

UM0985 User manual

Developing your STM32VLDISCOVERY application using the IAR Embedded Workbench software

Introduction

This document provides an introduction on how to use IAR Embedded Workbench[®] for ARM software development toolchain (EWARM version 5.40 and later) with the STM32VLDiscovery board.

It provides novice users of the IAR Embedded Workbench with some guidelines to build and run a sample program provided with this document. This should provide them with the skills required to create and build their own applications.

The DISCOVER project, referenced in this document, corresponds to the demo Flashed into the STM32VLDISCOVERY board during production. The project is available within the STM32VLDISCOVERY firmware package available at http://www.st.com/stm32vldiscovery.

Although this application note cannot show all the topics relevant to IAR Embedded Workbench tool, it demonstrates the first basic steps necessary to get started with the compiler/debugger.

Contents

1	About IAR Embedded Workbench for ARM toolchain				
2	Starting an existing IAR Embedded Workbench project				
3	Crea	nting your first application9			
	3.1	Source file management9			
	3.2	Project options configuration 12			
4	Revi	sion history			



1 About IAR Embedded Workbench for ARM toolchain

The IAR Embedded Workbench for ARM is a window-based software development platform that combines a robust and modern editor with a project manager and make facility tool. It integrates all the tools needed to develop embedded applications including C/C++ compiler, macro assembler, linker/locator, and a HEX file generator. The IAR Embedded Workbench helps expedite the development process of embedded applications by providing the following:

- Integrated development environment with project management tools and editor.
- Highly optimizing C and C++ compiler for ARM.
- Automatic checking of MISRA C rules (MISRA C:2004).
- ARM EABI and CMSIS compliance.
- Run-time libraries including source code.
- Relocating ARM assembler.
- Linker and librarian tools.
- C-SPY® debugger with ARM simulator, JTAG support and support for RTOS-aware debugging on hardware.
- RTOS plugins available from IAR Systems and RTOS vendors.
- User and reference guides in PDF format.
- Context-sensitive online help.



2 Starting an existing IAR Embedded Workbench project

Installing the **IAR Embedded Workbench** toolchain (using the default settings) results in the toolchain being installed in *C*:*Program Files**IAR Systems**Embedded Workbench 5.4* on the PC's local hard disk. To start an existing project, follow these steps:

1. Open IAR Embedded Workbench from the Start menu. *Figure 1* shows the names of the windows referred to in this document.

	File Edit View Project Tools Window Help
	Workspace ×
	Files 👫 📴
Files window	
Workspace window –	
	× Messages File
Build window —	

Figure 1. IAR Embedded Workbench windows



2. Select **Open -> Workspace...** from the File menu.

File Edit View Pro	oject Simulat	or Tools	Window Help	p
New	+	alp	OI I	
Open	•	File	C	TRL+O
Close		Worksp	ace	
Save Workspace		Header	/Source File C	TRL+MAJ
Close Workspace			Col P	
Save Save As Save All	CTRL+S			
Page Setup Print	CTRL+P		AR >	~
Recent Files Recent Workspace:	s Þ	<u></u> fo[
Exit				

Figure 2. Open workspace

3. From the **Open Workspace** dialog box, browse and find *DISCOVER.eww.*

Figure 3. Find Discover.eww

Open Workspac	:e					? 🔀
Look in:	C EWARMv5		~	G 🦻	۳. 📂	
My Recent Documents	DISCOVER.EW	N				
Desktop						
My Documents						
My Computer						
	File name:	DISCOVER.EWW			~	Open
My Network	Files of type:	Workspace Files (*.ew	w)		~	Cancel

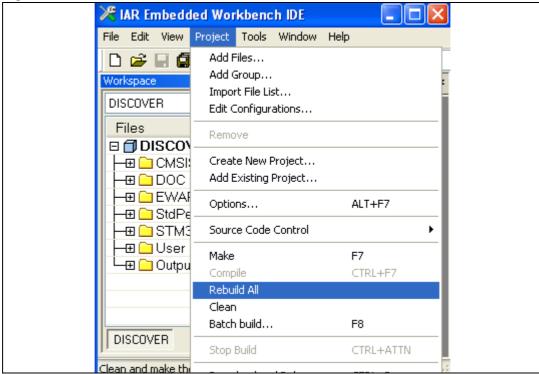


4. Double-click this file to launch it within the Project window

Figure 4.	Launch the project
	🔀 IAR Embedded Workbench IDE 📃 🗖 🔀
	File Edit View Project Tools Window Help
	- C 😂 🖬 🕼 👗 階 🛍 🗠 🗠 🗌
	Workspace × ×
	DISCOVER
	Files 82 Bit
	🗉 🗇 DISCOVER - DIS 🗸
	🔄 🛏 🗀 StdPeriph_Driver 🔹 🔹
	The state of the second
	🖌 🕂 🔁 🛄 User 💦 🔹 👘
	🖉 🖵 🗀 Output
	DISCOVER

5. Select **Rebuild All** from the Project menu.

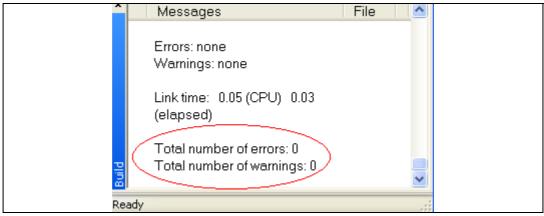




Doc ID 17875 Rev 1

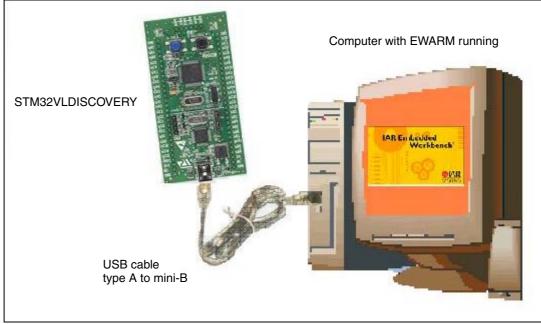


- 6. Your project is successfully compiled.
- Figure 6. Successful compile



7. Before running your application, you should establish the connection with the STM32VLDiscovery board as shown in *Figure 7*.





Note:

For more details on how to establish your hardware environment you can refer to the UM0919 User Manual: STM32VLDiscovery board available at www.st.com/stm32-discovery.

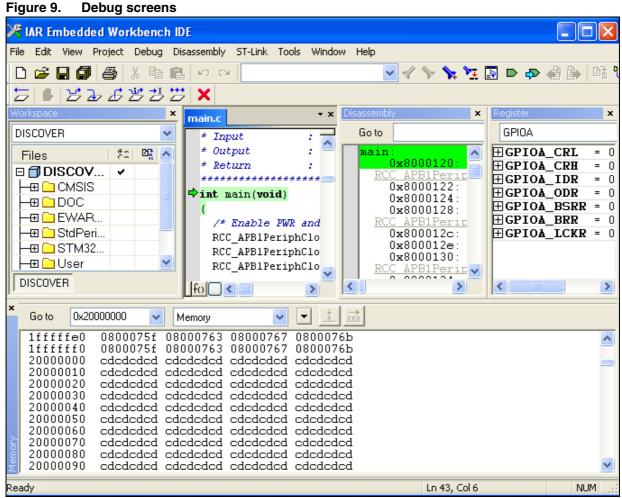
8. Choose **Project>Download and Debug**. Alternatively, click the **Download and Debug** button in the toolbar to program the Flash and begin debugging.

Figure 8. Download and debug a project





9. The debugger in IAR Embedded Workbench allows for source code debugging at C and assembly levels, breakpoint setting, individual variables monitoring and watching during the code execution.



10. Choose **Debug>Go**. Alternatively, click the **Go** button in the toolbar to run your application.

Figure 10. Go button

¥ Z 🖅 🗙

If everything is working correctly, LED3 should be blinking. Every time the user button is pressed, the LED3 blinking frequency changes and LED4 turns on.



3 Creating your first application

3.1 Source file management

1. Select Create New Project from the Project menu.

Figure 11. Create New Project

🔀 IAR Embedd	led Workbend	h IDE		
File Edit View	Project Tools	Window	Help	
Workspace	Add Files Add Group Import File Li: Edit Configur Remove Create New F Add Existing	st ations Project		• ×
	Options Source Code	Control	ALT+F7	•
Create a new proje	Make		F7	

2. Click on **OK**.

Figure 12. Create New Project OK

Create New Proje	ect			\mathbf{X}
Tool chain:	ARM		~	
Project templates: Empty project asm C++ C DLIB				
Externally built of Description: Creates an empty pr				
		ОК	Cancel	

- 3. Name the project **newproj.ewp**.
- 4. Click on Save.

Save As						? 🗙
Save in:	🚞 EWARM		*	G 🦻	ب 🔝 👏	
My Recent Documents						
My Documents						
My Computer	File name:	newproj.ewp			▼ (Save
S	Save as type:	Project Files (*.ewp)			~ (Cancel

Figure 13. Save newproj.ewp

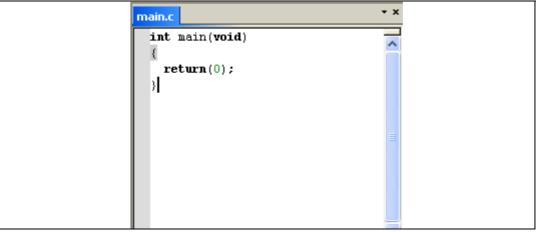
- 5. The IDE interface is displayed.
- Figure 14. IDE

🔀 IAR Embedded Workt	ench IDE	
File Edit View Project S	imulator Tools Window	Help
D 😅 🖬 🎒 🎒 X		
Workspace	x	• x
Debug	~	
Files 8a	0.	
🗇 newproj 🗸		
	_	
	-	
newproj		
Ready		.::



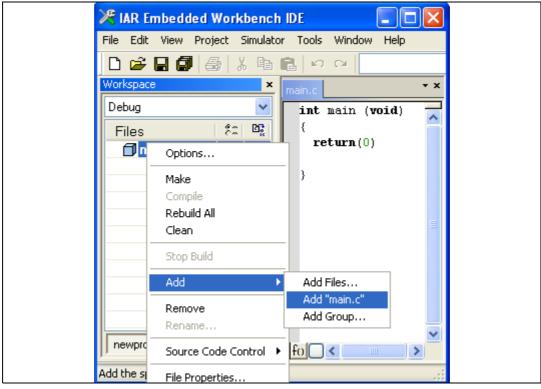
6. You may create a new source file with the menu File – New – File. This opens an empty editor window where you can enter your source code. IAR Embedded Workbench enables the C color syntax highlighting when you save your file with the dialog File – Save As... under a filename with the extension *.c . We are saving our example file under the name main.c.

Figure 15. Main.c



7. Once you have created your source file you can add this file to your project by selecting Add -> Add "main.c".

Figure 16. Add Files



Your *main.c* file is successfully added.



Figure 17. Successful creation of main.c..

Files 🛛 👫 🗠
🗆 🗇 newproj 🖌
🗕 🖃 🖻 main.c 🔹 🔹

3.2 **Project options configuration**

- 1. Right-click on the project name in the Project window.
- 2. Select **Options**.

Figure 18. Select options

Files 8 2 Pre-
Options
Make Compile Rebuild All
Clean

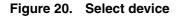
3. Select General Options - Target.

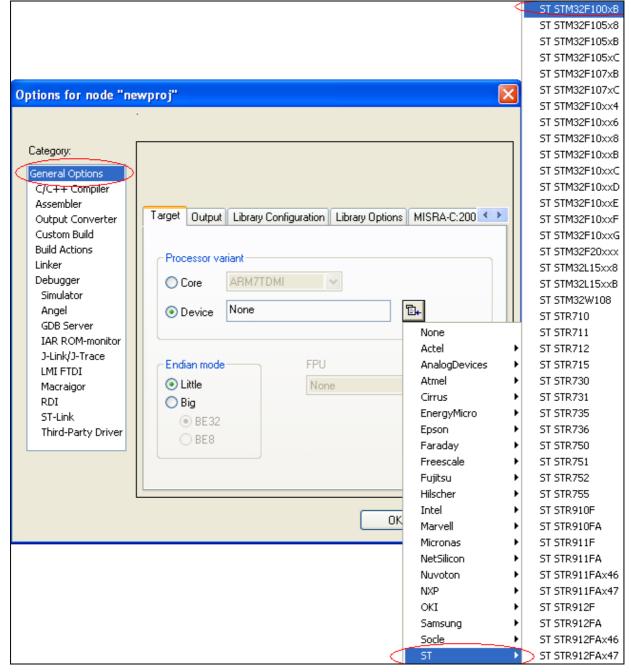
Figure 19. Target options

Options for node "ne	wproj"	×
Category: General Options C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator Angel GDB Server IAR ROM-monitor J-Link/J-Trace LMI FTDI Macraigor	Image: Output Library Configuration Library Options MISRA-C: 200 Processor variant Ore ARM7TDMI Ore Device None Endian mode FPU Output Little	×
Macraigur RDI ST-Link Third-Party Driver	BE32 BE8	



4. Select Device - ST -STM32F100xB.





5. Select Linker - Config.

Figure 21. Linker configuration options

	Assembler Output Converter Custom Build		Config Library Input Output List #define Diagnostics
	Build Actions		CLinker configuration file
\leq	Linker Debugger	2	Override default
	Simulator		\$TOOLKIT_DIR\$\CONFIG\generic.icf

- 6. Check Override default linker.
- 7. Click on Edit.
- 8. Set the .intvec.start at 0x08000000.

Figure 22. Save vector table

9. Fill memory regions.

Figure 23. Define memory regions

Linker configuration file editor 🛛 🔀			
Vector Table	Memory Regions	Stack/Heap Sizes	_
ROM RAM	Start: 0x08000000 0x20000000	End: 0x08020000]
		Save Cancel	

10. Click on **Save** to save the linker settings. The linker file is saved automatically under the project directory.



11. If your source files include header files, you should specify their paths in the project settings as shown in *Figure 24*.

Figure 24. Include header files

C/C++ Compiler Assembler	Discard Unused Publics
Output Converter	Language Code Optimizations Output List Preprocessor D
Custom Build	
Build Actions	Ignore standard include directories \$TOOLKIT_DIR\$\INC\
Linker	
Debugger	Additional include directories: (one per line)
Simulator	SPROJ_DIR\$\\inc >
Angel	
GDB Server	
IAR ROM-monitor	

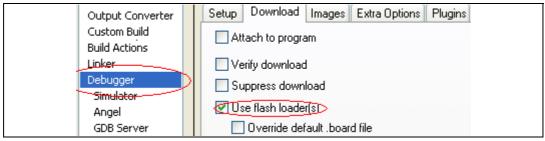
- 12. The path of the include directory is a relative path, it always starts by the project directory location referenced by \$PRO J_D IR\$
- 13. Select Debugger Set up.
- 14. Choose ST-Link from the Driver menu

Figure 25. Select ST-Link

Assembler Output Converter Custom Build	Setup Downloa	ad Images Extra Options Plugins
Build Actions	Driver	🗹 Run to
Linker	ST-Link	v main
Debugger		
Simulator	Setup macros	
Angel	Use mac	ro file(s)
GDB Server		

15. Select Download and check the Use Flash loader(s) case.

Figure 26. Use Flash loader(s)





- 16. Select ST-Link and SWD as connection protocol.
- Figure 27. Select SWD protocol

Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator Angel GDB Server IAR ROM-monitor J-Link/J-Trace LMI FTDI Macraigor RDI ST-Link Third-Party Drive	Interface JTAG SWD
---	--------------------------

- 17. Click on **OK** to save the project settings.
- 18. Build your project using Project -Rebuild All.
- 19. IAR Embedded Workbench then asks you to save your workspace. Save it under the name **newproj.eww**.

Figure 28. Save workspace

🚞 EWARM	🔽 🕝 🏂 📂 🛄 -	
Debug		
C settings		
Elle weener	newproj Save	
File name:		-
Save as type:	Workspace Files (*.eww)	



20. In the Build window it should report "0 Errors (s), 0 Warnings".

```
Figure 29. Success
```

×	Messages	File	>
	Errors: none Warnings: none		
	Link time: 0.05 (CPU) 0.03 (elapsed)		
Build	Total number of errors: 0 Total number of warnings: 0		-
R	eady		

- 21. Before running your application, you should establish the connection with the STM32VLDiscovery board as shown in *Figure 7: Hardware environment*.
- 22. Choose Project>Download and Debug to program the Flash and begin debugging.
- 23. Now you can start debugging your application.



4 Revision history

Table 1.Document revision history

Date	Revision	Changes
09-Sep-2010	1	Initial release.



Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST's terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST'S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER'S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

© 2010 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan -Malaysia - Malta - Morocco - Philippines - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com



Doc ID 17875 Rev 1