Introduction

This document provides an introduction on how to use the following software development toolchains with the STM32L-DISCOVERY and 32L152CDISCOVERY boards.

- IAR Embedded Workbench® for ARM (EWARM) by IAR Systems
- Microcontroller Development Kit for ARM (MDK-ARM) by Keil™
- TrueSTUDIO® by Atollic
- TASKING VX-toolset for ARM® Cortex™-M by Altium™.

It provides guidelines to novice users on how to build and run a sample program provided with this document and allows them to create and build their own application.

When running the sample program supplied with this application note, the Red LED LD2 (PWR) lights up. Then the user will be able to run a series of functions (VDD voltage measurement, STM32L current consumption...) by pressing the user button B1 to switch from a function to an other (please refer to AN3413).

Although this application note cannot cover all the topics relevant to software development environments, it demonstrates the first basic steps necessary to get started with the compilers/debuggers.*

STM32L1xxDISCOVERY stands either for STM32L-DISCOVERY or 32L152CDISCOVERY evaluation kit throughout the document.

Reference documents

- STM32L-DISCOVERY and 32L152CDISCOVERY user manual (UM1079)
- STM32L1x current consumption measurement and touch sensing demonstration (AN3413)

The above documents are available at www.st.com/stm32l.
Contents

1 Overview of STLINK/V2 interface ............................................. 6

2 Hardware environment setup ................................................... 7

3 Using the IAR Embedded Workbench® for ARM ......................... 8
   3.1 Building an existing EWARM project ................................. 8
   3.2 Debugging and running your EWARM project ...................... 9
   3.3 Creating your first application using the EWARM toolchain ....... 10
      3.3.1 Managing source files .................................... 10
      3.3.2 Configuring project options ................................. 12

4 Using the MDK-ARM microcontroller development kit by Keil™ ...... 16
   4.1 Building an existing MDK-ARM project ............................. 16
   4.2 Debugging and running your MDK-ARM project ................. 17
   4.3 Creating your first application using the MDK-ARM toolchain . 18
      4.3.1 Managing source files .................................. 18
      4.3.2 Configuring project options ............................... 20

5 Using the Atollic TrueSTUDIO® .............................................. 23
   5.1 Building an existing TrueSTUDIO project ....................... 23
   5.2 Debugging and running your TrueSTUDIO project ............... 26
   5.3 Creating your first application using TrueSTUDIO toolchain . 26

6 Using Altium TASKING toolset ............................................. 31
   6.1 Building an existing TASKING project ....................... 31
   6.2 Debugging and running your TASKING project .................. 34
   6.3 Creating your first application using TASKING toolchain .... 35

7 Revision history .............................................................. 39
List of tables

Table 1. Document revision history ............................................. 39
## List of figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hardware environment</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>IAR Embedded Workbench IDE environment</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>EWARM project successfully compiled</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>Download and debug button</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>IAR Embedded Workbench debugger screen</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>Create new project dialog box</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>IDE interface</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>main.c example file</td>
<td>11</td>
</tr>
<tr>
<td>9</td>
<td>Adding files to a project</td>
<td>11</td>
</tr>
<tr>
<td>10</td>
<td>New project file tree structure</td>
<td>11</td>
</tr>
<tr>
<td>11</td>
<td>Configuring project options</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>General options</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>Linker &gt; Config tab</td>
<td>13</td>
</tr>
<tr>
<td>14</td>
<td>Linker configuration file editor dialog box &gt; Vector Table tab</td>
<td>13</td>
</tr>
<tr>
<td>15</td>
<td>Linker configuration file editor dialog box &gt; Memory Regions tab</td>
<td>13</td>
</tr>
<tr>
<td>16</td>
<td>C/C++ Compiler &gt; Preprocessor tab</td>
<td>14</td>
</tr>
<tr>
<td>17</td>
<td>Debugger &gt; Setup tab</td>
<td>14</td>
</tr>
<tr>
<td>18</td>
<td>Select Flash loaders</td>
<td>14</td>
</tr>
<tr>
<td>19</td>
<td>ST-Link communication protocol</td>
<td>15</td>
</tr>
<tr>
<td>20</td>
<td>MDK-ARM µVision4 IDE environment</td>
<td>16</td>
</tr>
<tr>
<td>21</td>
<td>MDK-ARM µVision4 project successfully compiled</td>
<td>16</td>
</tr>
<tr>
<td>22</td>
<td>Starting a MDK-ARM µVision4 debugging session</td>
<td>17</td>
</tr>
<tr>
<td>23</td>
<td>MDK-ARM IDE workspace</td>
<td>17</td>
</tr>
<tr>
<td>24</td>
<td>Creating a new project</td>
<td>18</td>
</tr>
<tr>
<td>25</td>
<td>Device selection dialog box</td>
<td>18</td>
</tr>
<tr>
<td>26</td>
<td>Copy the STM32 Startup Code dialog box</td>
<td>19</td>
</tr>
<tr>
<td>27</td>
<td>main.c example file</td>
<td>19</td>
</tr>
<tr>
<td>28</td>
<td>Adding source files</td>
<td>19</td>
</tr>
<tr>
<td>29</td>
<td>New project file tree structure</td>
<td>20</td>
</tr>
<tr>
<td>30</td>
<td>Target Options dialog box - Target tab</td>
<td>20</td>
</tr>
<tr>
<td>31</td>
<td>Target Options dialog box - Debug tab</td>
<td>21</td>
</tr>
<tr>
<td>32</td>
<td>Target Options dialog box - Utilities tab</td>
<td>21</td>
</tr>
<tr>
<td>33</td>
<td>MDK-ARM µVision4 project successfully built</td>
<td>22</td>
</tr>
<tr>
<td>34</td>
<td>TrueSTUDIO workspace launcher dialog box</td>
<td>23</td>
</tr>
<tr>
<td>35</td>
<td>Atollic TrueSTUDIO®/STM32 Lite import source select dialog box</td>
<td>24</td>
</tr>
<tr>
<td>36</td>
<td>Atollic TrueSTUDIO®/STM32 Lite import projects dialog box</td>
<td>25</td>
</tr>
<tr>
<td>37</td>
<td>TrueSTUDIO® project successfully compiled</td>
<td>25</td>
</tr>
<tr>
<td>38</td>
<td>TrueSTUDIO® debug window</td>
<td>26</td>
</tr>
<tr>
<td>39</td>
<td>TrueSTUDIO workspace launcher dialog box</td>
<td>27</td>
</tr>
<tr>
<td>40</td>
<td>TrueSTUDIO® C Project dialog box</td>
<td>27</td>
</tr>
<tr>
<td>41</td>
<td>TrueSTUDIO® Build Settings dialog box</td>
<td>28</td>
</tr>
<tr>
<td>42</td>
<td>TrueSTUDIO® software configuration dialog box</td>
<td>29</td>
</tr>
<tr>
<td>43</td>
<td>TrueSTUDIO® Misc Settings dialog box</td>
<td>29</td>
</tr>
<tr>
<td>44</td>
<td>TrueSTUDIO® project folder example</td>
<td>30</td>
</tr>
<tr>
<td>45</td>
<td>TrueSTUDIO® project successfully built</td>
<td>30</td>
</tr>
<tr>
<td>46</td>
<td>TASKING workspace launcher dialog box</td>
<td>31</td>
</tr>
<tr>
<td>47</td>
<td>TASKING VX-Toolset for ARM Cortex welcome screen</td>
<td>31</td>
</tr>
<tr>
<td>48</td>
<td>TASKING import source select dialog box</td>
<td>32</td>
</tr>
</tbody>
</table>
Figure 49. TASKING import projects dialog box .................................................. 33
Figure 50. TASKING project successfully compiled .............................................. 33
Figure 51. TASKING debug window ................................................................. 34
Figure 52. TASKING Workspace Launcher dialog box ................................. 35
Figure 53. TASKING New C/C++ Project dialog box ..................................... 35
Figure 54. Processor selection ....................................................................... 36
Figure 55. Debug configurations ..................................................................... 36
Figure 56. TASKING Import dialog box .......................................................... 37
Figure 57. Adding a new source file window .................................................... 37
Figure 58. Tasking project successfully built ..................................................... 38
1 Overview of STLINK/V2 interface

The STM32L1xxDISCOVERY board includes an ST-LINK/V2 embedded debug tool interface that is supported by the following software toolchain versions:

- **EWARM Version 6.50.3 and later available from www.iar.com**
  Installing the EWARM toolchain (using the default settings) results in the toolchain being installed in the C:\Program Files\IAR Systems\Embedded Workbench 6.2 directory on the PC’s local hard disk.
  After installing EWARM 6.20.4, install the ST-LINK/V2 driver by running the ST-Link_V2_USB.exe from [IAR_INSTALL_DIRECTORY]\Embedded Workbench 6.2\arm\drivers\ST-Link\ST-Link_V2_USBdriver.exe

- **MDK-ARM Version 4.71 and later available from www.keil.com**
  Installing the MDK-ARM toolchain (using the default settings) results in the toolchain being installed in the C:\Keil directory on the PC’s local hard disk; the installer creates a start menu uVision4 shortcut.
  When connecting the ST-LINK/V2 tool, the PC detects new hardware and asks to install the ST-LINK_V2_USB driver. The “Found New Hardware wizard” appears and guides you through the steps needed to install the driver from the recommended location.

- **TrueSTUDIO Version 4.0.0 and later available from www.atollic.com**
  Installing the TrueSTUDIO toolchain (using the default settings) results in the toolchain being installed in the “C:\Program Files\Atollic” directory on the PC’s local hard disk. The ST-Link_V2_USB.exe is installed automatically when installing the software toolchain.

- **TASKING VX-toolset Version 4.4r1 available from www.tasking.com**
  The toolchain is installed by default in the “C:\Program Files\TASKING” directory on your PC local hard disk. The ST-Link_V2_USB.exe is installed automatically when installing the software toolchain.
2 Hardware environment setup

Before running your application, you should establish the connection with the STM32L1xxDISCOVERY board as following.

Figure 1. Hardware environment

For more details on how to establish your hardware environment you can refer to the STM32L-DISCOVERY and 32L152CDISCOVERY user manual (UM1079) user manual available from http://www.st.com.
3 Using the IAR Embedded Workbench® for ARM

3.1 Building an existing EWARM project

1. Open the IAR Embedded Workbench® for ARM (EWARM).

   Figure 2 shows the basic names of the windows referred to in this document.

   **Figure 2. IAR Embedded Workbench IDE environment**

2. In the **File** menu, select **Open** and click **Workspace** to display the Open Workspace dialog box. Browse to select the *STM32L-Discovery.eww* workspace file and click **Open** to launch it in the Project window.

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

4. If your project is successfully compiled, the following window is displayed.
3.2 Debugging and running your EWARM project

In the IAR Embedded Workbench IDE, from the Project menu, select Download and Debug or, alternatively, click the Download and debug button the in toolbar, to program the Flash memory and begin debugging.

The debugger in the 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.

1. The above example is given for the 32L152CDISCOVERY board.
To run your application, from the Debug menu, select Go, or alternatively click the Go button in the toolbar.

### 3.3 Creating your first application using the EWARM toolchain

#### 3.3.1 Managing source files

1. In the Project menu, select Create New Project and click OK to save your settings.

#### Figure 6. Create new project dialog box

2. Name the project, NewProject.ewp for example, and click Save to display the IDE interface.

#### Figure 7. IDE interface
To create a new source file, in the File menu, open New and select File to open an empty editor window where you can enter your source code.

The IAR Embedded Workbench enables C color syntax highlighting when you save your file using the dialog File > Save As... under a filename with the *.c extension. In this example, the file is saved as main.c.

![Figure 8. main.c example file](image)

Once you have created your source file you can add this file to your project, by opening the Project menu, selecting Add and adding the selected file.

![Figure 9. Adding files to a project](image)

If the file is added successfully, the following window is displayed.

![Figure 10. New project file tree structure](image)
3.3.2 Configuring project options

1. In the Project Editor, right-click on the project name and select Options to display the Options dialog box.

![Figure 11. Configuring project options](image1)

2. In the Options dialog box, select the General Options category, open the Target tab and select either Device - ST -STM32L152xB (STM32L-DISCOVERY) or Device - ST -STM32L152xC (32L152CDISCOVERY).

![Figure 12. General options](image2)

1. The above example is given for the 32L152CDISCOVERY board.
3. Select the Linker category, open the Config tab, in the Linker configuration file pane select Override default and click Edit. to display the Linker configuration file editor.

**Figure 13. Linker > Config tab**

![Linker > Config tab](image)

4. In the Linker configuration file editor dialog box, open the Vector Table tab and set the .intvec.start variable to 0x08000000.

**Figure 14. Linker configuration file editor dialog box > Vector Table tab**

![Linker configuration file editor dialog box > Vector Table tab](image)

5. Open the Memory Regions tab, and enter the variables as shown in Figure 15.

**Figure 15. Linker configuration file editor dialog box > Memory Regions tab**

![Linker configuration file editor dialog box > Memory Regions tab](image)

6. Click Save to save the linker settings automatically in the Project directory.

7. If your source files include header files, select the C/C++ Compiler category, open the Preprocessor tab, and specify their paths as shown in Figure 16. The path of the include directory is a relative path, and always starts with the project directory location referenced by $PROJ_DIR$.
8. To set up the ST-Link embedded debug tool interface, select the **Debugger** category, open the **Setup tab** and select **ST-Link** from the drop-down Driver menu.

9. Open the **Download tab** and select **Use Flash loader(s)**.
10. Select the **ST-Link** category, open the ST6Link tab and select **SWD** as the connection protocol.

![Figure 19. ST-Link communication protocol](image)

11. Click **OK** to save the project settings.

12. To build your project, follow the instructions given in *Section 3.1: Building an existing EWARM project on page 8*.

13. Before running your application, establish the connection with the STM32L1xxDISCOVERY board as described in *Section 2: Hardware environment setup on page 7*.

14. To program the Flash memory and begin debugging, follow the instructions given in *Section 3.2: Debugging and running your EWARM project on page 9*. 
4 Using the MDK-ARM microcontroller development kit by Keil™

4.1 Building an existing MDK-ARM project

1. Open the MDK-ARM µVision4 IDE, debugger, and simulation environment. Figure 2 shows the basic names of the windows referred to in this document.

![Figure 20. MDK-ARM µVision4 IDE environment](image)

2. In the **Project** menu, select **Open Project...** to display the Select Project File dialog box. Browse to select the `STM32L-Discovery.uvproj` project file and click **Open** to launch it in the Project window.

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

4. If your project is successfully compiled, the following window is displayed.

![Figure 21. MDK-ARM µVision4 project successfully compiled](image)
4.2 Debugging and running your MDK-ARM project

In the MDK-ARM µVision4 IDE, click the magnifying glass to program the Flash memory and begin debugging.

Figure 22. Starting a MDK-ARM µVision4 debugging session

The debugger in the MDK-ARM IDE 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 23. MDK-ARM IDE workspace
4.3 Creating your first application using the MDK-ARM toolchain

4.3.1 Managing source files

1. In the Project menu, select New µvision Project... to display the Create Project File dialog box. Name the new project and click Save.

Figure 24. Creating a new project

2. When a new project is saved, the IDE displays the device dialog box. Select the device used for testing. In this example, we will use the STMicroelectronics device mounted on the STM32L1xxDISCOVERY board. In this case, double-click on STMicroelectronics, select either STM32L152xB (STM32L-DISCOVERY) or STM32L152xC (32L152CDISCOVERY) and click OK to save your settings.

Figure 25. Device selection dialog box

1. The above example is given for the 32L152CDISCOVERY board.
3. Click Yes to copy the STM32 Startup Code to the project folder and add the file to the project.

**Figure 26. Copy the STM32 Startup Code dialog box**

![Copy STM32 Startup Code to Project Folder and Add File to Project]

**Note:** The default STM32 startup file includes the SystemInit function. You can either comment out this file to not use it or add the system_stm32l1xx.c file from the STM32l1xx firmware library.

To create a new source file, in the **File menu**, select **New** to open an empty editor window where you can enter your source code.

The MDK-ARM toolchain enables C color syntax highlighting when you save your file using the dialog **File > Save As...** under a filename with the *.c extension. In this example, the file is saved as **main.c**.

**Figure 27. main.c example file**

```c
int main (void)
{
  return(0);
}
```

MDK-ARM offers several ways to add source files to a project. For example, you can select the file group in the **Project Window > Files** page and right-click to open a contextual menu. Select the **Add Files...** option, and browse to select the **main.c** file previously created.

**Figure 28. Adding source files**

If the file is added successfully, the following window is displayed.
### 4.3.2 Configuring project options

1. In the **Project** menu, select **Options for Target 1** to display the Target Options dialog box.

2. Open the Target tab and enter IROM1 and IARM1 start and size settings as shown in **Figure 30**.

### Figure 30. Target Options dialog box - Target tab

3. Open the **Debug** tab, click **Use** and select the **ST-Link Debugger**. Then, click **Settings** and select the **SWD** protocol. Click **OK** to save the ST-Link setup settings.
4. Select Run to main().

Figure 31. Target Options dialog box - Debug tab

5. Open the Utilities tab, select Use Target Driver for Flash Programming and select the ST-Link Debugger from the drop-down menu.

6. Verify that the Update Target before Debugging option is selected.

7. Click OK to save your settings.

Figure 32. Target Options dialog box - Utilities tab

8. In the Project menu, select Build Target.
9. If your project is successfully built, the following window is displayed.

**Figure 33. MDK-ARM µVision4 project successfully built**

```
Build target 'Target 1'
assembling startup_stm32l1xx_md.s...
compiling main.c...
linking...
Program Size: Code 340 RO-data 260 RW-data 0 ZI-data 1632
"main.obj" - 0 Error(s), 0 Warning(s).
```

10. Before running your application, establish the connection with the STM32L1xxDISCOVERY board as described in Section 2: Hardware environment setup on page 7.

11. To program the Flash memory and begin debugging, follow the instructions given in Section 4.2: Debugging and running your MDK-ARM project on page 17.
5 Using the Atollic TrueSTUDIO®

5.1 Building an existing TrueSTUDIO project

1. Open the TrueSTUDIO®/STM32 product folder and select the Atollic TrueSTUDIO® STM32 product name. The program launches and asks for the Workspace location.

   ![Figure 34. TrueSTUDIO workspace launcher dialog box](image)

   1. The above example is given for the 32L152CDISCOVERY board.
   2. Browse to select the AN3964-Temperature_sensor TrueSTUDIO workspace and click OK to save your settings.
   3. To load the AN3964-Temperature_sensor project, 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.
Figure 35. Atollic TrueSTUDIO®/STM32 Lite import source select dialog box
5. Click **Select root directory**, browse to the TrueSTUDIO workspace folder and select either **STM32L152xB** (STM32L-DISCOVERY) or **STM32L152xC** (32L152C-DISCOVERY) project.

![Figure 36. Atollic TrueSTUDIO®/STM32 Lite import projects dialog box](image)

1. The above example is given for the 32L152CDISCOVERY board.

6. In the **Projects** pane, select either the **STM32L152xB** (STM32L-DISCOVERY) or the **STM32L152xC** (32L152C-DISCOVERY) and click **Finish**.

7. In the **Project Explorer**, select either the **STM32L152xB** (STM32L-DISCOVERY) or the **STM32L152xC** (32L152C-DISCOVERY) project. Open the **Project** menu, and click **Build Project**.

8. If your project is successfully compiled, the following window is displayed.

![Figure 37. TrueSTUDIO® project successfully compiled](image)

1. The above example is given for the 32L152CDISCOVERY board.
5.2 Debugging and running your TrueSTUDIO project

In the Project Explorer, select the project and press F11 to start a debug session (see Figure 38).

![Figure 38. TrueSTUDIO debug window](image)

1. The above example is given for the 32L152CDISCOVERY board.

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 your application, from the Run menu, select Resume, or alternatively click the Resume button in the toolbar.

5.3 Creating your first application using TrueSTUDIO toolchain

TrueSTUDIO includes a dedicated connection to the STM32LxxDISCOVERY board. When choosing this connection, all required files (startup file, firmware library, etc.) are added to the workspace and sample files are generated in the project folder to simplify development. The debug settings are automatically configured by selecting either the STM32L152xB (STM32L-DISCOVERY) or the STM32L152xC (32L152CDISCOVERY) as microcontroller.

1. Open the TrueSTUDIO®/STM32 product folder and select the Atollic TrueSTUDIO® STM32 product name. The program launches and asks for the Workspace location. Browse to select an existing workspace, or enter a new workspace location and click OK to confirm.
1. The above example is given for the 32L152CDISCOVERY board.

2. When the Atollic TrueSTUDIO® displays its Welcome window, click **Start using TrueSTUDIO** to open the main window. In the **File** menu, select **New** and click **C Project**.

3. Name the new project, in the **Project** type pane select **STM32 C Project** and click **Next**.

Figure 39. TrueSTUDIO workspace launcher dialog box

Figure 40. TrueSTUDIO® C Project dialog box

1. The above example is given for the 32L152CDISCOVERY board.
4. In the TrueSTUDIO® Build Settings dialog box, select either the **STM32L152xB** (STM32L-DISCOVERY) or the **STM32L152xC** (32L152CDISCOVERY) as microcontroller, and configure the other settings as shown in Figure 41 and click **Next**.

**Figure 41. TrueSTUDIO® Build Settings dialog box**

1. The above example is given for the 32L152CDISCOVERY board.
5. In the TrueSTUDIO® software configuration dialog box, you can select one of the two available versions of newlib Runtime library and optimization options.

Figure 42. TrueSTUDIO® software configuration dialog box

6. Verify that the JTAG Probe is ST-LINK and click Finish to confirm your settings.

Figure 43. TrueSTUDIO® Misc Settings dialog box
7. Your project is successfully created. Atollic TrueSTUDIO® generates target specific sample files (main.c, stm32l1xx_it.c...) in the Project folder to simplify development. You can tailor this project to your needs by modifying these sample files.

**Figure 44. TrueSTUDIO® project folder example**

8. To build your project, in the **Project** menu, click **Build Project**.

9. Your project is successfully compiled.

**Figure 45. TrueSTUDIO® project successfully built**

10. Before running your application, establish the connection with the STM32L1xxDISCOVERY board as described in **Section 2: Hardware environment setup on page 7**.

11. To program the Flash memory and begin debugging, follow the instructions given in **Section 5.2: Debugging and running your TrueSTUDIO project on page 26**.
6 Using Altium TASKING toolset

6.1 Building an existing TASKING project

Follow these steps below to build an existing TASKING project:

1. Open the TASKING VX-toolset for ARM Cortex IDE. The program launches and requests the Workspace location.

   Figure 46. TASKING workspace launcher dialog box

   1. The above example is given for the 32L152CDISCOVERY board.
   2. Browse to select the AN3964-Temperature_sensor TASKING workspace and click **OK** to save your settings and display the Welcome screen. To start using TASKING, click **Go to the workbench**.

   Figure 47. TASKING VX-Toolset for ARM Cortex welcome screen

   3. The AN3964-Temperature_sensor TASKING workspace contains a project for the STM32L152RC (32L152C-DISCOVERY) and one for the STM32L152RB (STM32L-DISCOVERY). To load this project, 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.

Figure 48. TASKING import source select dialog box
5. Click **Select root directory**, browse to the TASKING workspace folder and select the STM32L152RC (32L152C-DISCOVERY) and one for the STM32L152RB (STM32L-DISCOVERY) project.

**Figure 49. TASKING import projects dialog box**

1. The above example is given for the 32L152CDISCOVERY board.

6. In the **Projects** window, click **Finish**.

7. In the **Project Explorer**, select either STM32L152RC (32L152C-DISCOVERY) and one for the STM32L152RB (STM32L-DISCOVERY) project. Open the **Project** menu, and click **Build Project**.

8. If your project is compiled successfully, the following window is displayed.

**Figure 50. TASKING project successfully compiled**

1. The above example is given for the 32L152CDISCOVERY board.
6.2 Debugging and running your TASKING project

Figure 51 shows the first step for debugging and running your TASKING project. From the project toolbar menu, select either Debug > Debug STM32L152RC(STM32L152C-DISCOVERY) or Debug > Debug STM32L152RB(STM32L-DISCOVERY).

1. The above example is given for the 32L152CDISCOVERY board.

The TASKING debugger can be used to debug source code at C and assembly levels, to set breakpoints, to monitor individual variables and to watch events during code execution.

To run your application, select Resume from the Run menu, or alternatively click the Resume button in the toolbar.
6.3 Creating your first application using TASKING toolchain

The debug session is launched as follows:

1. Open TASKING VX-Toolset for ARM Cortex. The program launches and requests the workspace location. Browse to select an existing workspace, or enter a new workspace location and click OK to confirm.

   ![Figure 52. TASKING Workspace Launcher dialog box](image)

2. When the TASKING toolset displays the Welcome window, click Go to workbench to open the main window. From the File menu, select New > TASKING VX-toolset for ARM C/C++ Project.

3. In the New C/C++ Project dialog box, enter the new Project name, then in the Project type box, select TASKING ARM Application and click Next.

   ![Figure 53. TASKING New C/C++ Project dialog box](image)
4. From the list of supported devices, select either STMicroelectronics > STM32L152>STM32L152RC as shown below in Figure 54.

**Figure 54. Processor selection**

![Processor selection](image1)

1. When using on STM32L-DISCOVERY, select STM32L152RB.

5. To configure the project for STM32L1xxDISCOVERY board, select Debug > Debug configurations and choose STMicroelectronics STM32L Discovery Kit. Choosing STMicroelectronics STM32L Discovery Kit as the evaluation board, automatically adds the needed linker file and configures the project as follows:
   - Microcontroller: STM32L152RC
   - Debug probe: ST-LINK
   - Connection: Serial Wire Debugging (SWD).

**Figure 55. Debug configurations**

![Debug configurations](image2)

1. The above example is given for the 32L152CDISCOVERY board.
6. To add source file to your project, right-click on the project from the C/C++ project window and select **Import**.
7. From the **Import** dialog box, select **General** and the desired file as shown in **Figure 56: TASKING Import dialog box**.
8. Click **Next**. Fill the displayed window as following and then browse to your source file.

**Figure 56. TASKING Import dialog box**

![TASKING Import dialog box](image)

8. Click **Next**. Fill the displayed window as following and then browse to your source file.

**Figure 57. Adding a new source file window**

![Adding a new source file window](image)

9. Select **main.c** and click **Finish**.

10. To build your project, click on **Project > Build Project** from the toolbar menu.

11. If your project is compiled successfully, the message shown in **Figure 58** is displayed.
12. Before running your application, connect your STM32L1xxDISCOVERY board as described in Section 1: Getting started.
# Revision history

## Table 1. Document revision history

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>27-Sep-2011</td>
<td>1</td>
<td>Initial release.</td>
</tr>
<tr>
<td>04-Jul-2013</td>
<td>2</td>
<td>Updated EWARM, MDK-ARM and TrueSTUDIO release numbers and added TASKING toolchain in Section 1: Overview of STLINK/V2 interface.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Updated Figure 5: IAR Embedded Workbench debugger screen, Figure 12: General options, Figure 21: MDK-ARM µVision4 project successfully compiled, Figure 23: MDK-ARM IDE workspace, Figure 25: Device selection dialog box, Figure 30: Target Options dialog box - Target tab, Section 3.3.2: Configuring project options, Section 4.3.1: Managing source files, Section 5.1: Building an existing TrueSTUDIO project and Section 5.2: Debugging and running your TrueSTUDIO project, and Section 5.3: Creating your first application using TrueSTUDIO toolchain to support 32L152CDISCOVERY.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added Section 6: Using Altium TASKING toolset</td>
</tr>
</tbody>
</table>