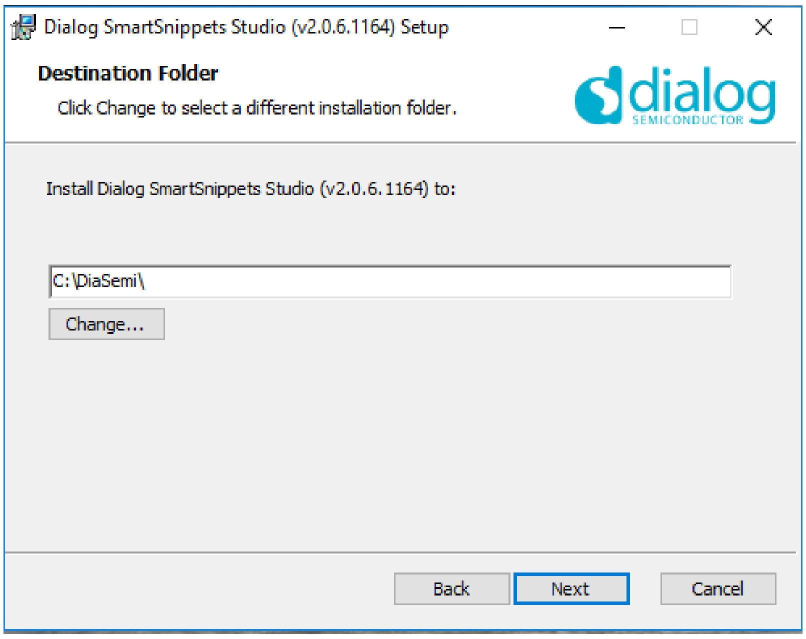
C:\Program Files (x86)\SEGGER\JLink_V614h\

Press Next to install recommended version J-Link Software v6.14h

Please specify existing SEGGER J-Link software installation:

C:\

Figure 2: Use Pre-Existing J-Link


Dialog SmartSnippets Studio (v2.0.6.1164) Setup
Destination Folder
Click Change to select a different installation folder.
dialog SEMICONDUCTOR
Back
Next
Cancel
Change...

Install Dialog SmartSnippets Studio (v2.0.6.1164) to:

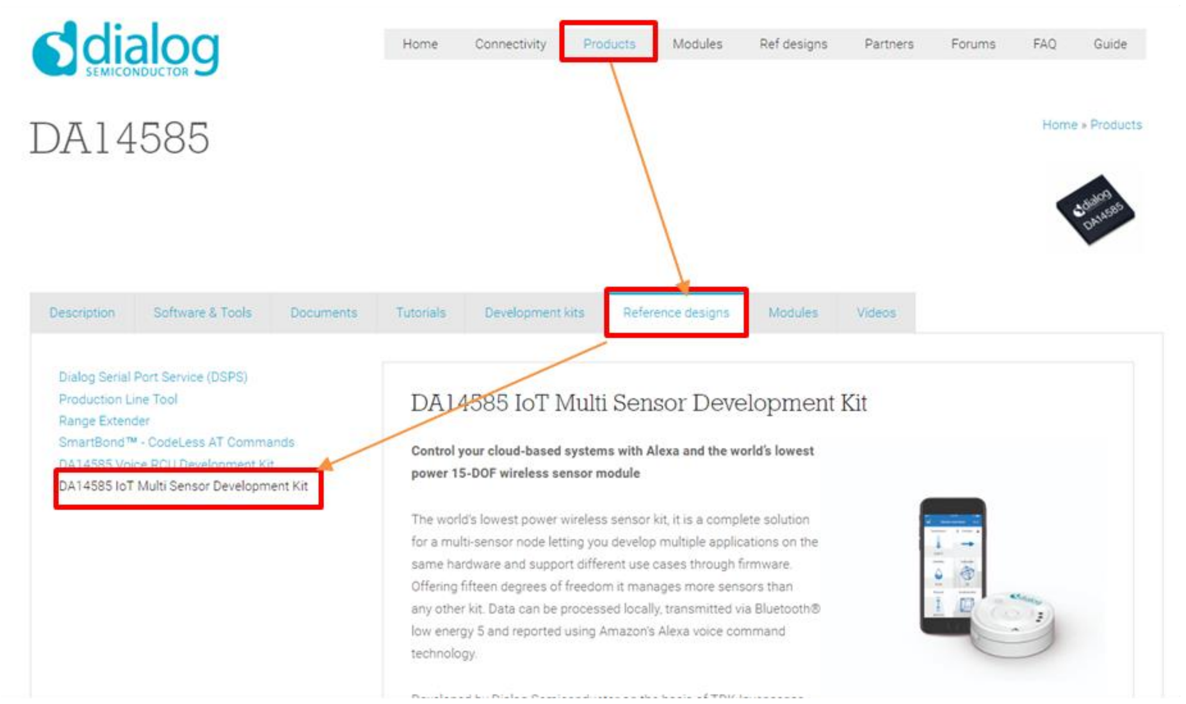
C:\DiaSemi\

Figure 3: Select SmartSnippets^{TM} Studio Installation Directory

4.3 Download the DA14585 IoT MSK Eclipse/GCC Example

Like SmartSnippets™ Studio, the Eclipse/GCC example can be downloaded from **Connectivity** > **Products** > **DA14585** > **Reference Designs** > **DA14585 IoT Multi Sensor Development Kit** (Figure 4) in the support portal.

The filename of the example is `DA14585_IOT_MULTI_SENSOR_DK_GCC_EXAMPLE_v6.160.x.zip` and includes only the GCC port of the software. The release folder `v6.160.x` included in the zipped file should be extracted by users and placed in a convenient path.



DA14585 IoT Multi Sensor Development Kit

Control your cloud-based systems with Alexa and the world's lowest power 15-DOF wireless sensor module

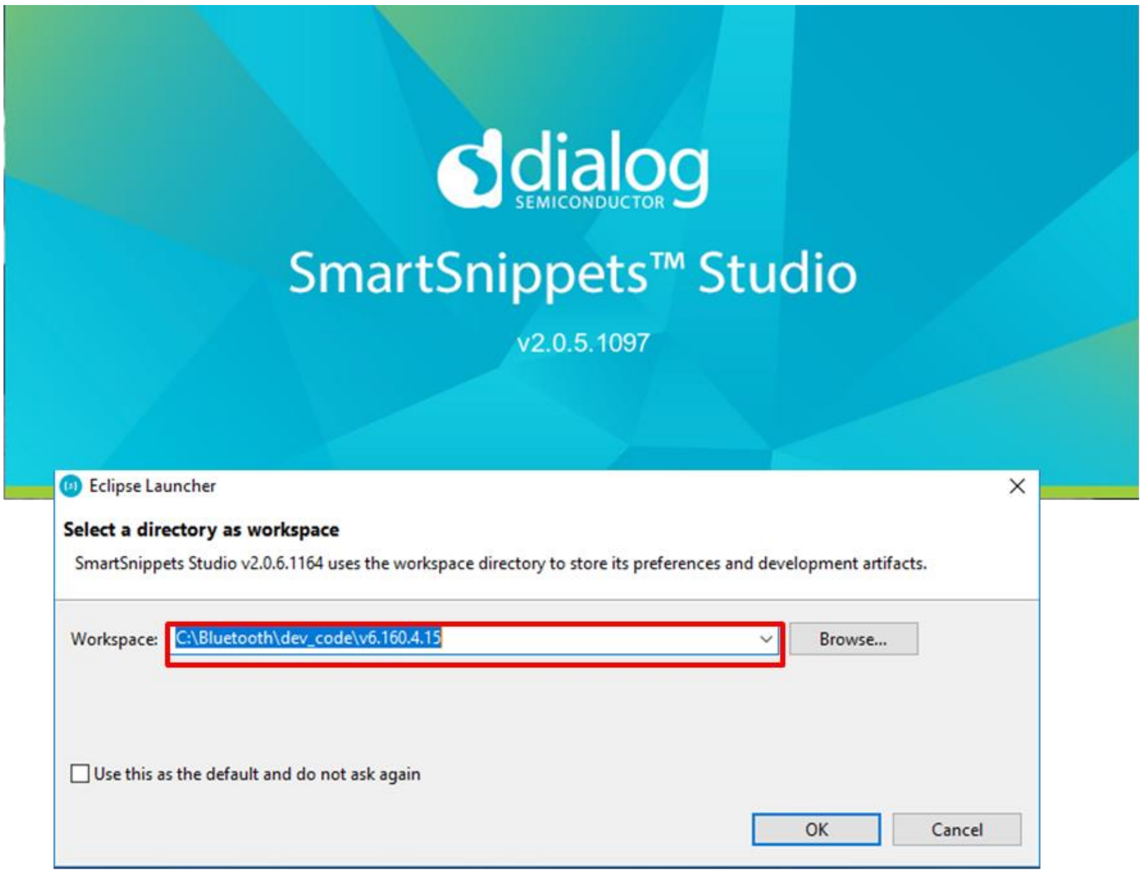
The world's lowest power wireless sensor kit, it is a complete solution for a multi-sensor node letting you develop multiple applications on the same hardware and support different use cases through firmware. Offering fifteen degrees of freedom it manages more sensors than any other kit. Data can be processed locally, transmitted via Bluetooth® low energy 5 and reported using Amazon's Alexa voice command technology.

Figure 4: Downloading Eclipse/GCC Example

5 Importing the Eclipse/GCC Project

1. Start SmartSnippets^{TM} Studio by clicking on the icon.

The SmartSnippets^{TM} Studio will ask the workspace directory. Select the root directory of the project that is named `v6.160.x` (for example, v6.160.4.15, Figure 5).



Select a directory as workspace

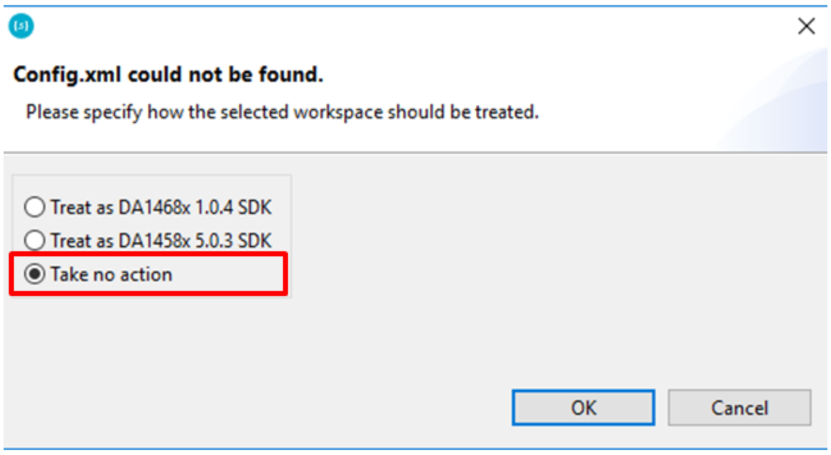
SmartSnippets Studio v2.0.6.1164 uses the workspace directory to store its preferences and development artifacts.

Workspace: C:\Bluetooth\dev_code\v6.160.4.15

☐ Use this as the default and do not ask again

Figure 5: Starting SmartSnippetsTM Studio and Selecting Workspace

2. Next, when the dialog asking for a workspace type pops up, select "**Take no action**".



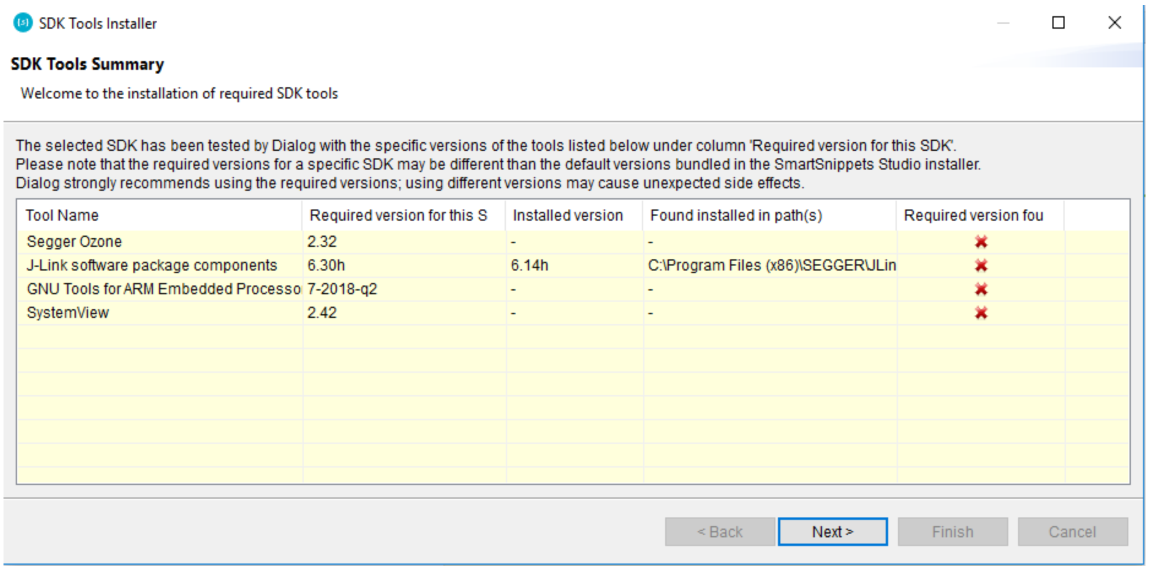
Config.xml could not be found.

Please specify how the selected workspace should be treated.

Treat as DA1468x 1.0.4 SDK [ ]
Treat as DA1458x 5.0.3 SDK [ ]
Take no action [✓]

Figure 6: Selecting Workspace Type

3. To proceed, a number of development tools should be installed (Figure 7), or their installation directory should be given (Figure 8) if they are already installed.



SDK Tools Installer

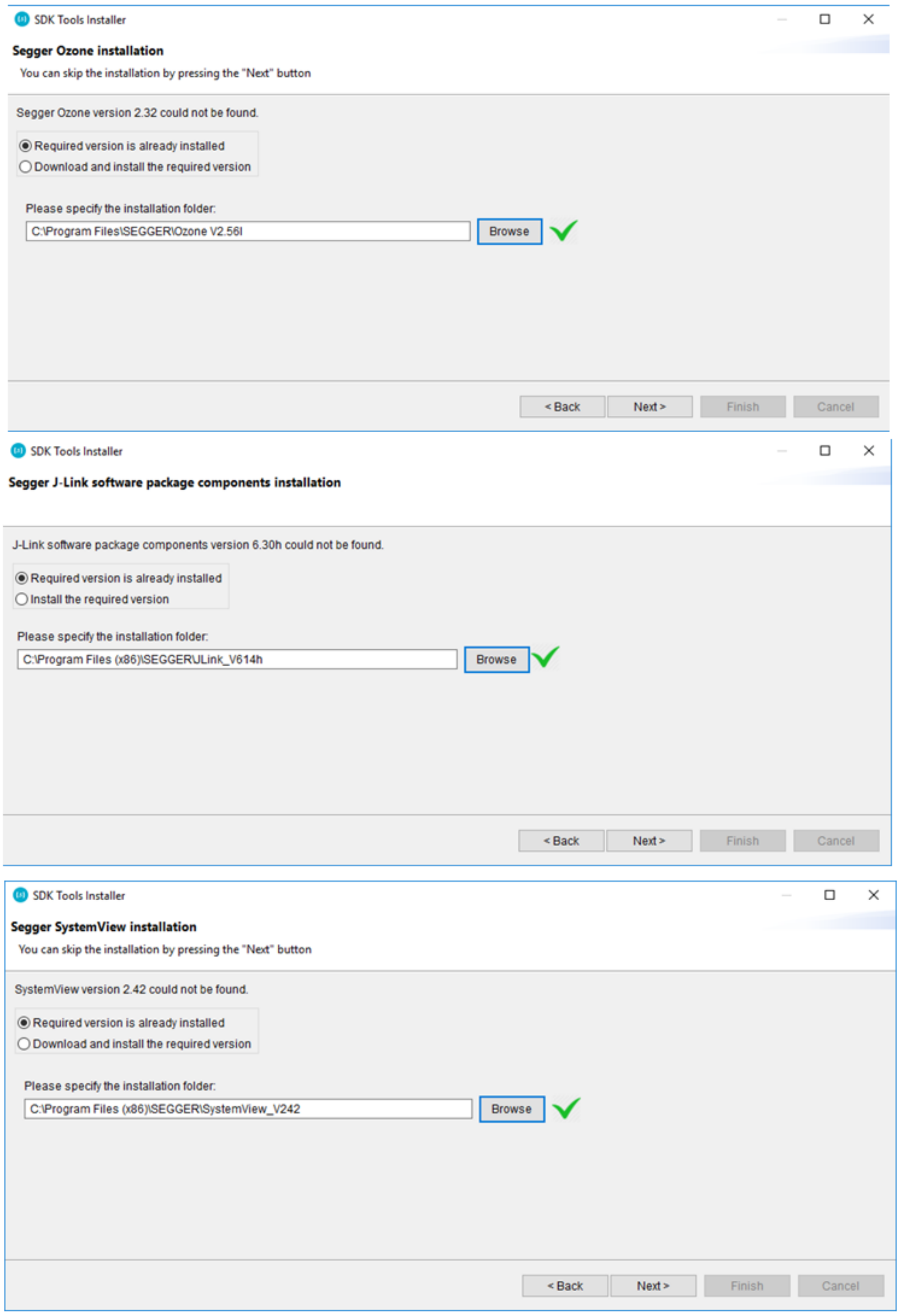
SDK Tools Summary

Welcome to the installation of required SDK tools

The selected SDK has been tested by Dialog with the specific versions of the tools listed below under column 'Required version for this SDK'.
Please note that the required versions for a specific SDK may be different than the default versions bundled in the SmartSnippets Studio installer.
Dialog strongly recommends using the required versions; using different versions may cause unexpected side effects.

Tool Name	Required version for this S	Installed version	Found installed in path(s)	Required version fou
Segger Ozone	2.32	-	-	✗
J-Link software package components	6.30h	6.14h	C:\Program Files (x86)\SEGGER\JLin	✗
GNU Tools for ARM Embedded Processo	7-2018-q2	-	-	✗
SystemView	2.42	-	-	✗

Figure 7: Tools to Be Installed


SDK Tools Installer

Segger Ozone installation

You can skip the installation by pressing the "Next" button

Segger Ozone version 2.32 could not be found.

[✓ Required version is already installed]
Download and install the required version

Please specify the installation folder:

C:\Program Files\SEGGER\Ozone V2.56l

SDK Tools Installer

Segger J-Link software package components installation

J-Link software package components version 6.30h could not be found.

[✓ Required version is already installed]
Install the required version

Please specify the installation folder:

C:\Program Files (x86)\SEGGER\JLink_V614h

SDK Tools Installer

Segger SystemView installation

You can skip the installation by pressing the "Next" button

SystemView version 2.42 could not be found.

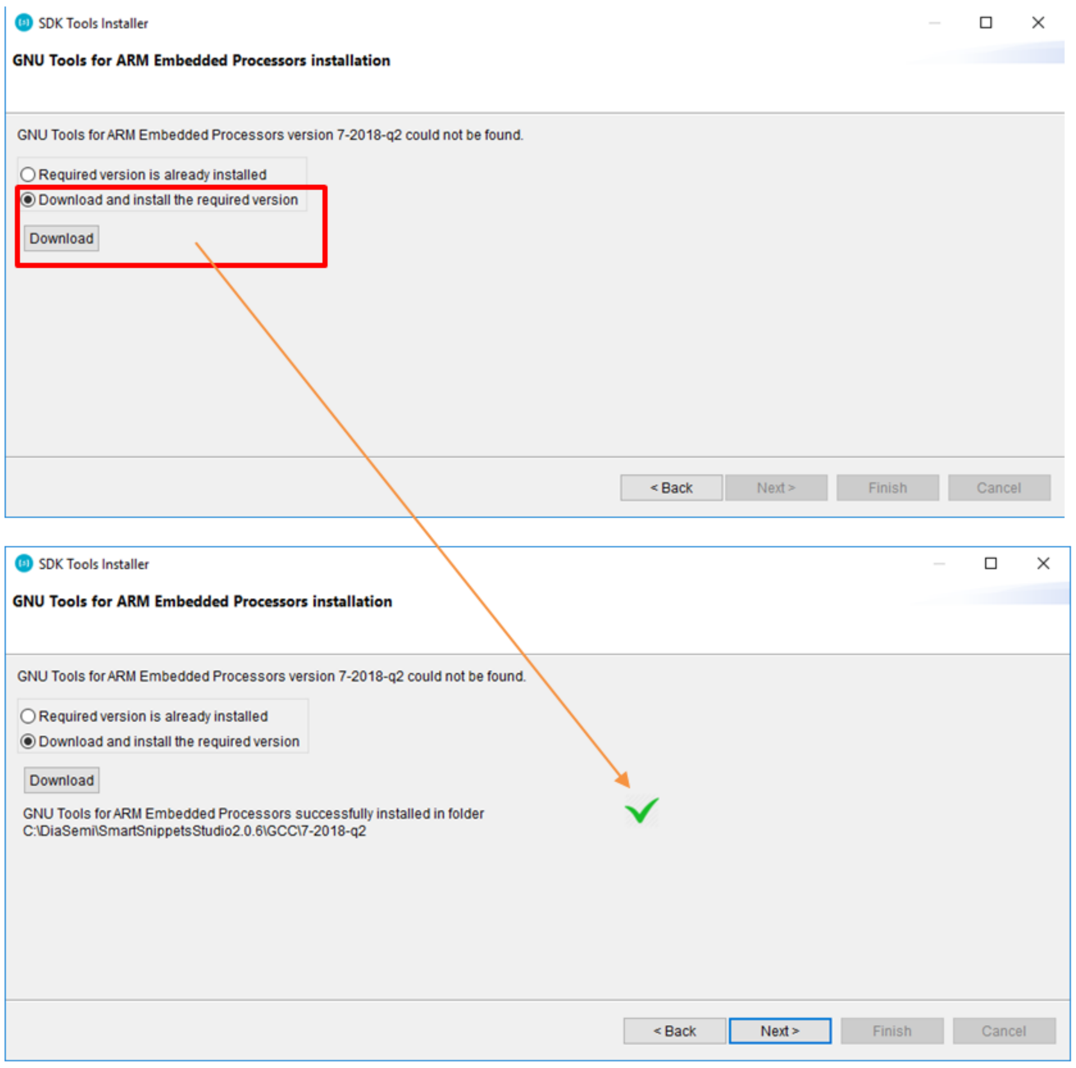
[✓ Required version is already installed]
Download and install the required version

Please specify the installation folder:

C:\Program Files (x86)\SEGGER\SystemView_V242

Figure 8: Set SEGGER Ozone & J-Link Installation Folder

4. The **SDK Tools Installer** will prompt for the installation of GCC and the default version 7-2018-q2 is recommended to download. Select **Download and installed the required version** and click **Download** (Figure 9)**.** After the download is complete the green completion mark is shown and the **Next** may be pressed to continue (Figure 9).


SDK Tools Installer

GNU Tools for ARM Embedded Processors installation

GNU Tools for ARM Embedded Processors version 7-2018-q2 could not be found.

Required version is already installed [✓ Download and install the required version]

GNU Tools for ARM Embedded Processors successfully installed in folder C:\DiaSemi\SmartSnippetsStudio2.0.6\GCC\7-2018-q2

Figure 9: GCC Tools Installation

5. The SmartSnippets^{TM} Studio environment will now appear. Users should click the IDE icon to start the Eclipse CDT IDE (Figure 10).

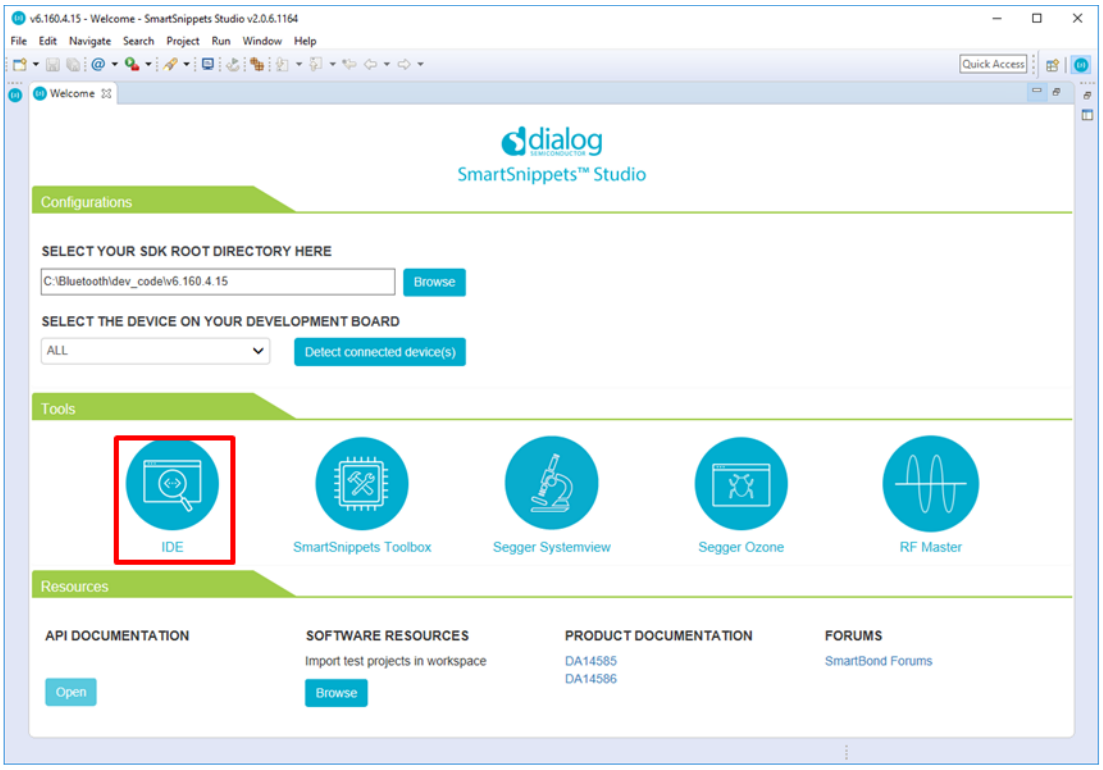
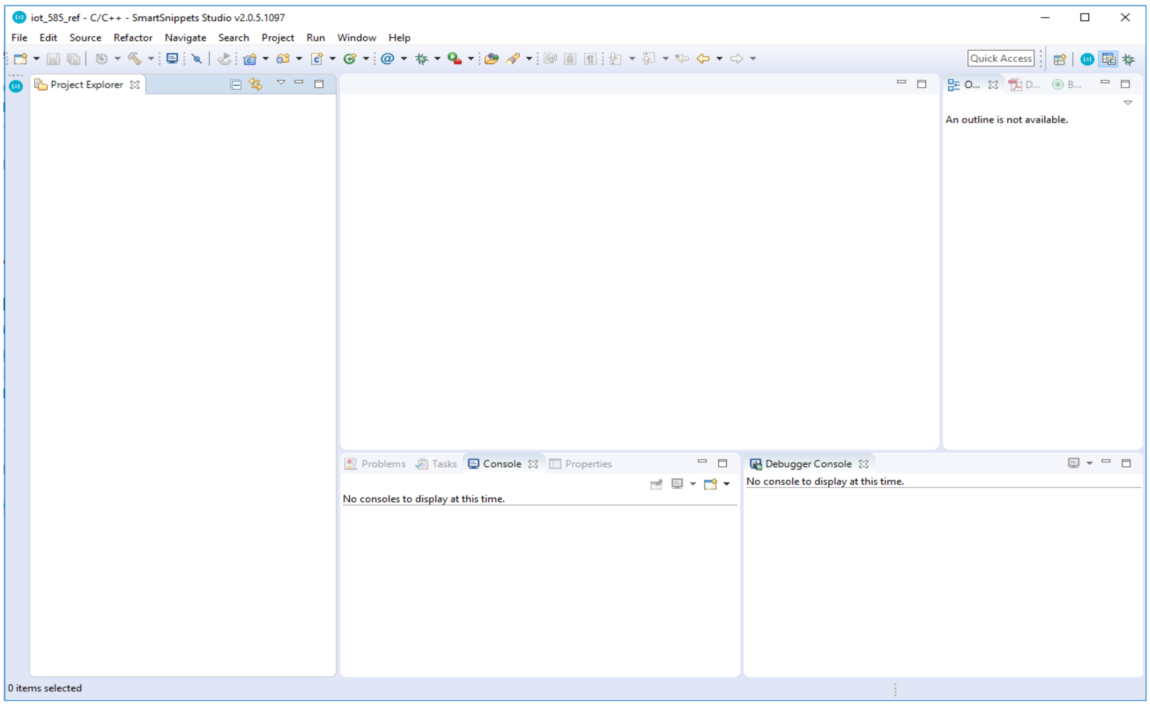

v6.160.4.15 - Welcome - SmartSnippets Studio v2.0.6.1164
File Edit Navigate Search Project Run Window Help
Quick Access
Welcome
dialog
SmartSnippets™ Studio
Configurations
SELECT YOUR SDK ROOT DIRECTORY HERE
C:\Bluetooth\dev_code\v6.160.4.15
Browse
SELECT THE DEVICE ON YOUR DEVELOPMENT BOARD
ALL
Detect connected device(s)
Tools
IDE
SmartSnippets Toolbox
Segger Systemview
Segger Ozone
RF Master
Resources
API DOCUMENTATION
Open
SOFTWARE RESOURCES
Import test projects in workspace
Browse
PRODUCT DOCUMENTATION
DA14585
DA14586
FORUMS
SmartBond Forums

Figure 10: Starting the IDE

6. Now the Eclipse CDT IDE will open (Figure 11). In the left-side window there are no projects listed, so the next step is to insert the project example.


iot_585_ref - C/C++ - SmartSnippets Studio v2.0.5.1097
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Project Explorer
O... D... B...
Problems Tasks Console Properties
Debugger Console
0 items selected

An outline is not available.

No consoles to display at this time.

No console to display at this time.

Figure 11: Eclipse CDT IDE without project

7. To insert an existing project, click **File** > **Import**. The Import window now will open and users should select "**Existing Projects into Workspace**" from the root tree "**General**" (Figure 12) and then click "**Next**".

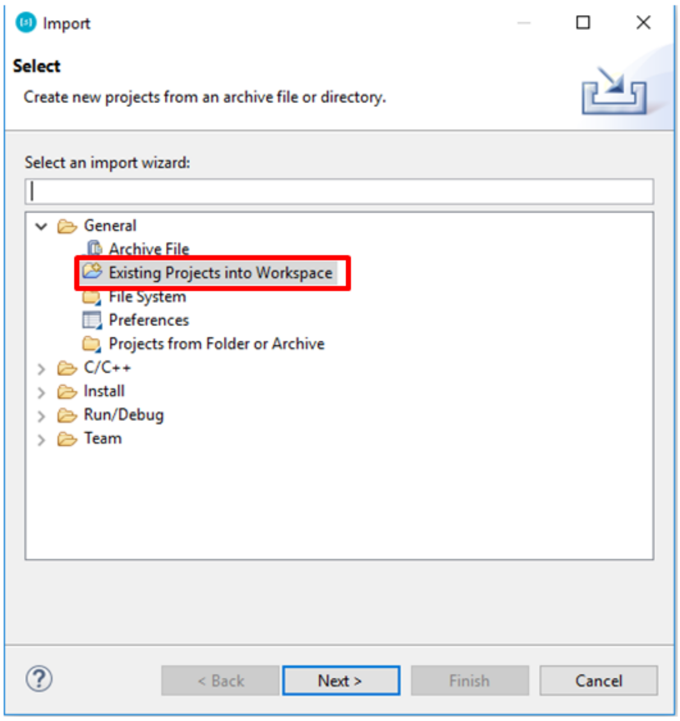


Figure 12: Insert Existing Project into Workspace

8. A window will appear and users should choose "**Browse**" next to "**Select root directory"**, then select the root directory of `v6.160.4.15` (Figure 13), and press **OK**.

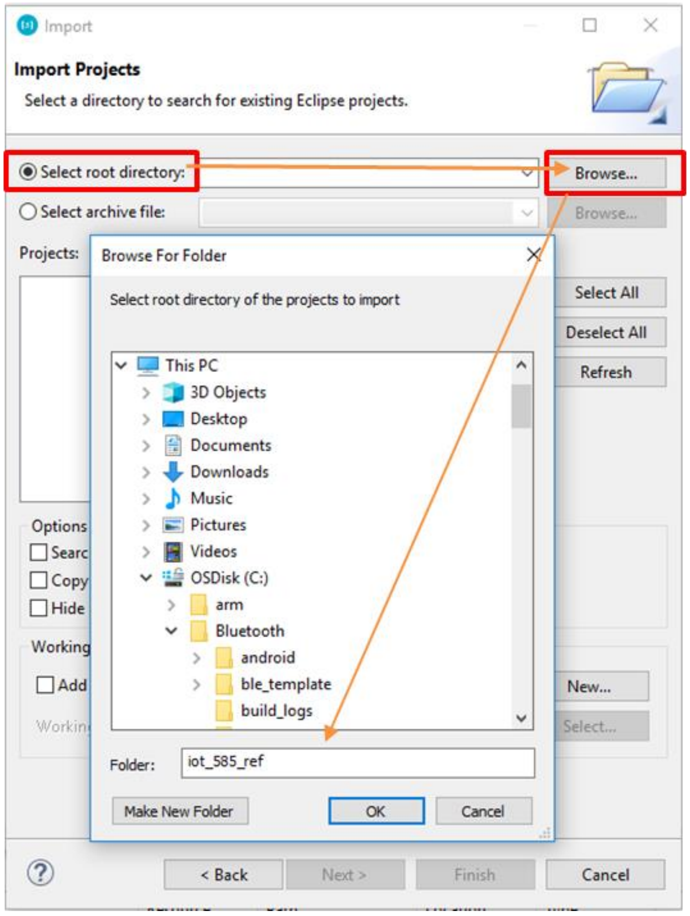
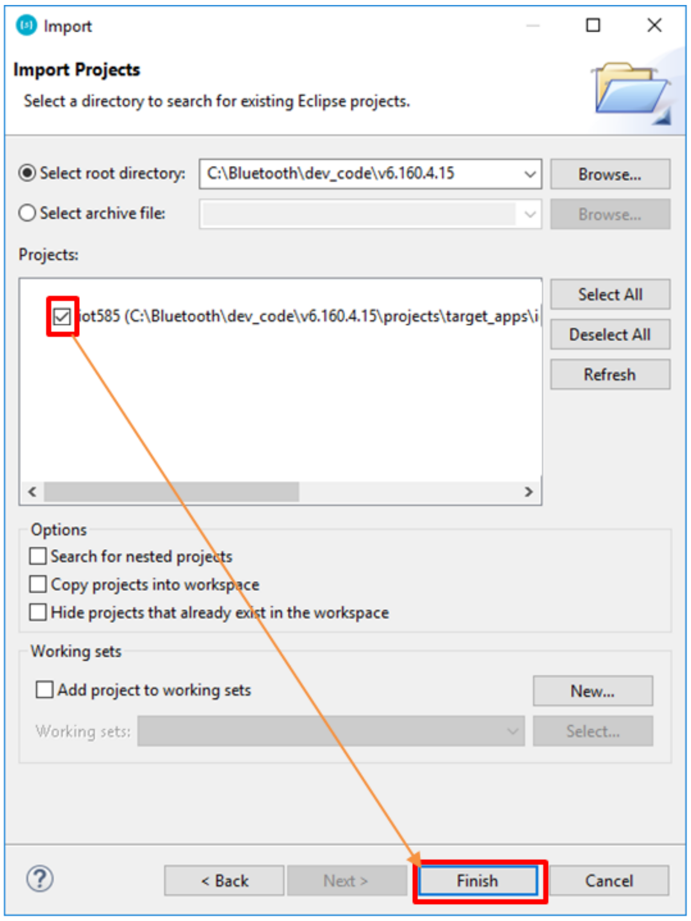


Figure 13: Select Project Root Directory

9. A list of projects compatible with Eclipse will appear, and users should only select the **iot585** project by ticking the selection box (Figure 14) and then press "**Finish**".


Import

Import Projects

Select a directory to search for existing Eclipse projects.

Select root directory: C:\Bluetooth\dev_code\v6.160.4.15

Select archive file:

Projects:

☑ iot585 (C:\Bluetooth\dev_code\v6.160.4.15\projects\target_apps\i

Options

☐ Search for nested projects

☐ Copy projects into workspace

☐ Hide projects that already exist in the workspace

Working sets

☐ Add project to working sets

Working sets:

Figure 14: Selecting the iot585 Project

10. After Eclipse imports the project, the **iot585** project will appear in the Project Explorer window (Figure 15).

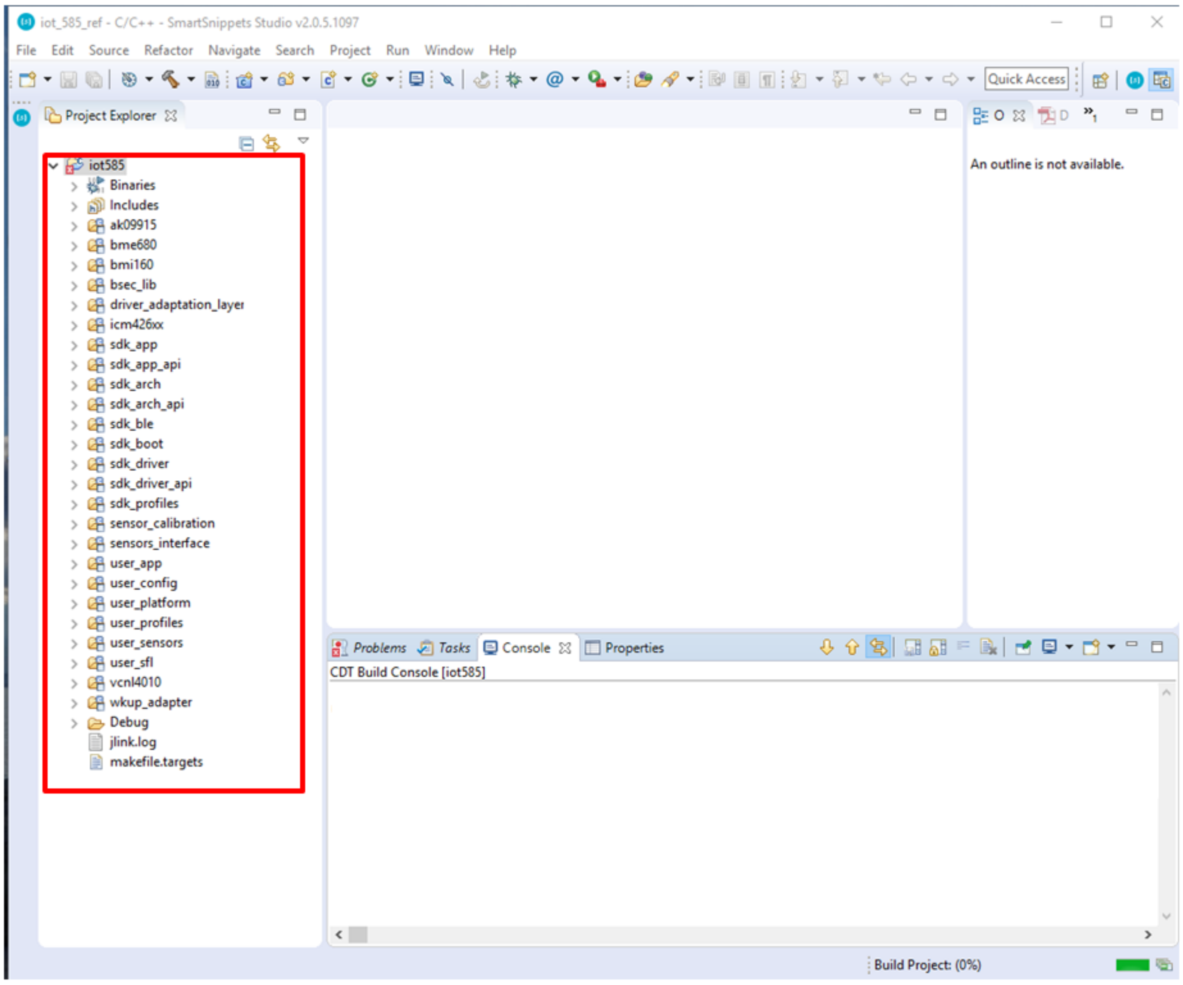
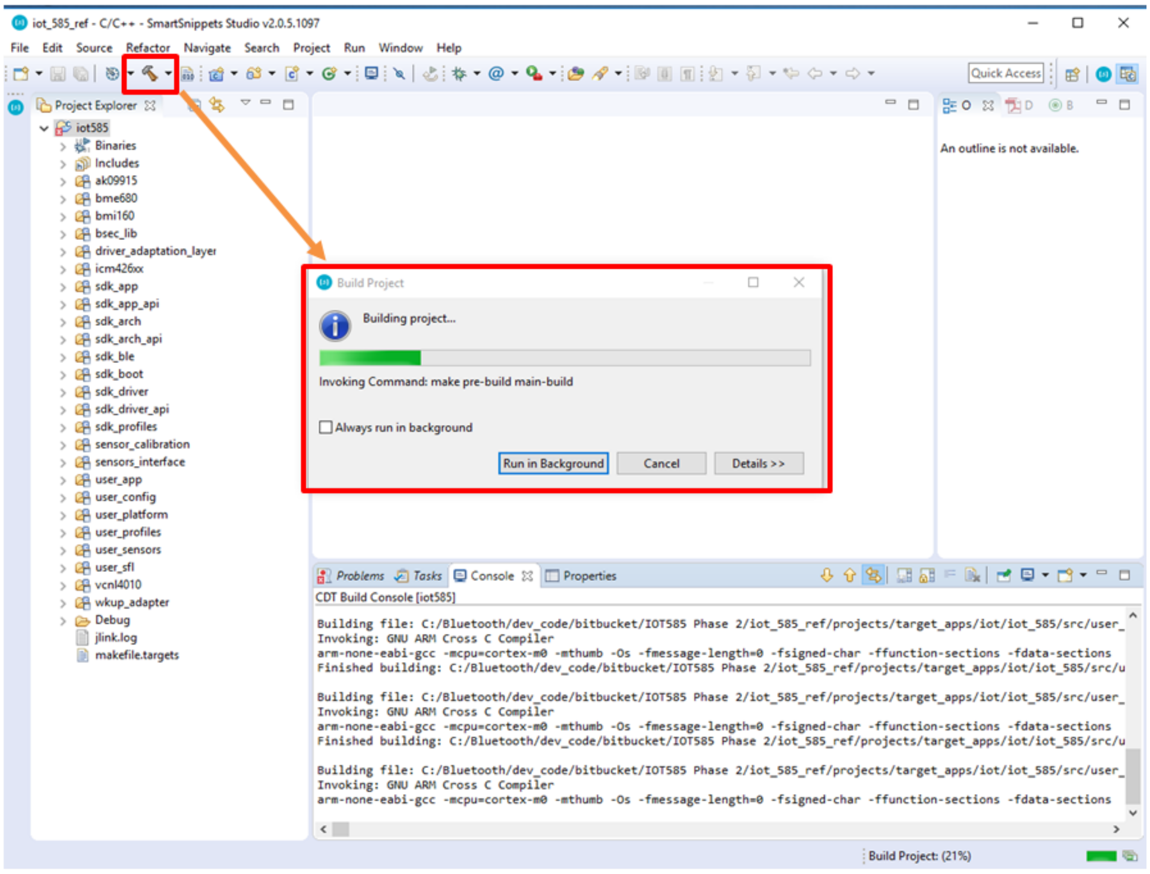


Figure 15: iot585 Project Imported

6 Building the Project

After the abovementioned steps, an existing project is imported into Eclipse CDT environment but it is not built. To build the project, click the "build" icon on the menu bar (Figure 16). The progress of building the project is shown in the bottom console window.



Invoking Command: make pre-build main-build

CDT Build Console [iot585]

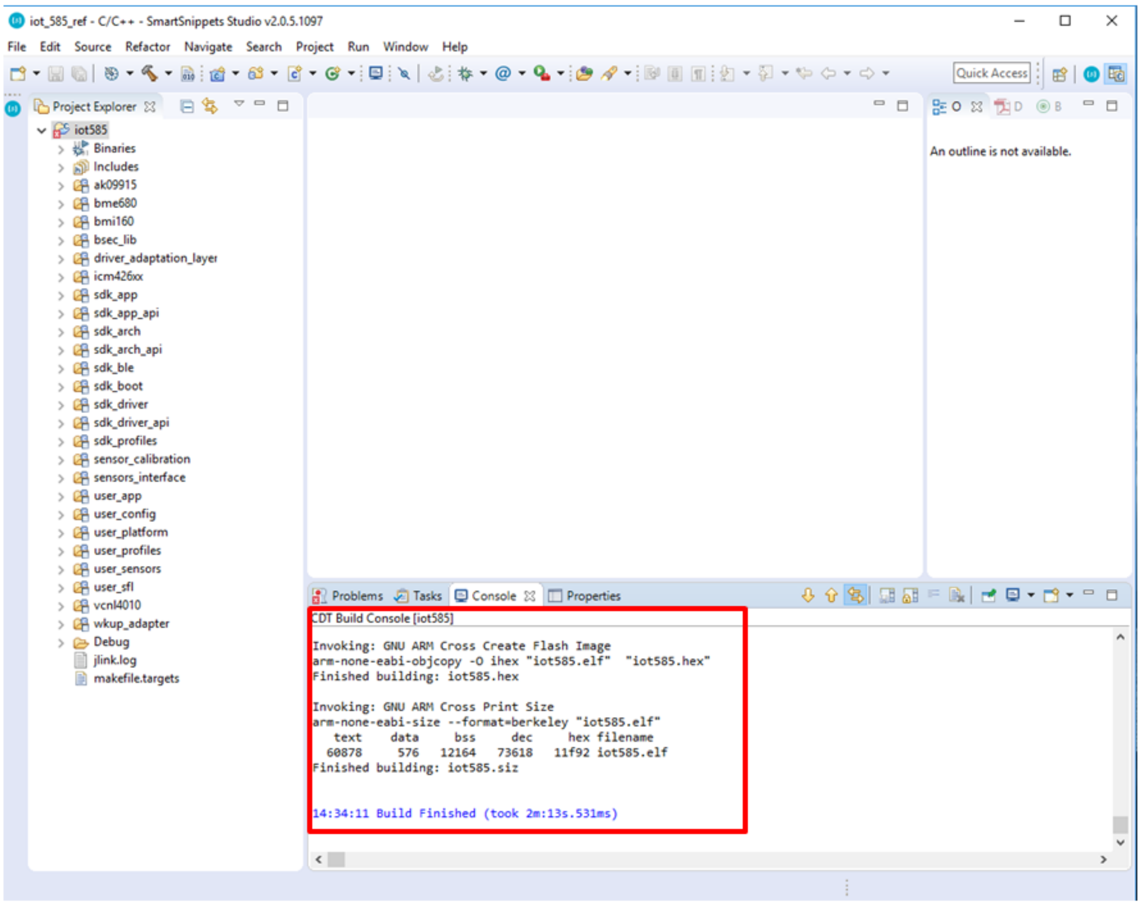
```
Building file: C:/Bluetooth/dev_code/bitbucket/IOT585 Phase 2/iot_585_ref/projects/target_apps/iot/iot_585/src/user_
Invoking: GNU ARM Cross C Compiler
arm-none-eabi-gcc -mcpu=cortex-m0 -mthumb -Os -fmessage-length=0 -fsigned-char -ffunction-sections -fdata-sections
Finished building: C:/Bluetooth/dev_code/bitbucket/IOT585 Phase 2/iot_585_ref/projects/target_apps/iot/iot_585/src/u

Building file: C:/Bluetooth/dev_code/bitbucket/IOT585 Phase 2/iot_585_ref/projects/target_apps/iot/iot_585/src/user_
Invoking: GNU ARM Cross C Compiler
arm-none-eabi-gcc -mcpu=cortex-m0 -mthumb -Os -fmessage-length=0 -fsigned-char -ffunction-sections -fdata-sections
Finished building: C:/Bluetooth/dev_code/bitbucket/IOT585 Phase 2/iot_585_ref/projects/target_apps/iot/iot_585/src/u

Building file: C:/Bluetooth/dev_code/bitbucket/IOT585 Phase 2/iot_585_ref/projects/target_apps/iot/iot_585/src/user_
Invoking: GNU ARM Cross C Compiler
arm-none-eabi-gcc -mcpu=cortex-m0 -mthumb -Os -fmessage-length=0 -fsigned-char -ffunction-sections -fdata-sections
```

Figure 16: Build the Project

At the end of the building procedure (Figure 17), output files `iot585.elf` and `iot585.hex` are generated. These files are located inside the folder `projects\target_apps\iot\iot_585\Eclipse\Debug`. The `iot585.elf` is the linker output and can be used for debugging. The `iot585.hex` is the stripped binary and will be used to create the multi-image file that can be burned directly into flash.



```
CDT Build Console [iot585]

Invoking: GNU ARM Cross Create Flash Image
arm-none-eabi-objcopy -O ihex "iot585.elf"  "iot585.hex"
Finished building: iot585.hex

Invoking: GNU ARM Cross Print Size
arm-none-eabi-size --format=berkeley "iot585.elf"
   text    data     bss     dec     hex filename
  60878     576   12164   73618   11f92 iot585.elf
Finished building: iot585.siz



14:34:11 Build Finished (took 2m:13s.531ms)
```

Figure 17: Building Finished

7 Debugging

For the purpose of debugging, click the **Debug** icon and then select the **RAM_DA14585** debug configuration (Figure 18).

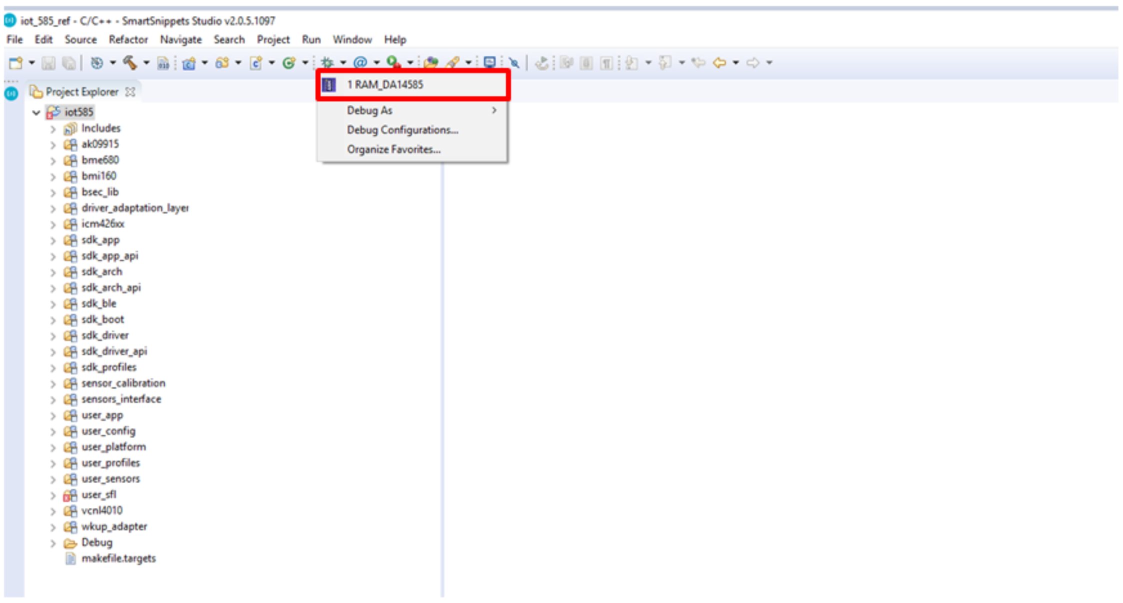
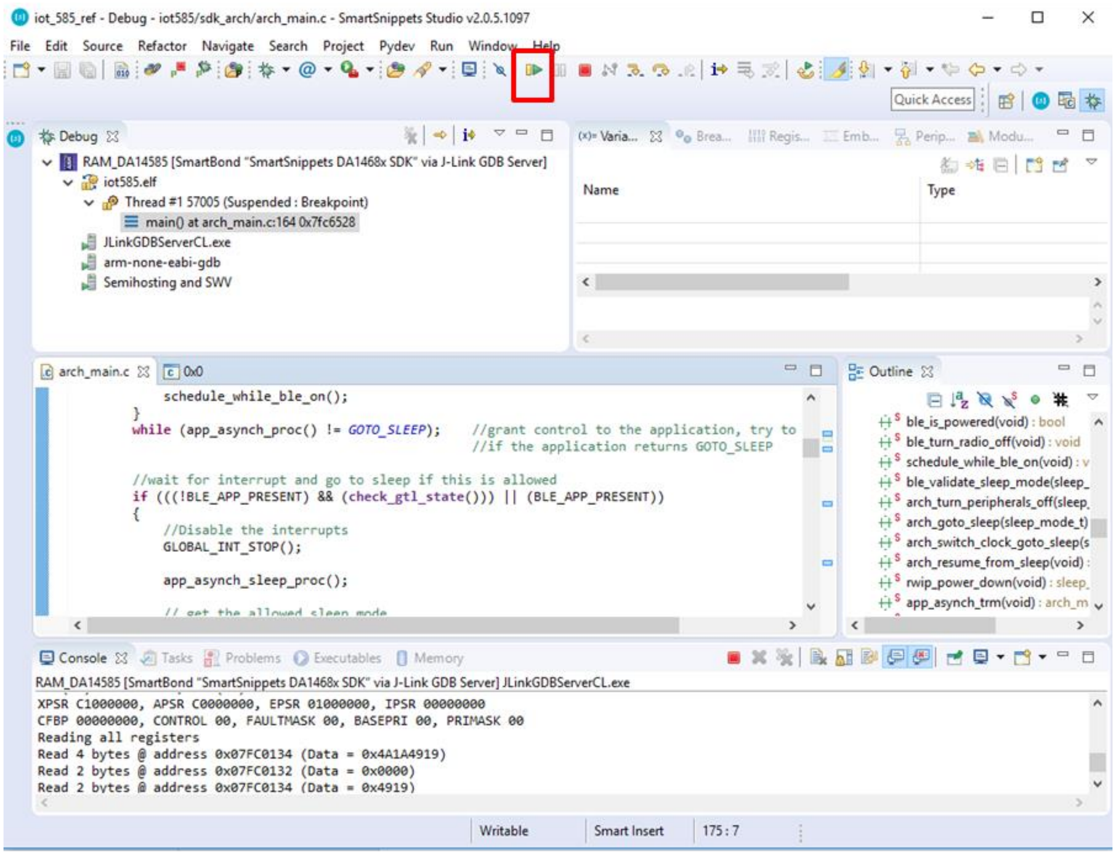


Figure 18: Start Debugging

Accept it when Eclipse asks to open the Debug perspective and then the Debug view will open. Press the **Resume** button to start running the program (Figure 19).


iot_585_ref - Debug - iot585/sdk_arch/arch_main.c - SmartSnippets Studio v2.0.5.1097
File Edit Source Refactor Navigate Search Project Pydev Run Window Help
Quick Access
Debug
Varia...
Brea...
Regis...
Emb...
Perip...
Modu...
Name
Type
arch_main.c
0x0
Outline
Console
Tasks
Problems
Executables
Memory
Writable
Smart Insert
175 : 7

```
RAM_DA14585 [SmartBond "SmartSnippets DA1468x SDK" via J-Link GDB Server]
    iot585.elf
        Thread #1 57005 (Suspended : Breakpoint)
            main() at arch_main.c:164 0x7fc6528
    JLinkGDBServerCL.exe
    arm-none-eabi-gdb
    Semihosting and SWV
```

```
        schedule_while_ble_on();
    }
    while (app_asynch_proc() != GOTO_SLEEP);    //grant control to the application, try to
                                                //if the application returns GOTO_SLEEP

    //wait for interrupt and go to sleep if this is allowed
    if (((!BLE_APP_PRESENT) && (check_gtl_state())) || (BLE_APP_PRESENT))
    {
        //Disable the interrupts
        GLOBAL_INT_STOP();

        app_asynch_sleep_proc();

        // get the allowed sleep mode
```

```
ble_is_powered(void) : bool
ble_turn_radio_off(void) : void
schedule_while_ble_on(void) : v
ble_validate_sleep_mode(sleep_
arch_turn_peripherals_off(sleep_
arch_goto_sleep(sleep_mode_t)
arch_switch_clock_goto_sleep(s
arch_resume_from_sleep(void) :
rwip_power_down(void) : sleep_
app_asynch_trm(void) : arch_m
```

```
RAM_DA14585 [SmartBond "SmartSnippets DA1468x SDK" via J-Link GDB Server] JLinkGDBServerCL.exe
XPSR C1000000, APSR C0000000, EPSR 01000000, IPSR 00000000
CFBP 00000000, CONTROL 00, FAULTMASK 00, BASEPRI 00, PRIMASK 00
Reading all registers
Read 4 bytes @ address 0x07FC0134 (Data = 0x4A1A4919)
Read 2 bytes @ address 0x07FC0132 (Data = 0x0000)
Read 2 bytes @ address 0x07FC0134 (Data = 0x4919)
```

Figure 19: Debug Perspective

8 Creating the Multi-Image

Detailed information on the provided scripts is included in *Appendix C* of [1]. Users should use the script `make_image_iot.bat` that is placed in "`..\utilities\mkimage_utils_scripts`". The `.hex` extension is added automatically and the command should be "`make_image_iot.bat io585`". The output file `multi_iot585.bin` can be used by the Flash Programmer of SmartSnippets™ Toolbox. Detailed instructions of how to burn the multi-image `multi_iot585.bin` using SmartSnippets™ Toolbox are included in *Appendix D.2* of [1].

IMPORTANT NOTE
Because a secondary bootloader is used (*Appendix C* of [1]) that selects the most recent/valid image between two images, `io585.hex` cannot be burnt directly into flash, but the multi-image should be created instead. This functionality is necessary to support firmware updates over the air (SUOTA).

Revision History

Revision	Date	Description
1.1	17-Jan-2022	Updated logo, disclaimer, copyright.
1.0	10-Oct-2018	Initial version.

Status Definitions

Status	Definition
DRAFT	The content of this document is under review and subject to formal approval, which may result in modifications or additions.
APPROVED or unmarked	The content of this document has been approved for publication.