Getting started with STM32 Nucleo board software
development tools

Introduction

The STM32 Nucleo board is a low-cost and easy-to-use development platform used to quickly evaluate and start a development with an STM32 in 32-pin package, 64-pin package and 144-pin package.
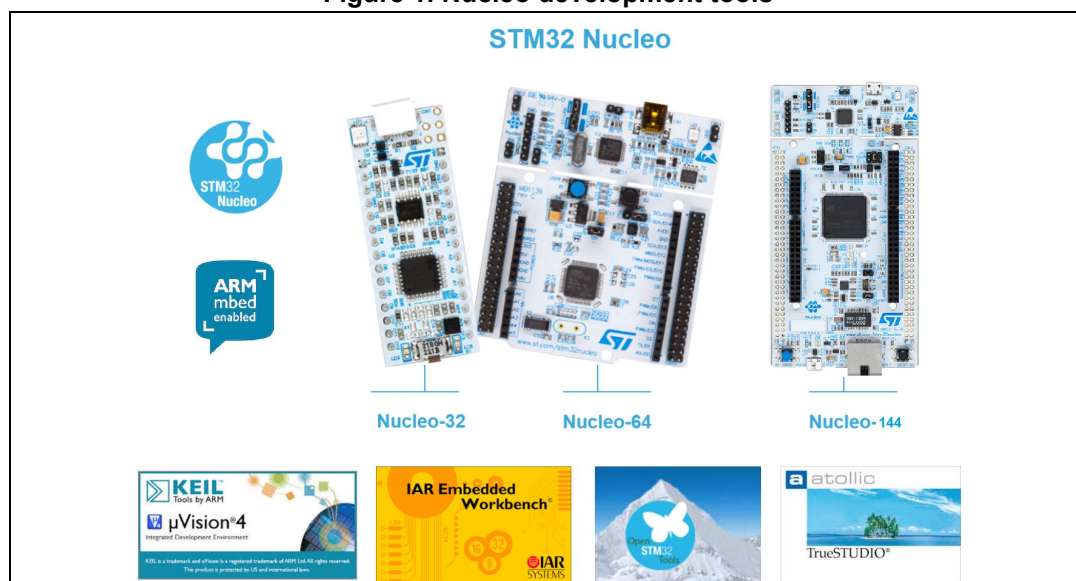
This document provides guidelines to beginners on how to build and run a sample application and allows them to build and debug their application. It has the following structure:

- The first chapter presents software and hardware requirements (some toolchains supporting the STM32 families, ST-LINK/V2-1 installation and firmware package presentation).

- The second chapter provides step by step guideline on how to build and debug an application using some toolchains:
    - IAR Embedded Workbench® for ARM® (EWARM) by IAR systems®
    - Microcontroller Development Kit for ARM® (MDK-ARM) by Keil®
    - TrueSTUDIO® by Atollic®
    - System Workbench for STM32 (SW4STM32) by AC6

This manual does not cover all the topics relevant to software development environments, but it demonstrates the first basic steps necessary to get started with the compilers/debuggers and offers links to the documents needed to fully understand every single step.

All boards belonging to the STM32 MCU Nucleo line are supported.

Figure 1. Nucleo development tools

Contents

List of figures

1 System requirements

Before starting the user should:

1. Install the preferred Integrated Development Environment (IDE).

2. ST-LINK/V2-1 driver will be installed automatically. In case of problem, the user can proceed with manual installation of the driver, from toolchains install directory (further details are available in *Section 3*).

3. Download the STM32 Nucleo firmware from the www.st.com/stm32nucleo webpage.

4. Establish the connection with the STM32 Nucleo board, by connecting CN1 of the Nucleo board to the USB port of the PC (*Figure 2*,*Figure 3* and *Figure 4* show the connector CN1 inside the red circle).

The above steps will be detailed in the coming sections.
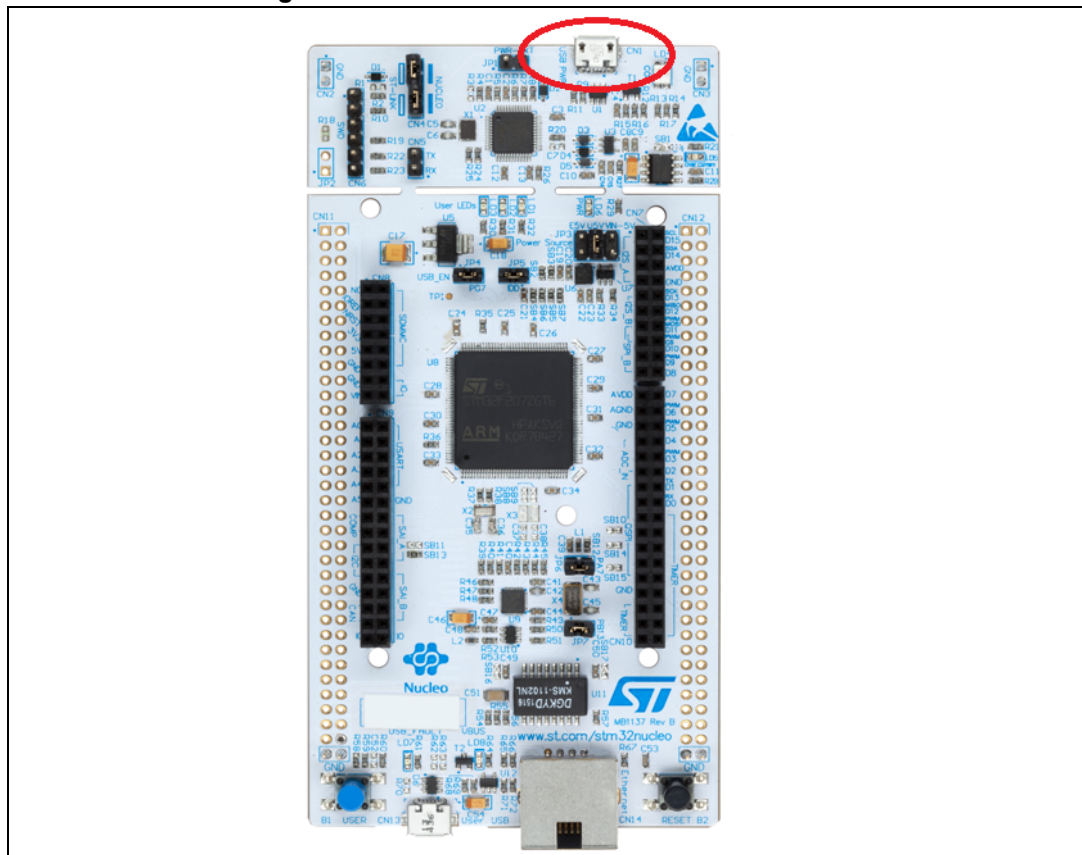
Figure 2. Connector CN1 for Nucleo 144 board

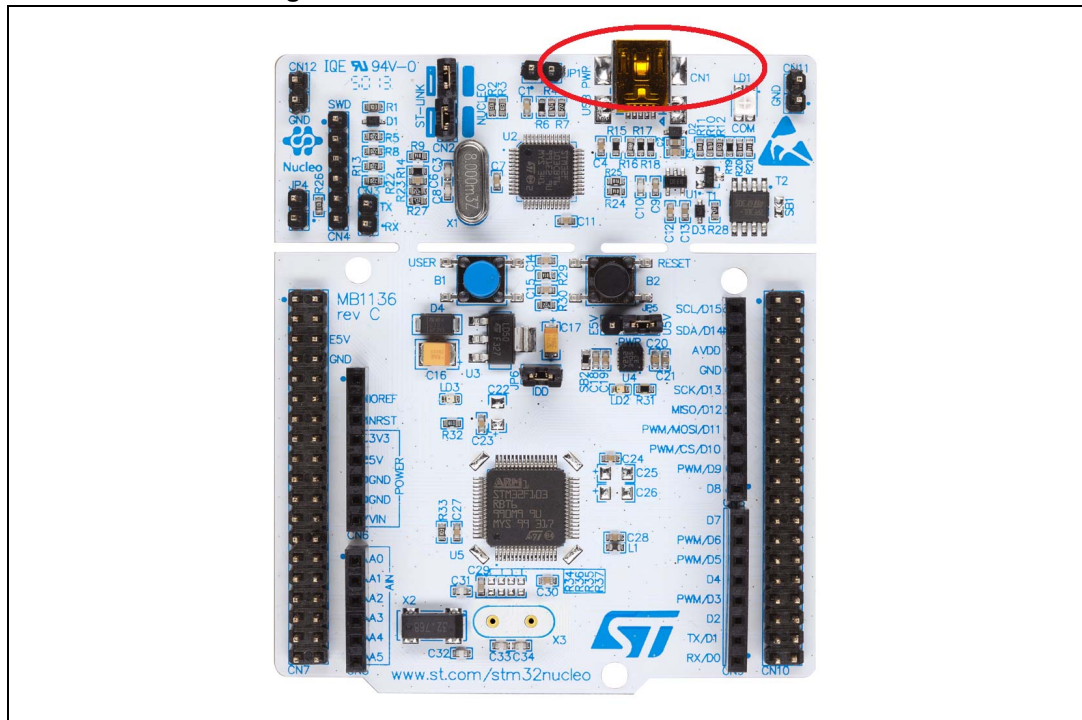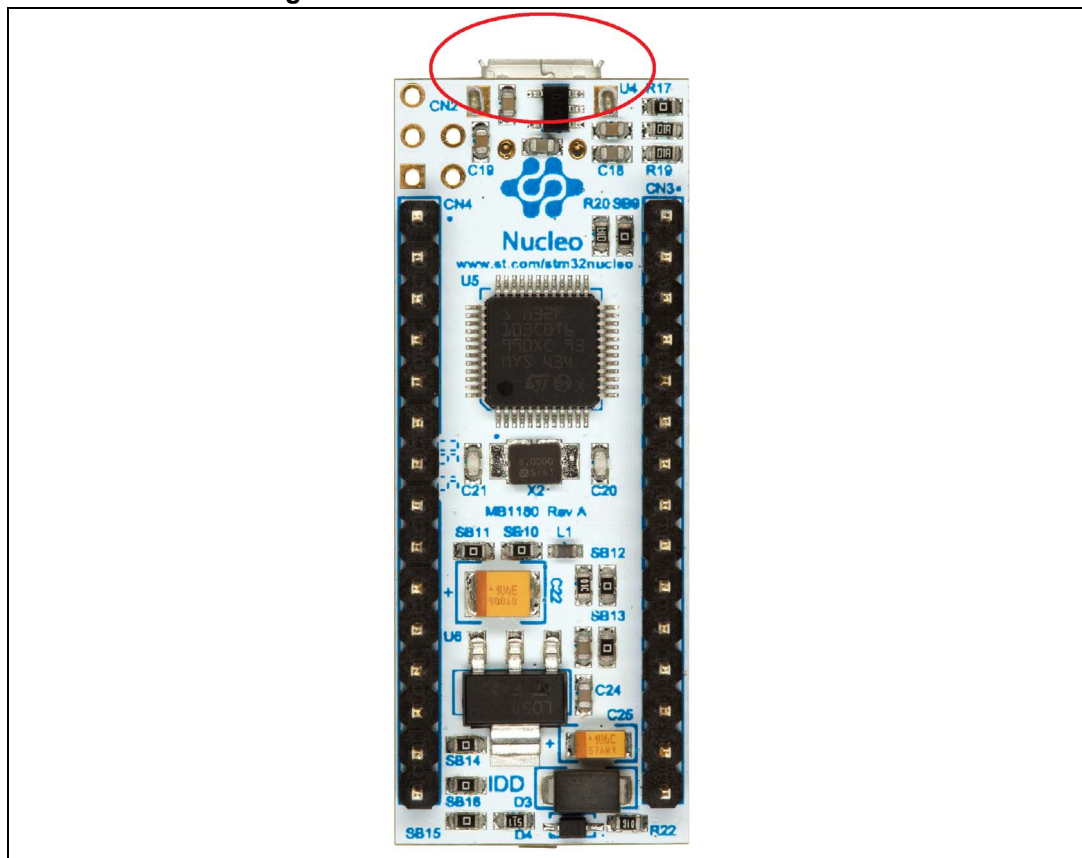**Figure 3. Connector CN1 for Nucleo 64 board**

Figure 4. Connector CN1 for Nucleo 32 board

2 IDEs supporting STM32 families

STMicroelectronics STM32 families of 32-bit ARM® Cortex® -M core-based microcontrollers are supported by a complete range of software tools. It encompasses traditional integrated development environment IDEs with C/C++ compilers and debuggers from major third-parties (free versions up to 64 Kbytes of code, depending on third-party), completed with innovative tools from STMicroelectronics.

Toolchains supporting all Nucleo boards are:

- EWARM v7.10.3 or later[a]
  - 30-day evaluation edition
  - 32-Kbyte Limited QuickStart edition (16-Kbyte limitation for Cortex M0)
- MDK-ARM v5.17 or later[a][b]
  - MDK-Lite (32-Kbyte code size limitation)
- TrueSTUDIO Lite v5 or later[b]
  - No limitation
- SW4STM32 v1.5 and later[a]
  - No limitation

Information on the toolchain version supporting the STM32 device is available on toolchain release note at third-party web site.

---

a. Registration before the download is required.

b. The device support pack is separate from the MDK-ARM releases.

3 ST-LINK/V2-1 installation

All the STM32 Nucleo boards include an ST-LINK/V2-1 embedded debug tool interface. This interface needs a dedicated USB driver to be installed. This driver is available at the *www.st.com* website and supported within the following software toolchains:

- IAR Embedded Workbench for ARM (EWARM).

  The toolchain is installed by default in the C:\Program Files\IAR Systems\Embedded Workbench x.x directory on the PC local hard disk.

  After installing EWARM, install the ST-LINK/V2-1 driver by running the ST-Link_V2_USB.exe from [IAR_INSTALL_DIRECTORY]\Embedded Workbench x.x\arm\drivers\ST-Link \ST-Link_V2_USBdriver.exe.

- Keil Microcontroller Development Kit (MDK-ARM) toolchain.

  The toolchain is installed by default in the C:\Keil directory on the PC local hard disk; the installer creates a start menu µVision5 shortcut.

  When connecting the ST-LINK/V2-1 tool, the PC detects new hardware and requires to install the ST-LINK_V2_USB driver. The "Found New Hardware wizard" appears and guides the user through the steps needed to install the driver from the recommended location.

- Atollic TrueSTUDIO STM32.

  The toolchain is installed by default in the C:\Program Files\Atollic directory on the PC local hard disk.

  The ST-Link_V2_USB.exe file is automatically installed when installing the software toolchain.

- AC6 System Workbench for STM32 (SW4STM32).
  The toolchain is installed by default in the C:\Program Files\AC6 directory on the PC local hard disk.
 The ST-Link_V2_USB.exe file is automatically executed when installing the software toolchain.

Complementary information on the firmware package content and the STM32 Nucleo requirements is available on the *Getting started with the STM32 Nucleo board firmware package* UM1726 User manual at the *www.st.com* website.

Note: *The embedded ST-LINK/V2-1 only supports SWD interface for STM32 devices.*

4 Firmware package

The STM32 Nucleo examples, applications and demonstration are provided in one single .zip file. The extraction of the .zip file generates one folder, STM32 Nucleo_FW_VX.Y.Z, which contains the following subfolders (see *Figure 5*).

Figure 5. Package contents
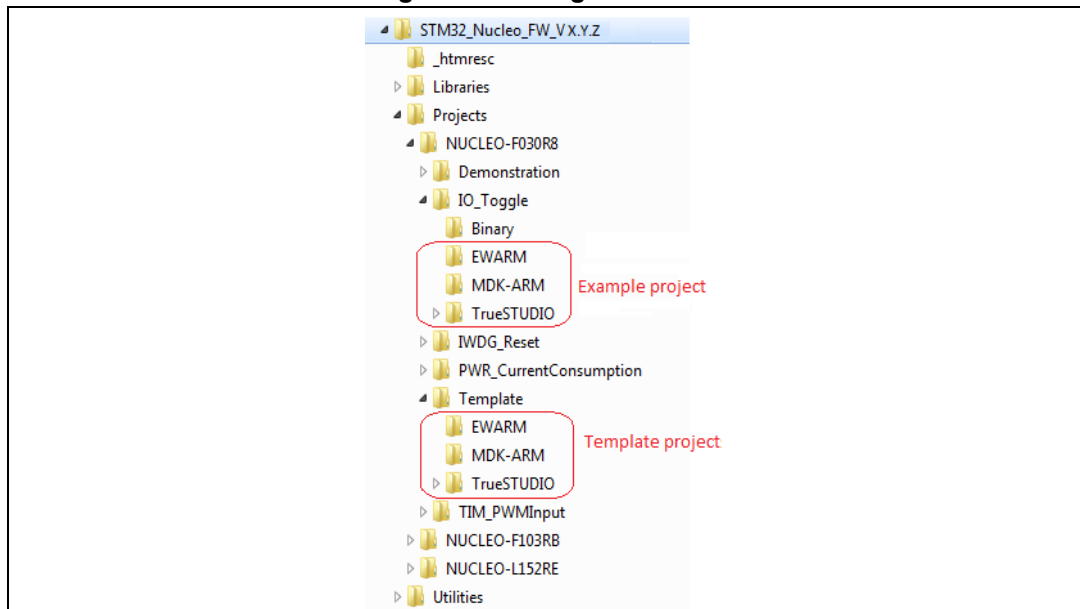
- **Template project** is a pre-configured project with empty main function to be customized by the user. This is helpful to start creating an application based on the peripherals drivers.
- **Example project** includes toolchain projects for each peripheral example ready to be run.
- **Applications** includes set of applications ready to be run.
- **Demonstration** includes the demonstration firmware ready to be run.

4.1 Executing and debugging firmware using software toolchains

The steps below can be applied to an already existing example, demonstration or template project available from STM32_Nucleo_FW_VX.Y.Z firmware at the *www.st.com* website.

First of all, the user must read the firmware/readme.txt file, which contains the firmware description and hardware/software requirements.
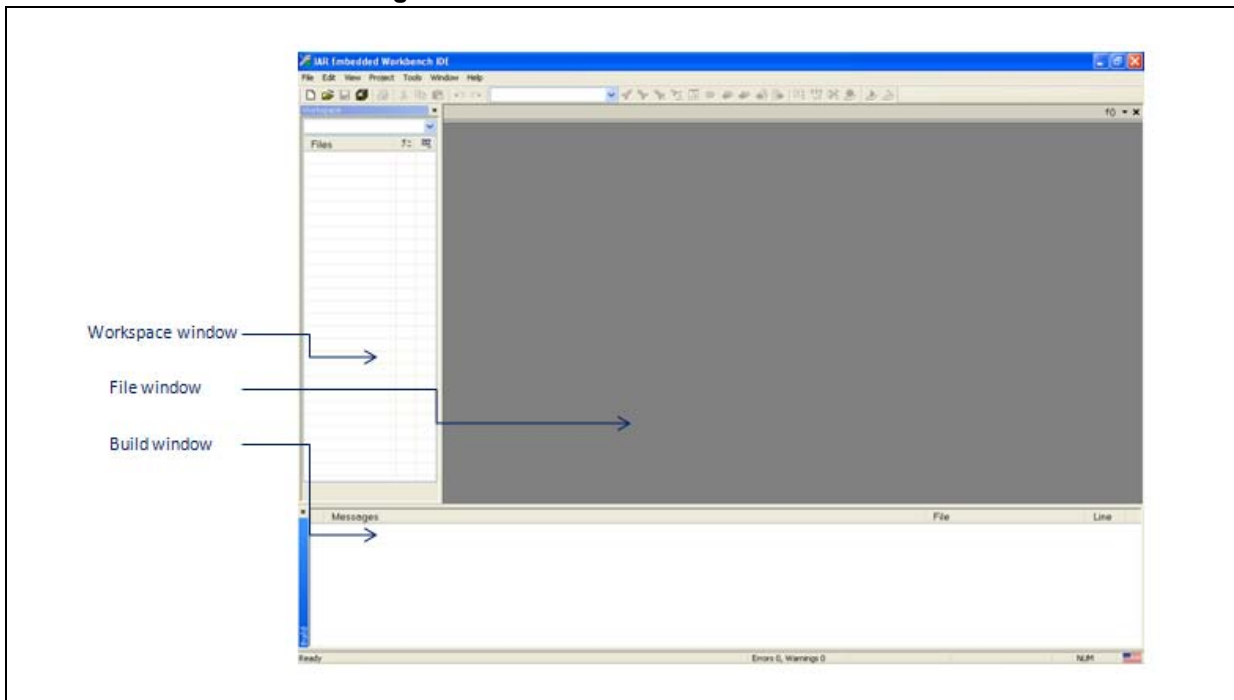
4.1.1 EWARM toolchain

The following procedure explains how to compile, link and execute an existing EWARM project.

1. Open IAR Embedded Workbench for ARM (EWARM). *Figure 6* shows the basic names of the windows referred to in this document:

Figure 6. IAR Embedded Workbench IDE

2. In the **File** menu, select **Open** and click **Workspace** to display the Open Workspace dialog box. Browse to select either an example or demonstration or template workspace file and click **Open** to launch it in the Project window.

3. In the **Project** menu, select **Rebuild All** to compile the project.

4. If the project is successfully compiled, *Figure 7* is displayed.

**Figure 7. EWARM project successfully compiled**
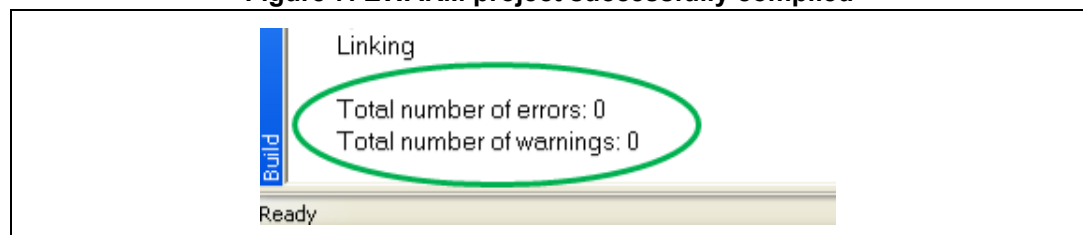
To change the project settings (Include and preprocessor defines), go through the following project options:

– For Include directories:

Project>Options…>C/C++ compiler>

– For pre-processor defines:

**Project>Options…C/C++ compiler>pre-processor>**

5. In IAR Embedded Workbench IDE, from the **Project** menu, select **Download and Debug** or, alternatively, click the **Download and Debug** button in the toolbar, to program the Flash memory and start debugging (see *Figure 8*).

Figure 8. Download and debug button
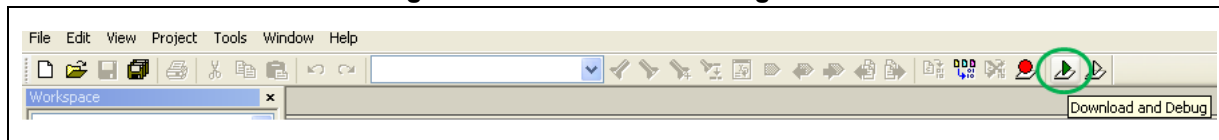
6. The debugger in IAR Embedded Workbench can be used to debug source code at C and assembly levels, set breakpoints, monitor individual variables and watch events during the code execution (see *Figure 9*).

Figure 9. IAR Embedded Workbench debugger screen
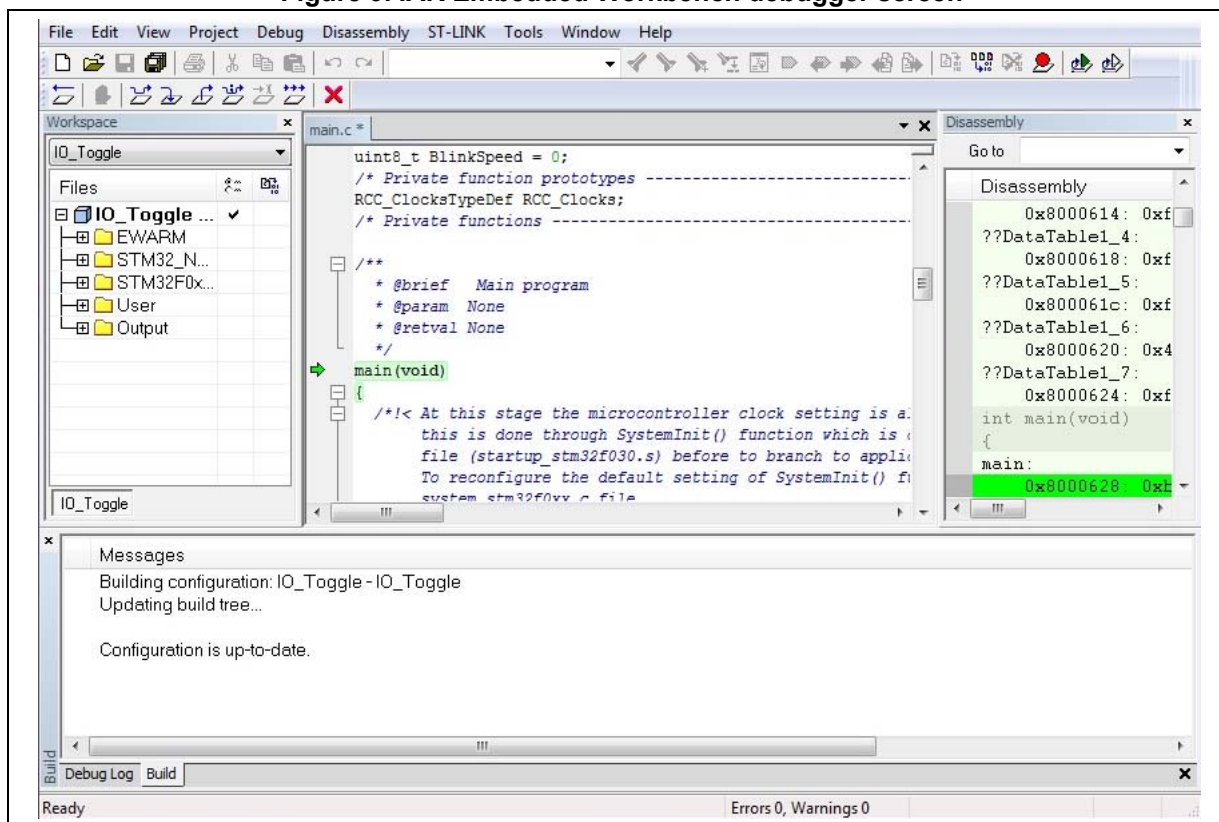
To run the application from the **Debug** menu, select **Go**. Alternatively, click the **Go** button in the toolbar to run the application (see *Figure 10*).
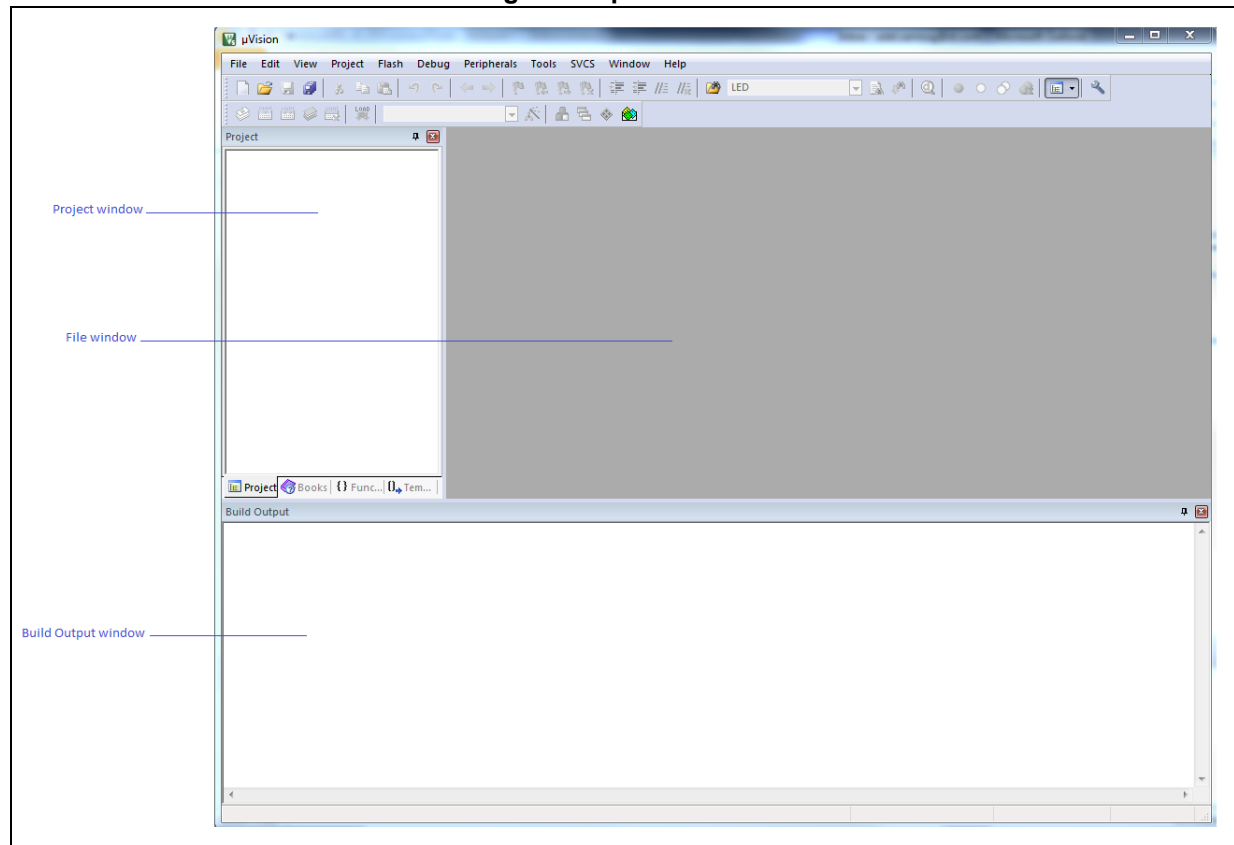
Figure 10. Go button
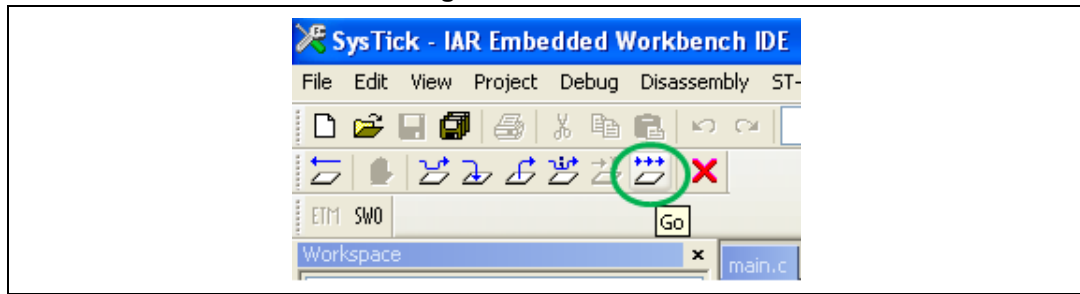
4.1.2 MDK-ARM toolchain

1. Open Keil MDK-ARM Microcontroller Development Kit. *Figure 11* shows the basic names of the "Keil µvision5" windows, to which this document refers.

Figure 11. µvision5 IDE

2. In the **Project** menu, select **Open Project...** Browse to select either an example or a demonstration or a template project file and click **Open** to launch it in the Project window.

3. In the **Project** menu, select **Rebuild All target files** to compile the project.

4. If the project is successfully compiled, the following window is displayed (see *Figure 12*):

Figure 12. MDK-ARM project successfully compiled
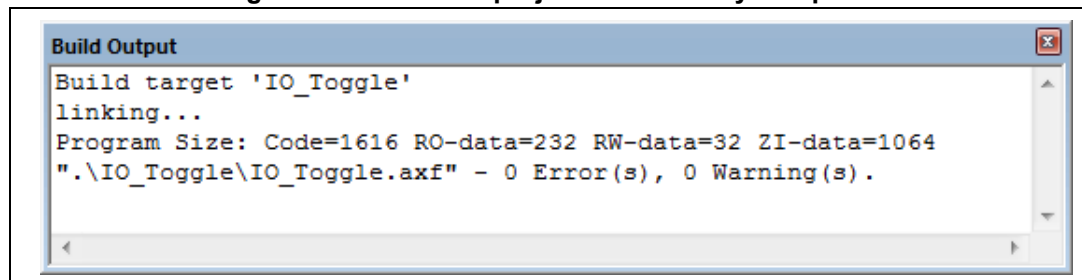
If user needs to change his project settings (Include and preprocessor defines), he must go through the project options:

– For Include directories:

Project>Options for Target > C/C++ > Include Paths

– For pre-processor defines:

**Project>Options for Target > C/C++ > Preprocessor symbols > Define**

5. In MDK-ARM IDE, from the **Debug** menu, select **Start/Stop Debug Session** or, alternatively, click the **Start/Stop Debug Session** button in the toolbar, to program the Flash memory and start debugging (see *Figure 13*).

Figure 13. Start/Stop Debug Session button
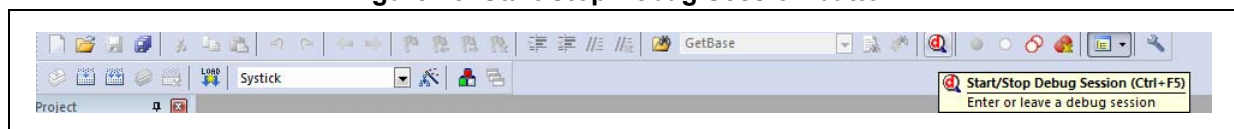
6. The debugger in the MDK-ARM can be used to debug source code at C and assembly levels, set breakpoints, monitor individual variables and watch events during the code execution.

Figure 14. MDK-ARM debugger screen
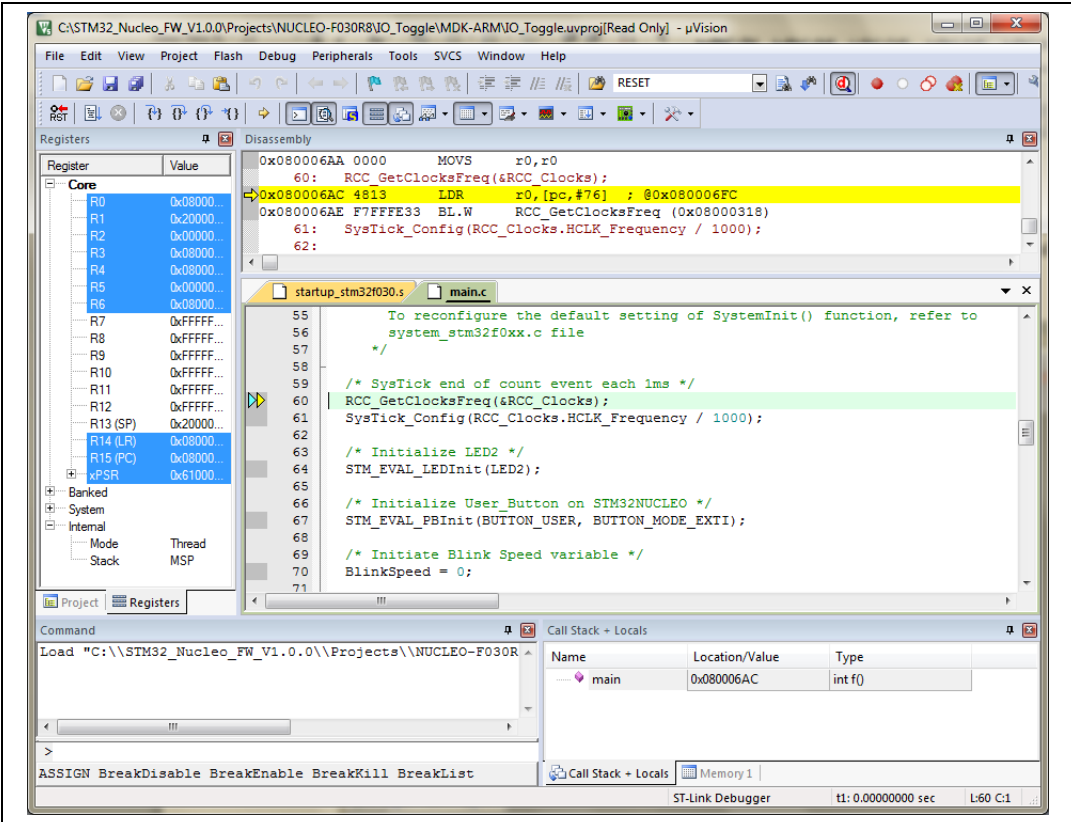
To run the application from the **Debug menu**, select **Run**. Alternatively, click the **Run** button in the toolbar to run the application (see *Figure 15*).
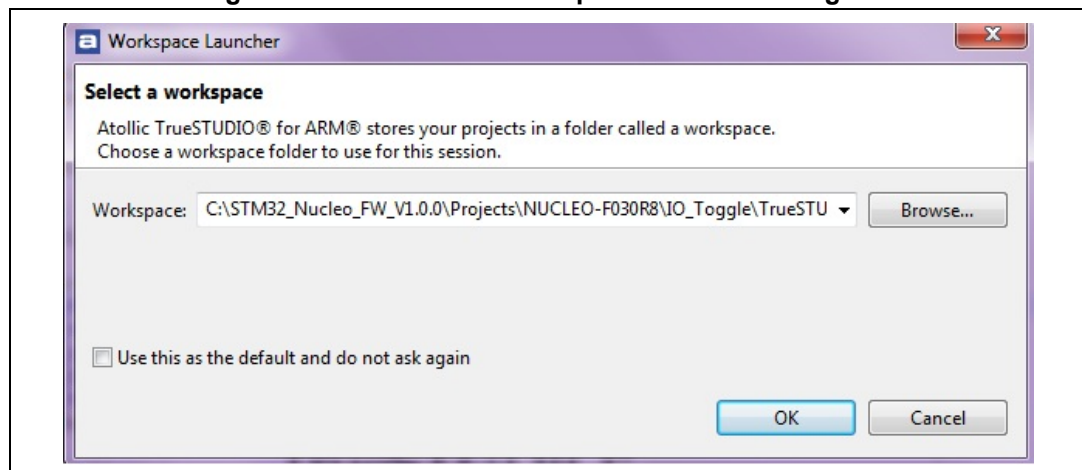
Figure 15. Run button
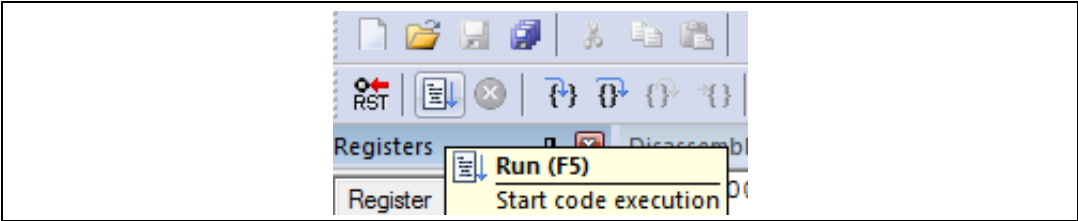
4.1.3 TrueSTUDIO toolchain

1. Open Atollic TrueSTUDIO for ARM product. The program launches and prompts for the workspace location.

Figure 16. TrueSTUDIO workspace launcher dialog box

2. Browse to select a TrueSTUDIO workspace of either an example or a demonstration or a template workspace file and click **OK** to load it.

3. To load an existing project in the selected workspace, select **Import** from the **File** menu to display the **Import** dialog box.

4. In the **Import** window, open **General**, select **Existing Projects into Workspace** and click **Next** (see *Figure 17*).

Figure 17. Atollic TrueSTUDIO import source select dialog box
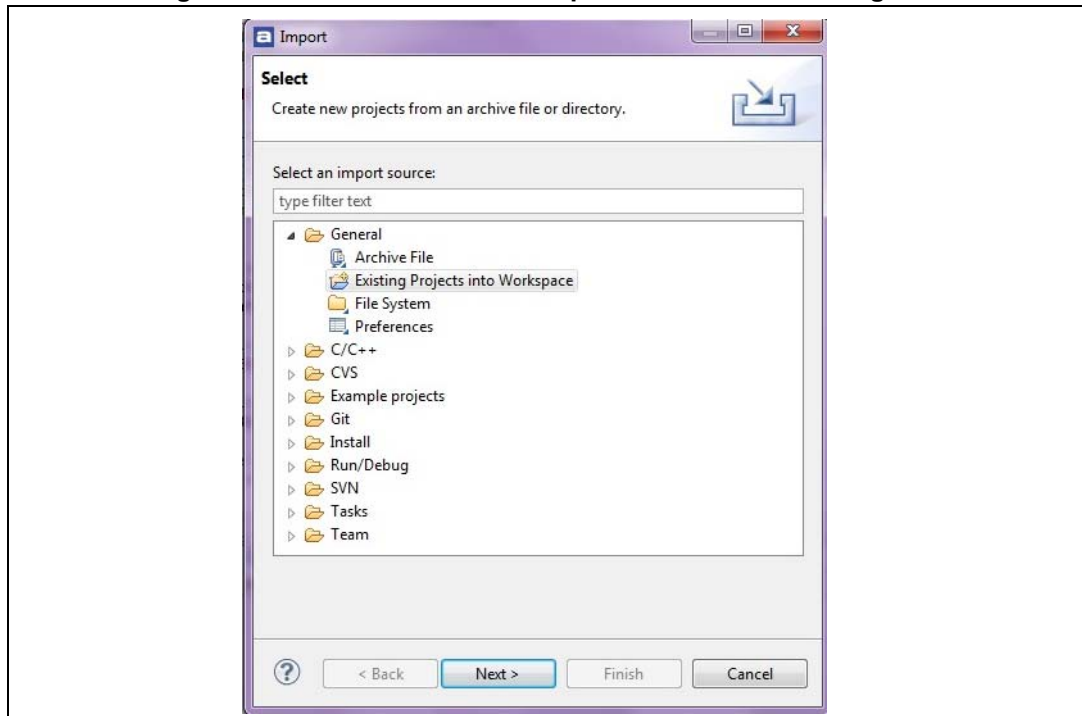
5. Click **Select** root directory and browse to the TrueSTUDIO workspace folder, as showed in *Figure 18*.

Figure 18. Atollic TrueSTUDIO import projects dialog box
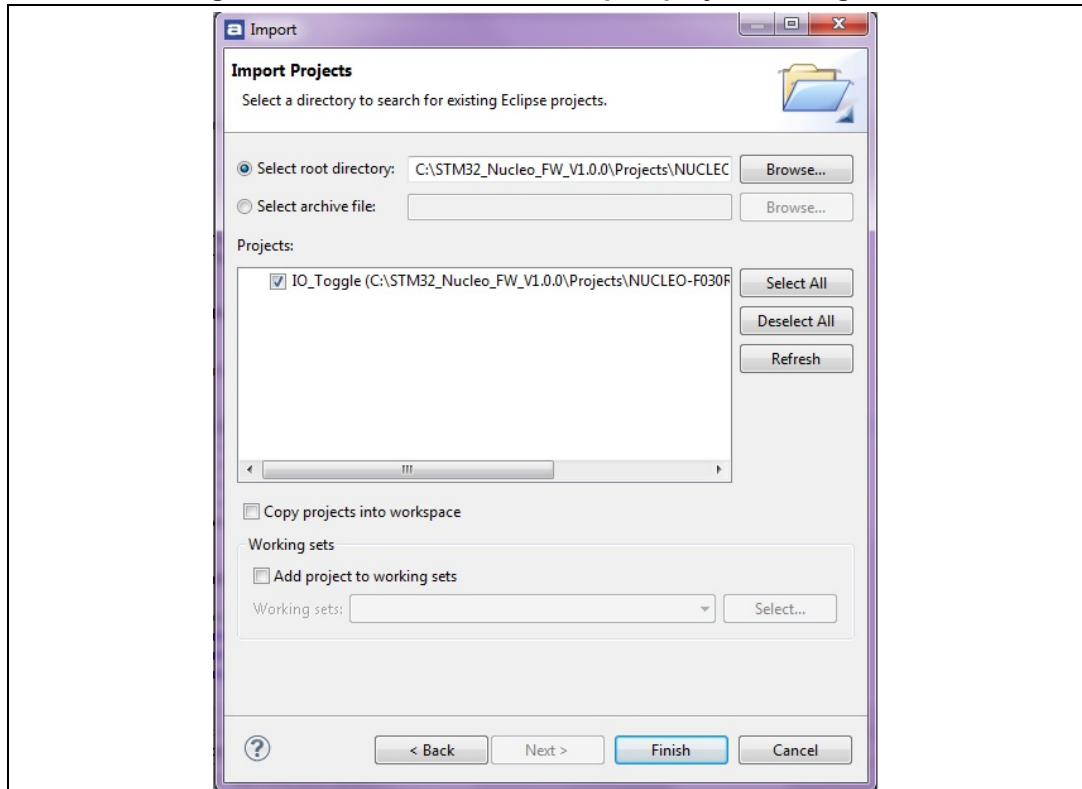
6. In the **Projects** panel, select the project and click **Finish**.

7. In the **Project Explorer**, select the project, open the **Project** menu, and click **Build Project**.

8. If the project is successfully compiled, the following messages will be displayed on the console window (see *Figure 19*).

Figure 19. TrueSTUDIO project successfully compiled

If user needs to change the project settings (Include directories and preprocessor defines), he must go through Project>Properties and select C/C++ Build>Settings from the left panel:

– For Include directories:

C Compiler>Directories>Include path

– For pre-processor defines:

C Compiler>Symbols> Defined symbols

9. To debug and run the application, select the project in the **Project Explorer** and press **F11** to start a debug session (see *Figure 20*).

Figure 20. TrueSTUDIO debug window
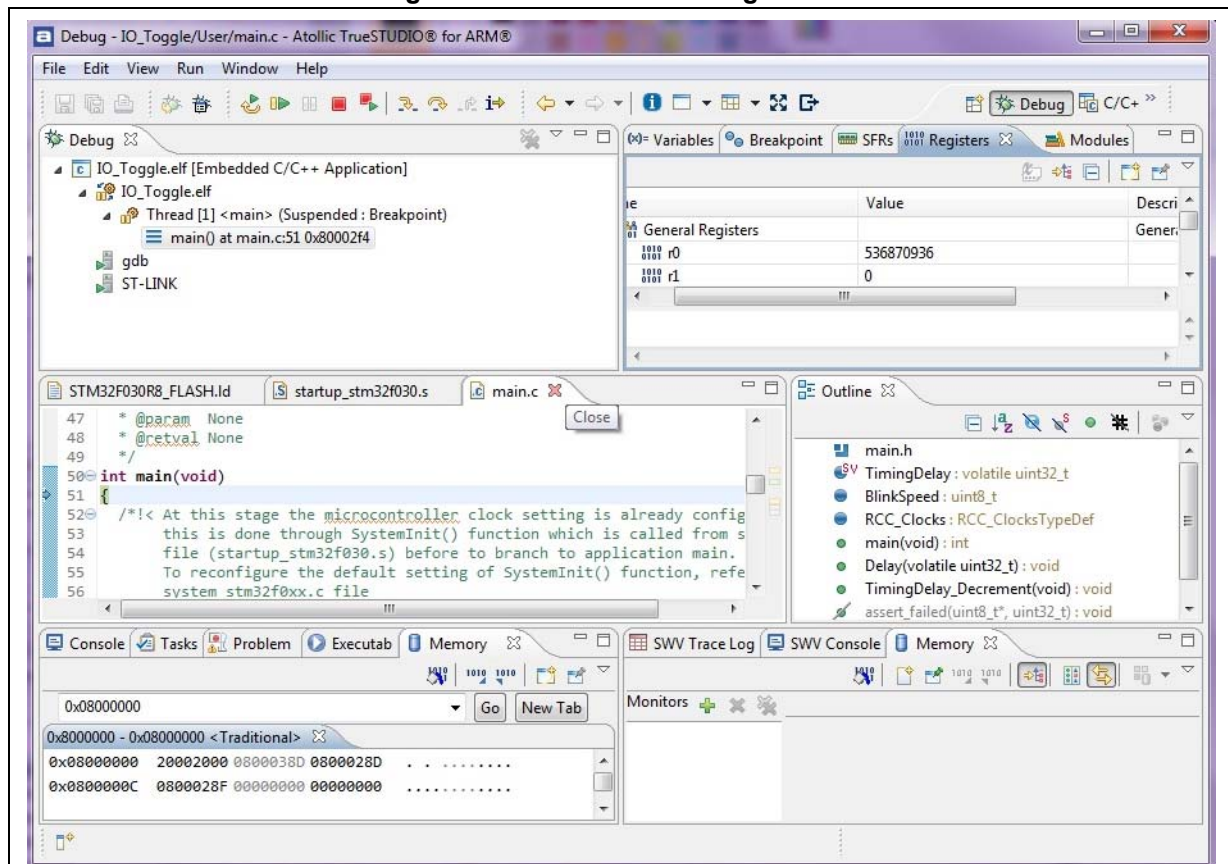
The debugger in the Atollic TrueSTUDIO can be used to debug source code at C and assembly levels, set breakpoints, monitor individual variables and watch events during the code execution.

To run the application, from the **Run** menu, select **Resume**, or alternatively click the **Resume** button in the toolbar.

4.1.4 SW4STM32 toolchain

1. Open the AC6 SW4STM32 for STM32. The program launches and prompts for the workspace location (See *Figure 21*).

**Figure 21. SW4STM32 workspace launcher dialog box**
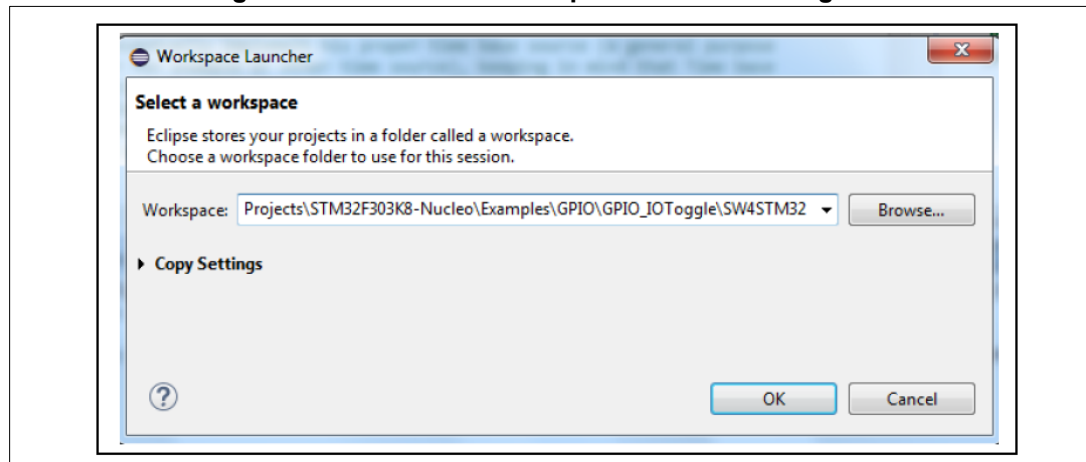
2. Browse to select a SW4STM32 workspace of either an example or a demonstration or a template workspace file and click OK to load it.

3. To load an existing project in the selected workspace, select Import from the File menu to display the Import dialog box.

4. In the Import window, open General, select Existing Projects into Workspace and click Next (See *Figure 22*).

Figure 22. SW4STM32 import source select dialog box
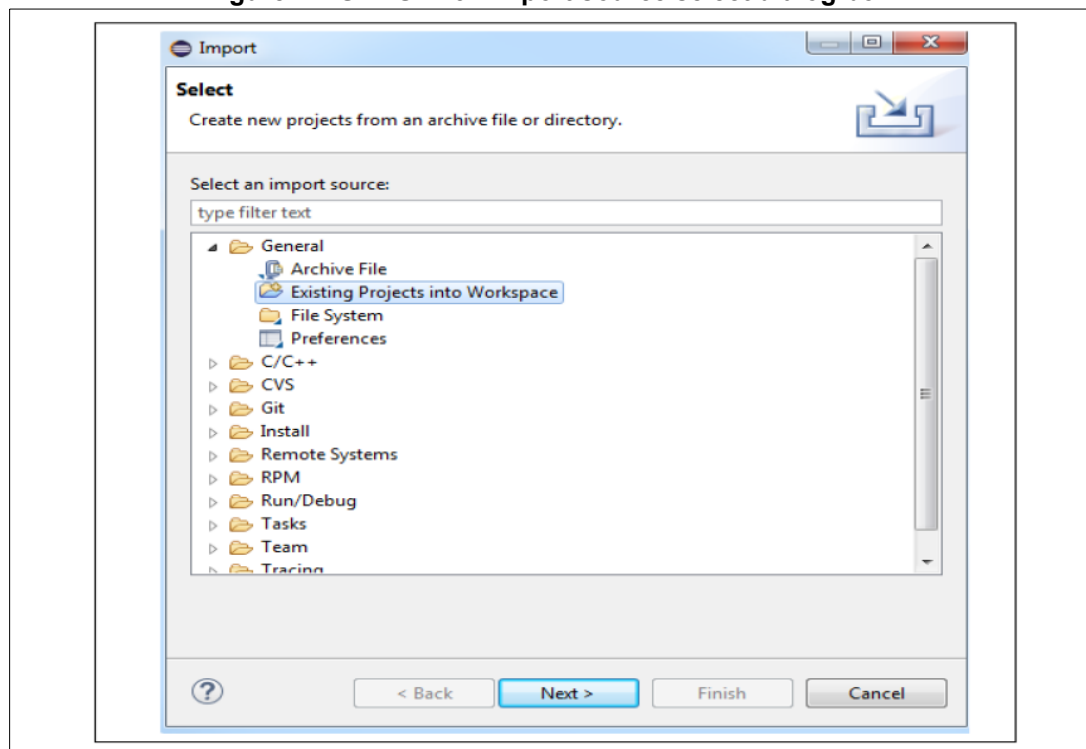
5. Click Select root directory, browse to the SW4STM32 workspace folder (See *Figure 23*).

Figure 23. SW4STM32 import projects dialog box
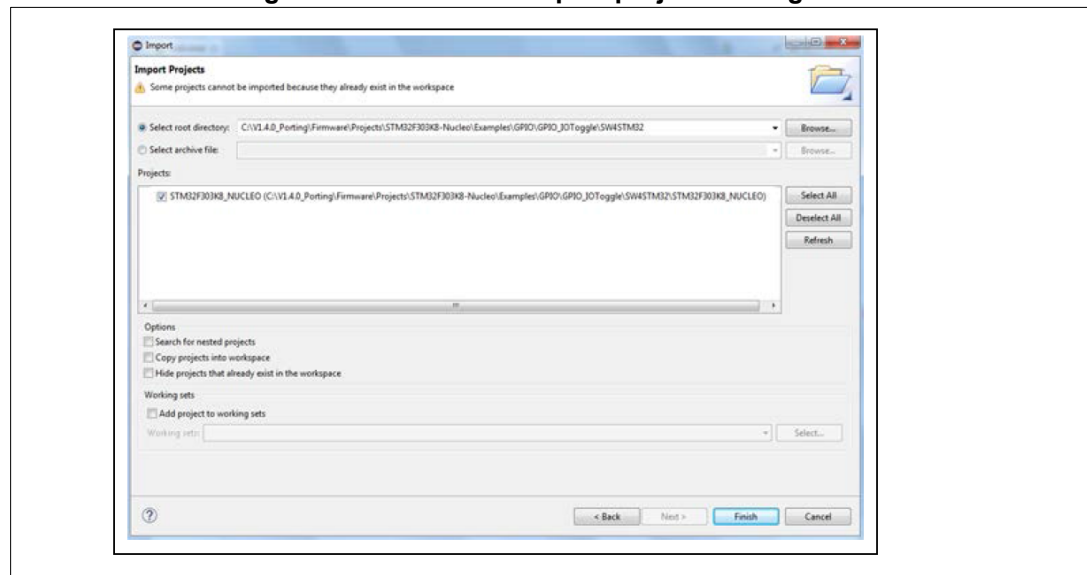
6. In the Projects panel, select the project and click Finish.

7. In the Project Explorer, select the project, open the Project menu, and click Build Project.

8. If the project is successfully compiled, the following messages display on the console window (See *Figure 24*).

Figure 24. SW4STM32 project successfully compiled
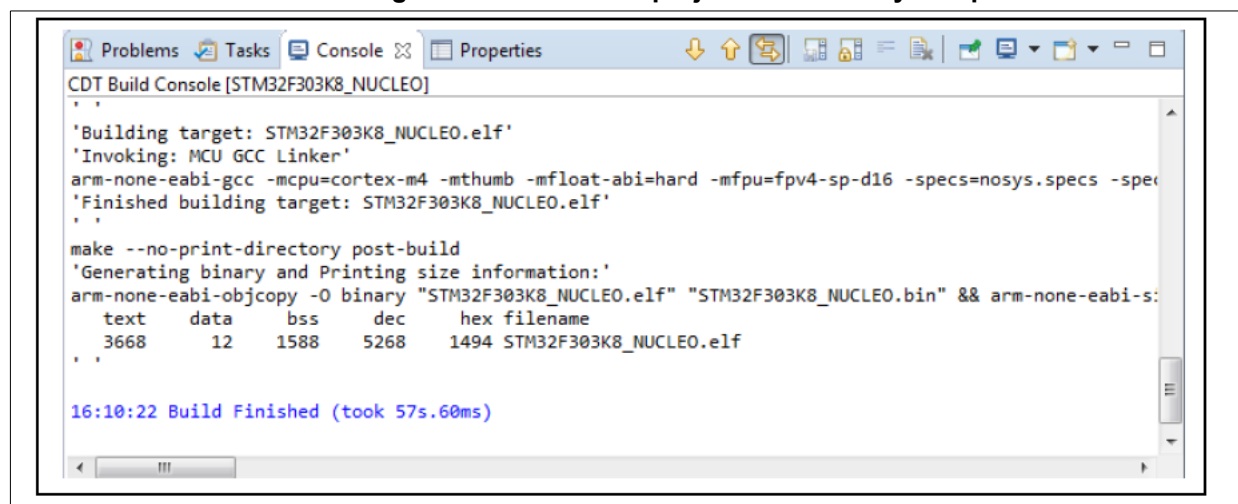
To change the project settings (Include directories and preprocessor defines), simply go through Project>Properties, select C/C++ Build>Settings from the left panel:

- For Include directories follow the path:

  C Compiler>Directories>Include path

- For pre-processor defines follow the path:

  C Compiler>Symbols> Defined symbols

9. To debug and run the application, select the project in the Project Explorer and press F11 to start a debug session (see *Figure 25*).

Figure 25. SW4STM32 debug window
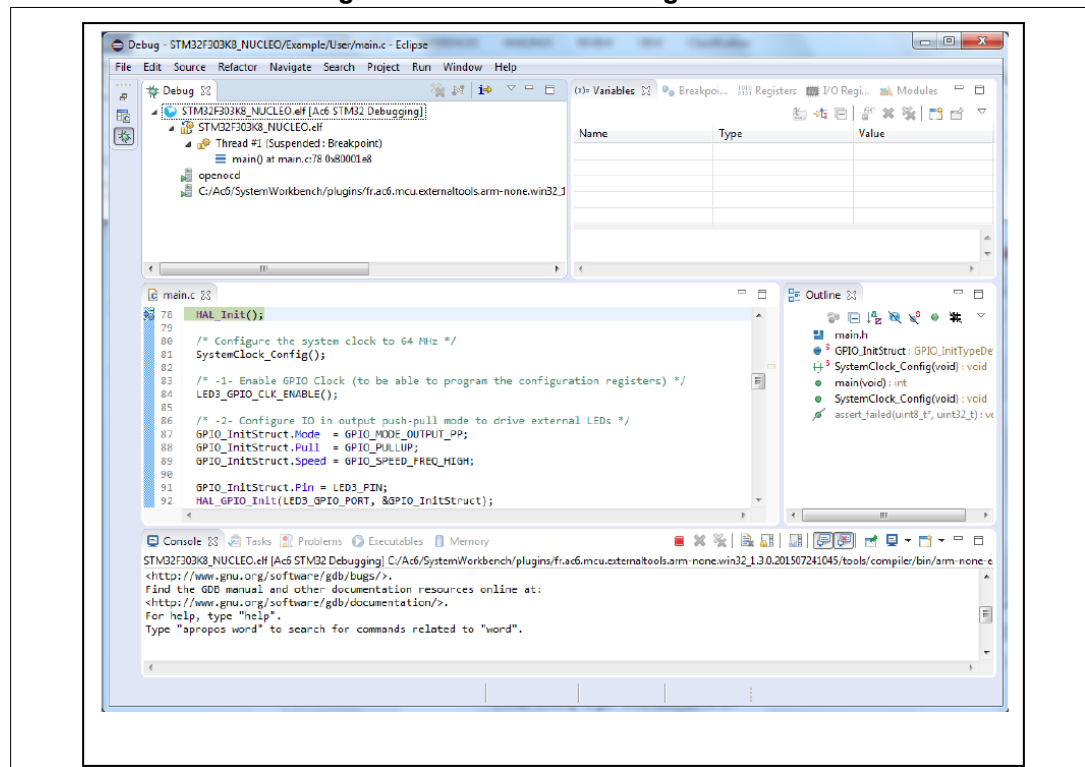
The debugger in the AC6 SW4STM32 can be used to debug source code at the C and assembly levels, to set breakpoints, to monitor individual variables and to watch events during the code execution.

To run an application select Resume, from the Run menu, or alternatively click the Resume button in the toolbar.

5 Revision history

Table 1. Document revision history

Date	Revision	Changes
19-May-2014	1	Initial release
20-Jun-2014	2	Added new part numbers in the cover page. Removed Table1 Nucleo board General information.
09-Sep-2014	3	Replaced all rpns of NUCLEO with reference to the "STM32 MCU Nucleo" line.
30-Nov-2015	4	Added *Section 4.1.4: SW4STM32 toolchain*. Updated *Section : Introduction*, *Section 3: ST-LINK/V2-1 installation*, *Section 4: Firmware package*.
19-Jan-2016	5	Updated *: Introduction* and *Section 1: System requirements* to introduce the Nucleo 144 board.

IMPORTANT NOTICE – PLEASE READ CAREFULLY