InvenSense

Using Eclipse IDE with J-Link debugger

Application Note for
ICM-306XX embedded

Table of Contents

Table of Figures

1. ABSTRACT

This application note describes how to set up a software-based debugging environment for the ICM306XX eMD shield with SensorStudio using Eclipse Integrated Development Environment (IDE) and the SEGGER J-Link debugger.

2. INTRODUCTION

The ICM306XX eMD shield used with SensorStudio is based on GCC development tools. The purpose of this solution is to provide an easy to use environment enabling sensors management and algorithms development. The ICM306XX eMD running through SensorStudio solution was created as an advanced sensor hub, easy to modify and upgradable for developers. The ICM306XX eMD and SensorStudio tool provide a full software solution allowing the end-user to develop new features and applications in order to program their own logic.

This document describes which software packages are necessary, and provides installation and configuration instructions for each.

Section 2 lists the software packages and hardware required. Installation process for the software packages are provided in Section 3. The next section describes how to use the Eclipse IDE through SensorStudio to build the project and run a debugging session.

3. PREREQUISITES

3.1. PLATFORM

The setup has been tested on a Windows 7 platform. Linux platforms are not supported in this application note.

3.2. HARDWARE

The following hardware is required:

- Micro-USB to USB cable
- Arduino Zero board (connected to PC through USB Programming port)
- ICM306XX-eMD shield
- SEGGER J-Link Debugger
- SWD cable to connect to ICM306XX eMD shield through CN7 connector (for more information, please refer to ICM306XX eMD shield datasheet).
  The connector is sold with a white full pin (pin 7) to be use as keyed, not needed for the ICM306XX shield, you can safely remove it.

For debugging session, please be careful to jumper’s configuration.

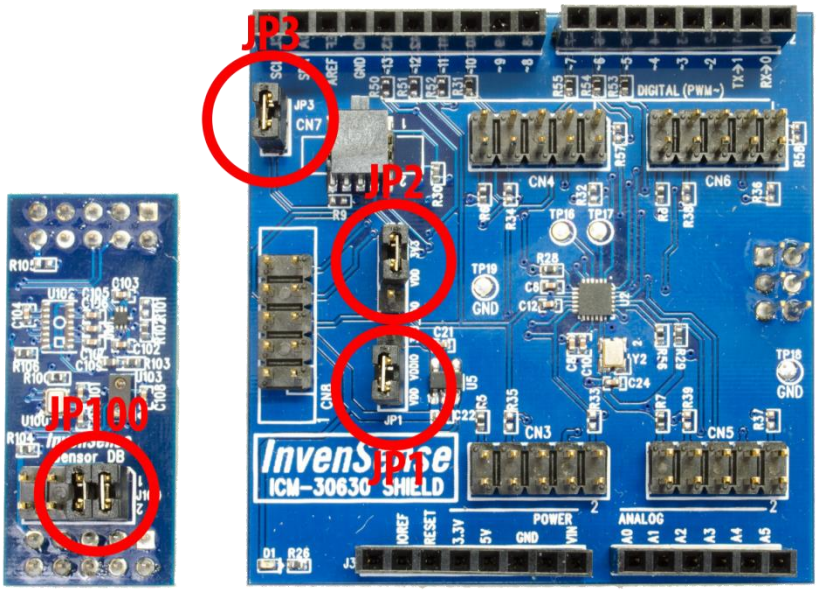

Figure 1 – ICM30630 embedded jumper's configuration for debug

On ICM306XX shield:

- JP1 connects VDD to VDDIO
- JP2 connects VDD to 3V3
- JP3 is open

On Sensor DB:

- J100 connects pin 1 to 2 and pin 3 to 4

The Sensor DB is optional and adds available external sensors: magnetometer, proximity and pressure sensor.

3.3. SOFTWARE

This section describes the software packages that are required for going through this application note. The below packages might be used as well as newer version available after this document was written.

- The Tool Suite: SensorStudio

- The IDE : Eclipse and the following components
  - Java Runtime Environment – JRE
  - Eclipse Mars (the oldest version supporting GNU ARM Eclipse plug-ins is Eclipse Kepler 4.4 SR2) GNU Toolchain for ARM Embedded Processors linaro-4.7-2013 included in the SensorStudio tool suite package
  - GNU ARM Eclipse plug-ins

- The SEGGER J-Link software

4. SOFTWARE INSTALLATION INSTRUCTIONS

4.1. J-LINK DEBUGGER

Download the J-Link installer available here https://www.segger.com/jlink-software.html . Unzip the setup folder and run **Setup_JLink_xxxx.exe**, where "xxxx" is the software version. The version package tested during writing this document is v5.02c.

Figure 2 - Run the J-Link Flash programmer installer

Install also USB Driver for J-Link. The default path for the J-Link software is C:\Program Files (x86)\SEGGER\JLink_V502c. Click on Next to install the J-Link Debugger.

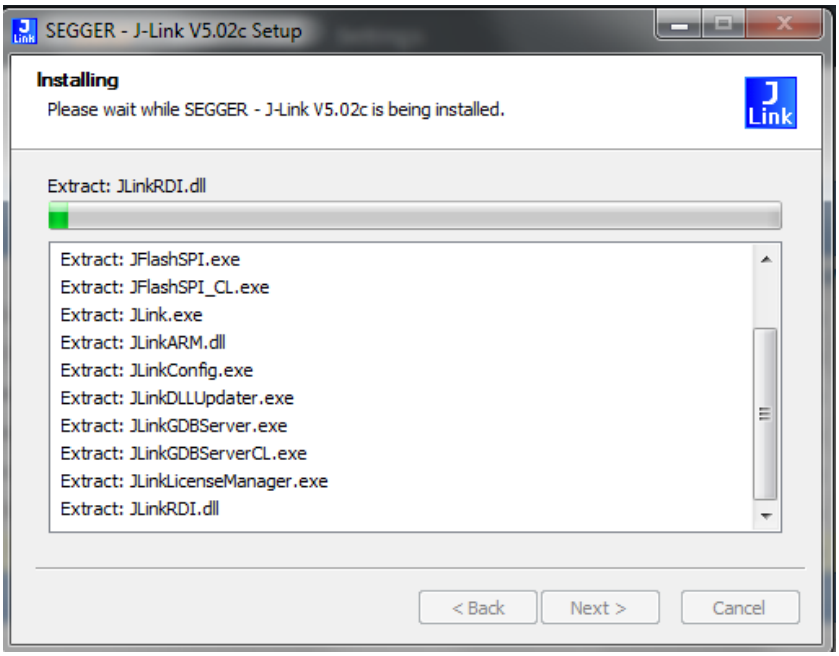

Figure 3 - Installing J-Link Flash programmer

4.2. ECLIPSE IDE

Preamble: The Eclipse Integrated Development Environment (IDE) is dependent on the Java Runtime Environment (JRE) being installed on the machine. Both must be downloading for the same platform, for example the Eclipse IDE x64 version needs the JRE x64 version. Make sure the correct JRE version is installed on your system before running Eclipse installer. Otherwise you can download it here: http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html .
Download Eclipse IDE from the eclipse website https://eclipse.org/downloads/ . Please select "**Eclipse IDE for C/C++ Developers"** that contains C/C+ Development Tooling (CDT) needed.
Unzip download package, and run **eclipse.exe**. Eclipse will ask for a folder to use for workspace location when started.

Please follow these instructions to install Eclipse needed components for GNU ARM Eclipse plug-ins:

1. Download GNU ARM Eclipse plug-ins on website http://sourceforge.net/projects/gnuarmeclipse/?source=typ_redirect
2. Install GNU ARM Eclipse plug-ins; navigate to Help->Install New Software…

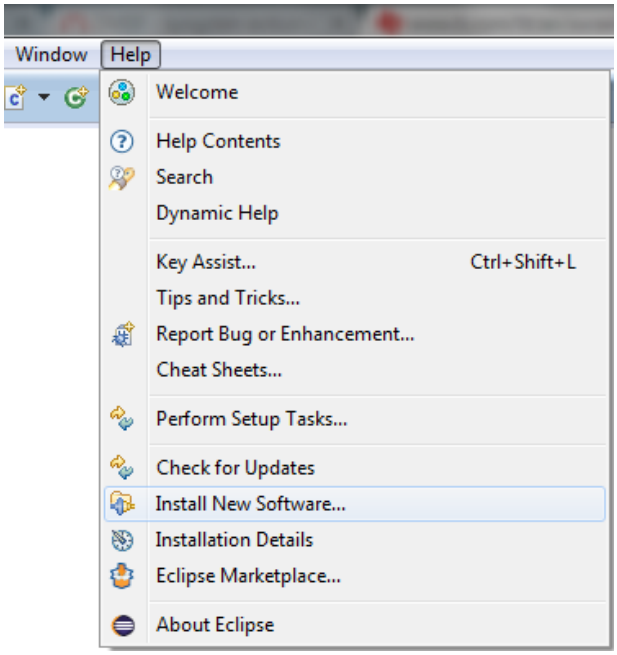

Figure 4 - Install New Software plug-ins

3. Click to Add... to select GNU ARM Eclipse plug-ins. Select the Archive to directly load the zip folder.

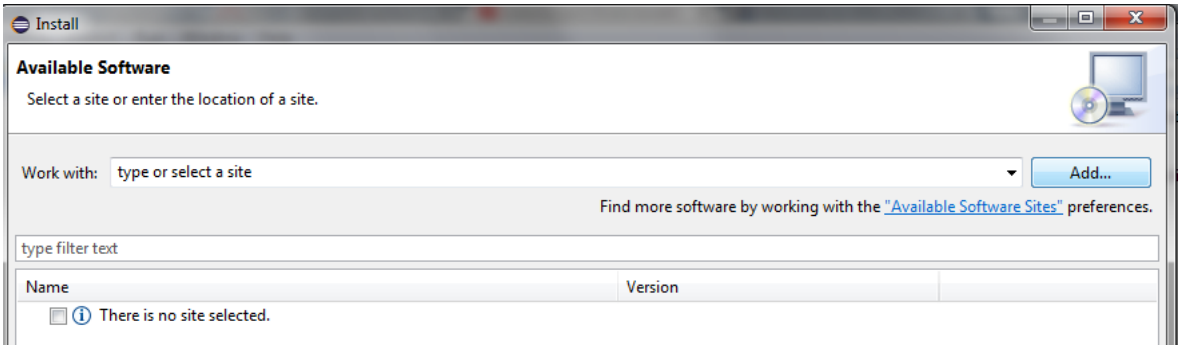

Figure 5 - Add the GNU ARM Eclipse plug-ins archive

4. Select all items available

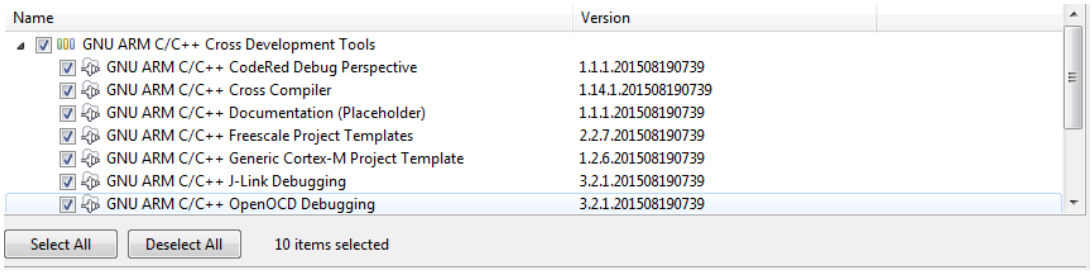

Figure 6 - Select all GNU ARM Eclipse plug-ins

5. Finish the installation

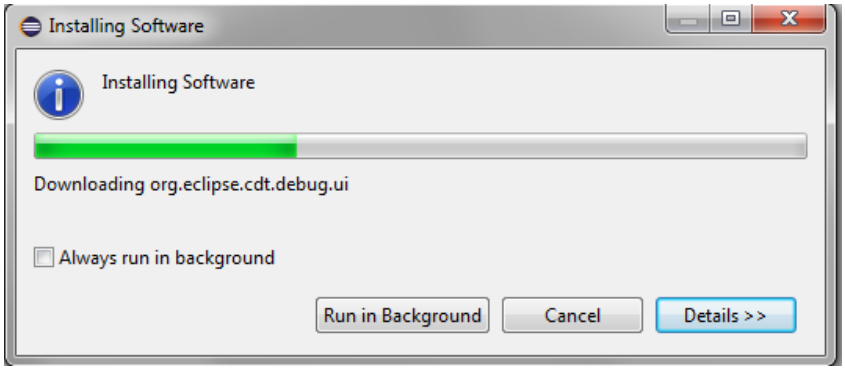

Figure 7 - Installing GNU ARM Eclipse plug-ins

You will need to restart Eclipse to taking in account the plug-ins installation.
Once these package installations completed, you can verify that the CDT and the GNU ARM Eclipse plug-ins are correctly installed. Navigate to Help-> Installation Details. Make sure that the C/C++ Development Platform, The GNU ARM C/C++ Cross Compiler and the GNU ARM C/C++ J-Link Debugging are installed.

4.1. SENSORSTUDIO

Please refer to SensorStudio documentation.

5. BUILD AND DEBUG PROJECT

5.1. BUILD THE PROJECT

To build the FireFly sample project you generated with SensorStudio, click on “Build” button on device toolbar.

Figure 8 – Build code button

5.2. RUNNING THE PROJECT FROM DEBUGGER

Once you start your flow on SensorStudio, you can start the debugging session on Eclipse IDE. Your Eclipse project is automatically generated and available. Before running the debug session, make sure you connect the J-Link SWD connector to the ICM306XX eMD shield. Then click on “Debug” button on SensorStudio.

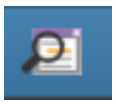

Figure 9 - Debug firmware button

The first time you launch “Debug”, SensorStudio asks for the location of Eclipse and J-Link debugger:

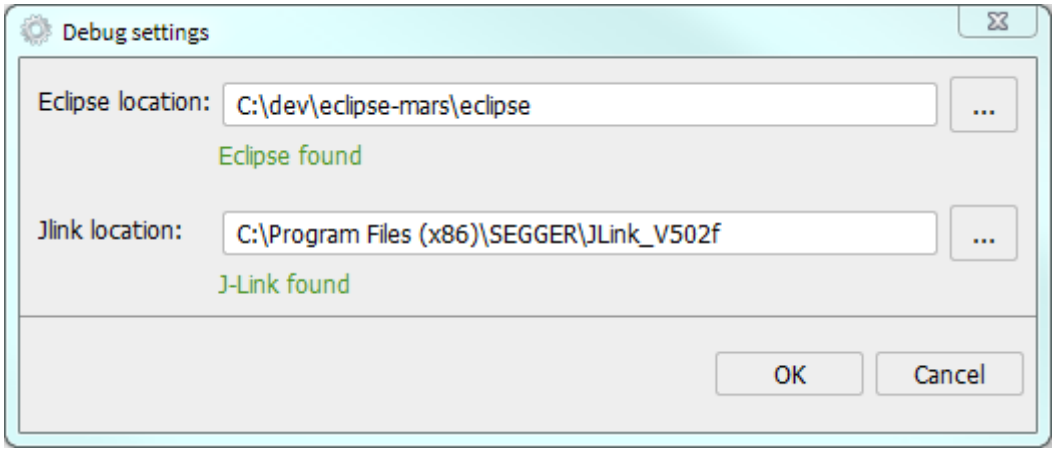

Figure 10 - Debug settings

These settings can be modified at any time via Device -> Debug settings… menu:

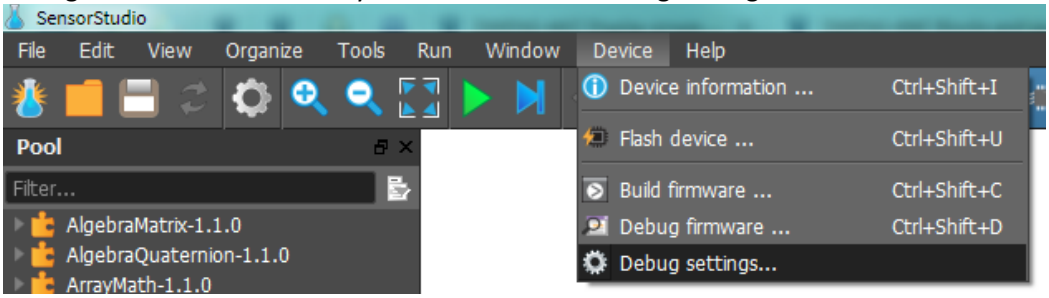

Figure 11 - Accessing Debug settings

Once the settings set, the Eclipse IDE is automatically launched (this can take up to a minute on the first launch). At the first time you start Eclipse IDE, there is a Welcome page that you can close. Now you have the C/C++ Project view, with your project files available.

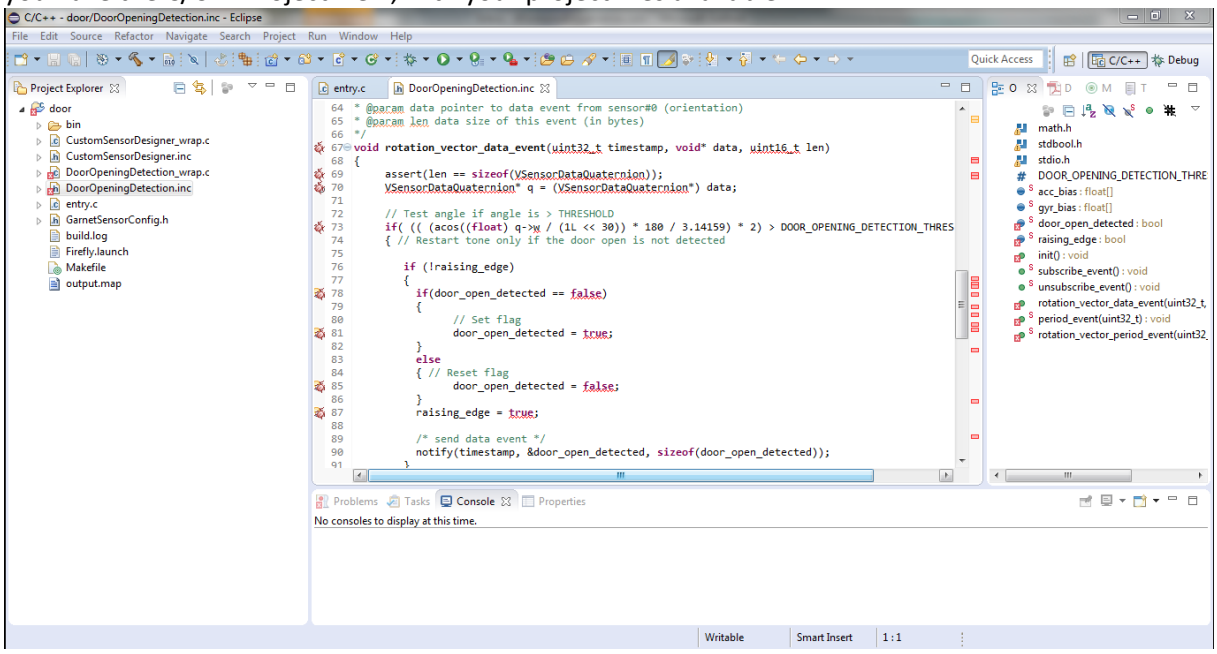

Figure 12 – Eclipse IDE project view

The debug session is already configured for the project in the FireFly debug configuration. The debugger connects to the target through J-Link GDB server on the running target. To access the predefined debug configuration you have to select it the first time you start the debugging.
Click Run->Debug Configuration… On the left side of the window, select GDB SEGGER J-Link Debugging -> **Firefly-*My-Project***, then **Debug**.

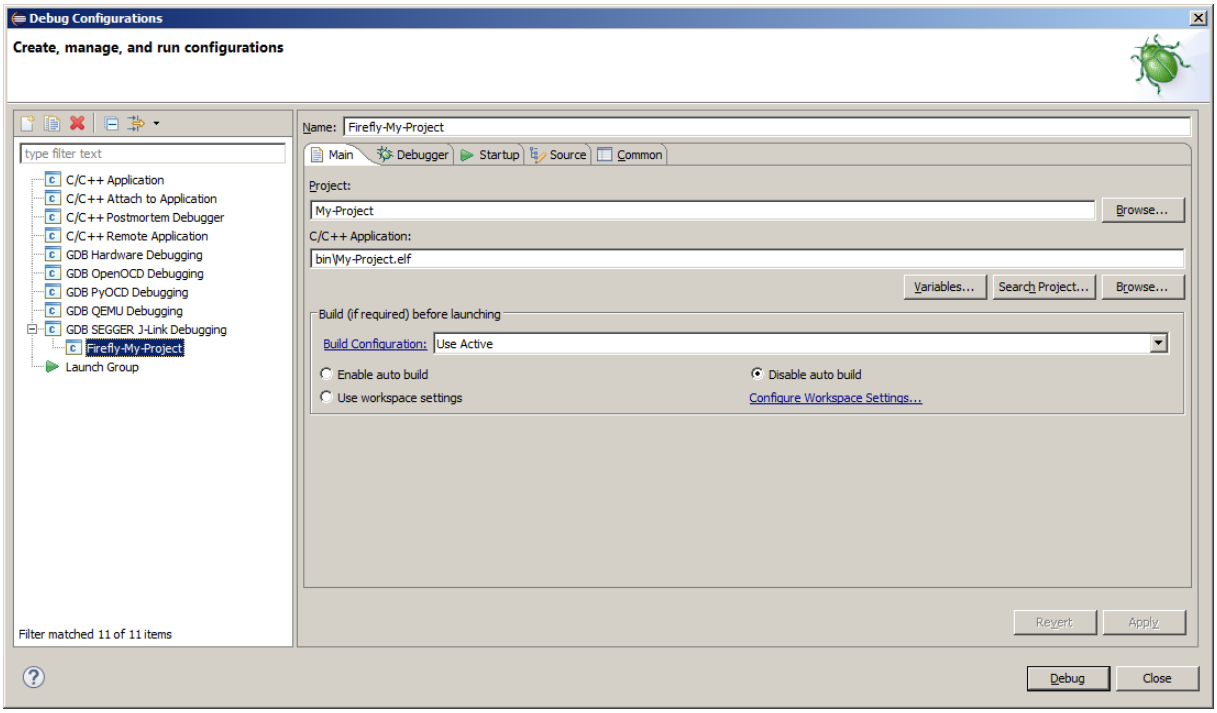

Figure 13 - Select the Firefly debug configuration

Now, you can see the Eclipse Debug perspective. The execution of the application can now be controlled from the J-Link debugger in Eclipse IDE. Press Suspend/Resume button to stop/start running the application, press Terminate button to close the debug session. You can also put breakpoints in your code in double-click in code margin.

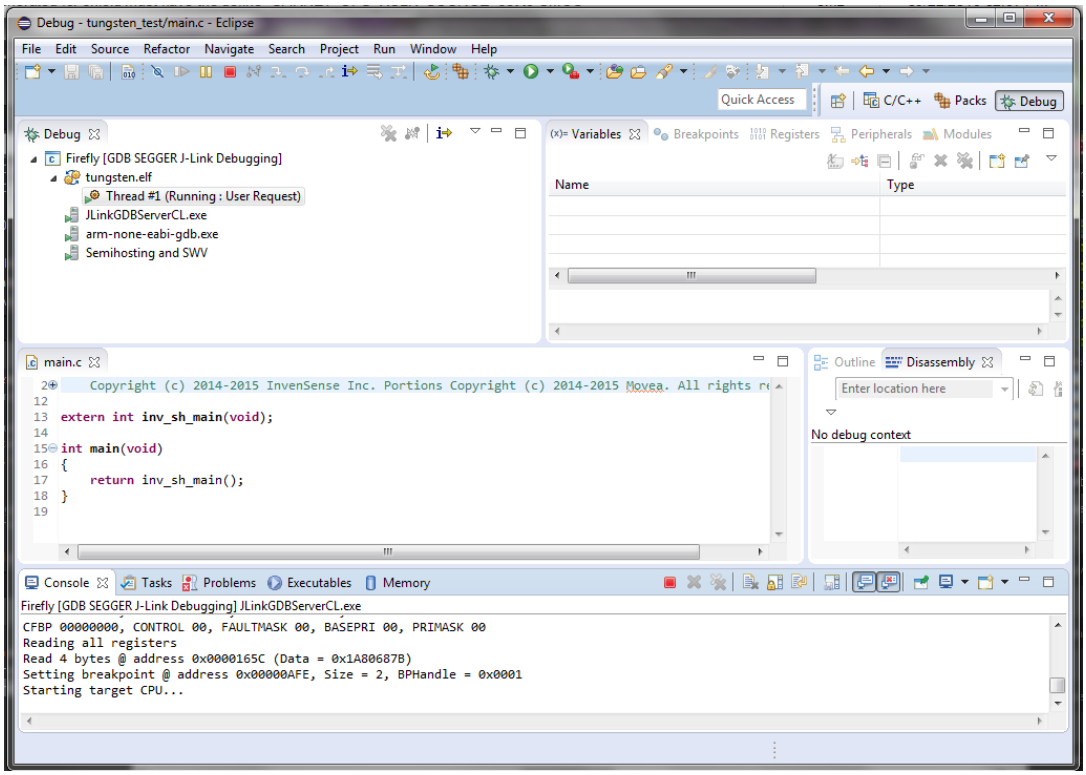

Figure 14 - Running the project from debugger