

UM2052 User manual

Getting started with STM32 MCU Discovery Kits software development tools

Introduction

The STM32 Discovery boards are low-cost and easy-to-use development platforms used to quickly evaluate and start a development with an STM32 device.

This document provides guidelines for beginners on STM32 MCU Discovery Kits 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 installation and firmware package presentation).
- The second chapter provides a 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 describes the first basic steps necessary to get started with the compilers and debuggers. It also offers links to the documents needed to fully understand every single step.

Contents

1	Syst	em requirements
	1.1	IDEs supporting STM32 families 6
	1.2	ST-LINK/V2 installation
	1.3	Firmware package
2	Exec	cuting and debugging firmware using the software toolchains 9
	2.1	EWARM toolchain
	2.2	MDK-ARM toolchain
	2.3	TrueSTUDIO toolchain
	2.4	SW4STM32 toolchain 15
3	Revi	sion history



List of tables

Table 1.	Most used integrated development environments	6
Table 2.	Document revision history	18



List of figures

Figure 1.	Hardware environment	5
Figure 2.	Firmware package content	8
Figure 3.	IAR embedded workbench IDE	
Figure 4.	EWARM project successfully compiled.	9
Figure 5.	Download and Debug button	0
Figure 6.	IAR embedded workbench debugger screen1	0
Figure 7.	Go button	0
Figure 8.	uVision5 IDE	1
Figure 9.	MDK-ARM project successfully compiled 1	1
Figure 10.	Start/Stop debug session button1	1
Figure 11.	MDK-ARM debugger screen	
Figure 12.	Run button	2
Figure 13.	TrueSTUDIO workspace launcher dialog box	2
Figure 14.	Atollic TrueSTUDIO import source select dialog box 1	3
Figure 15.	Atollic TrueSTUDIO import projects dialog box1	3
Figure 16.	TrueSTUDIO project successfully compiled 14	4
Figure 17.	TrueSTUDIO debug window14	4
Figure 18.	SW4STM32 workspace launcher dialog box1	5
Figure 19.	SW4STM32 import source select dialog box1	5
Figure 20.	SW4STM32 import projects dialog box1	6
Figure 21.	SW4STM32 project successfully compiled 1	
Figure 22.	SW4STM32 debug window1	7



Note:

1 System requirements

Before running the application, the user must:

- 1. Install the preferred Integrated Development Environment (IDE)
- 2. The ST-LINK/V2 driver is installed automatically. In case of problem, the user can install it manually from the toolchains install directory (further details are available in *Section 1.2: ST-LINK/V2 installation*).

This step is not needed for STM32VLDISCOVERY since it embeds an ST-LINK (not an ST-LINK/V2) which does not require a driver installation.

- 3. Download the STM32 Discovery firmware from the STMicroelectronics website at *www.st.com*.
- 4. Establish the connection with the STM32 Discovery board as shown hereafter. In *Figure 1*, the STM32F072 Discovery board is chosen as an example.



Figure 1. Hardware environment



The steps numbered above are detailed in the coming sections.

The minimum requirements to run and develop any firmware application on the STM32 Discovery board are:

- Windows[®] OS (XP, 7, 8) or Linux 64-bit or Mac OS[®] X
- USB type A to Mini-B cable, used to power the STM32 Discovery board (through USB connector CN1) from the host PC and connect to the embedded ST-LINK/V2 for debugging and programming.

1.1 IDEs supporting STM32 families

The 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 environments such as IDEs with C/C++ compilers and debuggers from major third-party companies, free versions of up to 64 KB of code (depending on partner) and completed with innovative tools from STMicroelectronics.

The following table includes general information about some integrated development environments as well as the versions supporting the STM32 products.

Toolchain	Company	Compiler	Version	Information ⁽¹⁾
EWARM ⁽²⁾	IAR Systems	IAR C/C++	7.60.1 and later	www.iar.com – 30-day evaluation edition KickStart edition – 16 Ko limitation for Cortex M0
MDK-ARM ⁽²⁾	Keil	ARMCC	5.18a and later	www.keil.com – MDK-Lite – 32 Ko code size limitation
TrueSTUDIO ⁽²⁾	Atollic	GNUC	5.5.1 and later	www.atollic.com – 30 days professional version trial – 32 Ko limitation – 8 Ko on Cortex-M0 and Cortex-M1
SW4STM32 ⁽³⁾	AC6	GNUC	1.8 and later	www.openstm32.org No limitation

Table 1. Most used integrated development environments

1. Register prior to downloading the toolchain.

2. On Windows only.

3. The SW4STM32 does not support ST-LINK. Consequently, STM32VLDISCOVERY can not be used with SW4STM32 software toolchain.

Information on the toolchain version supporting the STM32 devices is available on the toolchain release note at the third-party website.



1.2 ST-LINK/V2 installation

All the STM32 Discovery boards include the ST-LINK/V2 embedded debug tool interface. This interface requires to install the ST-LINK/V2 dedicated USB driver. The STM32 Discovery boards include an ST-LINK/V2 embedded debug tool interface that is supported by 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 the EWARM, the user should install the ST-LINK/V2 driver by running the ST-Link_V2_USB.exe from *IAR_INSTALL_DIRECTORYJ\Embedded Workbench x.x\arm\drivers\ST-Link \ST-Link_V2_USBdriver.exe.*

• RealView Microcontroller Development Kit for ARM (MDK-ARM) by Keil toolchain

The toolchain is installed by default in the C:\Keil directory on the PC local hard disk; the installer creates a start menu μ Vision5 shortcut.

When connecting the ST-LINK/V2 tool, the PC detects the new hardware and asks 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.

• AC6 SW4STM32 STM32

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 is installed automatically when installing the software toolchain.

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



1.3 Firmware package

The STM32 Discovery firmware applications, demonstration and IPs examples are provided in one single package and supplied in one single .zip file. The extraction of the .zip file generates the folder STM32-Discovery_FW_VX.Y.Z, which contains the subfolders shown in *Figure 2* where the STM32 F072B-Discovery_FW_VX.Y.Z is chosen as an example.

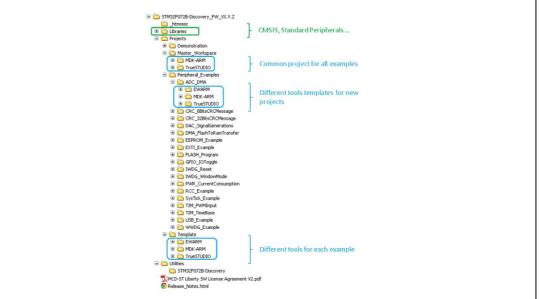


Figure 2. Firmware package content

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 the 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.



2 Executing and debugging firmware using the software toolchains

The user can follow the procedure described below to compile/link and execute an existing EWARM project. The following steps can be applied to an already existing example, demonstration or template project available at STM32 Discovery firmware package available on STMicroelectronics website. In this manual the STM32 F072B-Discovery_FW_VX.Y.Z is used as an example.

The user should go through the firmware/readme.txt file which contains the firmware description and hardware/software requirements.

2.1 EWARM toolchain

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

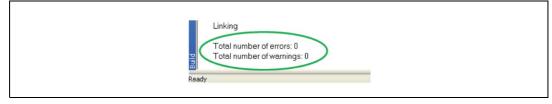
Figure 3 shows the basic names of the windows referred to in this document.

	FIAR Embedded Warkbench IDI			E 10 🔀
	File Edit View Project Tools Window Help			
	D # H Ø @ I B B P P	● 《子书注册》》》 《 》 》 》 》 》	23	
	e el gasti			10 - x
	Files 72 RL			
Workspace window —				
	$ \longrightarrow $			
File window				
Build window	_	\rightarrow		
	* Messages		File	Line
	\rightarrow			
	Featr	Errors D, Warnings D		NA

Figure 3. 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, the window shown in *Figure 4* is displayed.

Figure 4. EWARM project successfully compiled





To change the project settings (Include and preprocessor defines), the user should go through project options:

- For 'Include directories', select Project>Options...>C/C++ compiler>
- For pre-processor, define Project>Options...C/C++ compiler>pre-processor>
- 5. In the IAR Embedded Workbench IDE, from the **Project** menu, select **Download and Debug** or alternatively, click on the **Download and Debug** button the in tool bar, to program the Flash memory and start debugging.

Figure 5. Download and Debug button

File Edit View Project Tools Win	idow Help
🗅 🛩 🖬 🕼 🐇 🖪 🖻	L ∽ ∽ L
Workspace ×	Download and Debug

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

Project - JAR Emetoded Workersh DF - ARM Y01 File Set Vew Project Dolug Descendly Sflink Tools Window Help File Set Vew Project Dolug Descendly Sflink Tools Window Help Sinter Strate Control Strate Control Strate		5			
Image: State of the system clock to d8 ME */ System(control control () released by clock clock control () released by clock cloc	Project - IAR Embedded Workb	ench IDE - ARM 7.60.1			
Image: Second	File Edit View Project Debu	ig Disassembly ST-Link Tools Window Help			
SiMa32028.0hccomy * Sima32000.0hcc					
SM22B028-Discovery Files Files </th <th></th> <th></th> <th></th> <th></th> <th></th>					
Files #1 Big Files #1 Big Files #1 Big Consistencies Configures Static Unité Diference Configures #1 Big Configures #2 Config Configures #2 Config <t< th=""><th></th><th>main.c 🕯</th><th>f0 🕶 🗙</th><th></th><th></th></t<>		main.c 🕯	f0 🕶 🗙		
Project - S * * Disassening * Disassening * * * SystemClock_Config(): * * Messages * * * Messages * * * Messages * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * <			-	Go to • Memory	• 🗈
<pre>static void Error_Handler (void); int min(void) /* Brives functions</pre>	Files 👫		ath).	Disassembly	^
B: Example File B: Output * Strict lists * Sprame Sprame			gou, ,		
Coupui Coupui Constant None Sparsm None S		/* Private functions*/			DUCU
		* @brief Main program	E		
/ Ox80014e0: 0x4000 wtf802 EL # htt main (void) */* * for SM0370x Mil library initialization: - - Configure the Fissh properties 0x80014e0: 0x4000 wtf802 EL BSP_IDD_Init(LDDS) 0x80014e0: 0x4000 wtf802 EL BSP_IDD_Init(LDDS) 0x80014e0: 0x4000 wtf802 EL BSP_IDD_Init(LDDS) 0x80014e0: 0x4000 wtf822 EL BSP_IDD_Init(LDDS) 0x80014b0: 0x4000 wtf822 EL BSP_IDD_Init(LDS)					BL
* Messages * Messages * Messages * Messages * File * File					PT
Configure the Fissh properties Configure the fissh properties control with a source of time base, but user can eventually implement his proper time bases source of time base, but user can eventually implement his proper time bases source of time base, but user can eventually implement his proper time bases source of time base, but user can eventually implement his proper time bases source of time base, but user can eventually implement his proper time bases source of time bases outcome of time bases direction should be kept ins since FPF TIMEOUT_VALUEs are defined and handled in milliseconds basis. Low Level Initialisation '/ HaL_Init(); // Configure the system clock to 48 MHs */ SystemClock_Config(); * Messages * Messages File Line * Messages File Line * Messages Total number of errors:0					DE
Configure the Fissh properties Configure the fissh properties control with a source of time base, but user can eventually implement his proper time bases source of time base, but user can eventually implement his proper time bases source of time base, but user can eventually implement his proper time bases source of time base, but user can eventually implement his proper time bases source of time base, but user can eventually implement his proper time bases source of time bases outcome of time bases direction should be kept ins since FPF TIMEOUT_VALUEs are defined and handled in milliseconds basis. Low Level Initialisation '/ HaL_Init(); // Configure the system clock to 48 MHs */ SystemClock_Config(); * Messages * Messages File Line * Messages File Line * Messages Total number of errors:0				0x80014a4: 0x2002	MOVS
- Systick timer is configured by default as source of ine base, but user or aventually implement his proper time bases source (a general purpose dimersion should be kept as since PEP_TIMEOUT_VALUEs are defined and bendled in millseconds basis. - Low level Initialization '' Low level Initialization '' HAL_Init(); 'Configure the system clock to 46 Mit */ SystemClock_Config(); * Messages stm320bx_kC systemClock_Config(); * Messages Stm320bx_kC SystemClock config(); Total number of errors: 0					BL
con eventually implement his proper time base source (e general purpose duration should be kept ins since PPP_TIMEOUT_VALUES are defined and handled in milliseconds basis. - Low Level Initialization 0x80014ab: 0x2000 0x1d25 BL BSP_IED_Init(IEDD) 0x80014b: 0x2000 0x1d23 BL INITIAL (); // Configure the system clock to 48 MHz */ SystemClock_Config(); 0x80014b: 0x2000 STR 0x80014b: 0x4038 LIR.N 0x80014b: 0x4038 STR 0x80014b: 0x800 STR 0x80014b: 0x800 STR 0x800 STR 0x800 STR 0x800 S					NORE
Image: State for example or char time source), keeping in mind that Time bases BES LED. Init(LED3): Image: State for example or char time source), keeping in mind that Time bases BES LED. Init(LED3): Image: State for example or char time source), keeping in mind that Time bases BES LED. Init(LED3): Image: State for example or char time source), keeping in mind that Time bases BES LED. Init(LED3): Image: State for example or char time source), keeping in mind that Time bases BES LED. Init(LED3): Image: State for example or char time source Image: State for example or keeping in mind that Time bases Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source Image: State for example or char time source <td< td=""><td></td><td></td><td></td><td></td><td></td></td<>					
handled in millseconds basis. - - -					
* * <td></td> <td></td> <td></td> <td></td> <td></td>					
BAL_Init(); 0x80014bi: 0x4238 LDR N /* Configure the system clock to 48 Mit */ 0x80014bi: 0x4638 LDR N ystemClock_Config();					
V Ox8001bit 0x8021bit 0x8021bit <td></td> <td></td> <td></td> <td></td> <td></td>					
/* Configure the system clock to 49 Mfiz */ 0x8001 the: 0x602 0 STR Projecti /* * Messages stm200x_itc system_clock_config(); system_clock_config(); * * File Linking File Projectut Convening Convening File Total number of enors: 0 Total number of enors: 0 Total number of enors: 0 *		HAL_Init();			
Project m m * Messagas stra200x.tc system_stra200x.c Linking Projectuit Convening Total number of enors: 0 Total number of enors: 0 Total number of enors: 0 E • • • • • • •		/* Configure the system clock to 48 MHz */			
Image: Second		SystemClock_Config();			= EEPRC -
strat20bc_tc system_stra20bc_tc Linking Projectout Converting Total number of errors: 0 Total number of warnings 0	Project	€ II	+ =	<	۲.
system_stm32t0xcc Linking Project out Converting Total number of errors: 0 Total number of warnings. 0	* Messages			File	Line ^
Linking Project out Converting Total number of errors: 0 Total number of warnings: 0 Protect out Converting Total number of warnings: 0 X					
Popiečout Converting Total number of errors: 0 Total number of warnings. 0					
Converting Total number of errors: 0 Total number of warnings: 0 Pobug tag Buld X					
Total number of errors: 0 Total number of warnings: 0					
Didai number of warnings. U	j				
Bebug Log Buld					=
		5: 0			*
Ready Errors 0, Warnings 0 Ln 95, Col 1 System NUM Errors 0, Warnings 0 Ln 95, Col 1 System NUM	Debug Log Build				×
	Ready	Errors 0,	, Warnings 0	Ln 95, Col 1 System N	IUM MU

Figure 6. IAR embedded workbench debugger screen

To run the application, from the **Debug** menu, select **Go**. Alternatively, click on the **Go** button in the toolbar to run the application.

	Figure 7. (Go button	
×	SysTick - IAR Embe	dded Workbench IDE	
File	Edit View Project	Debug Disassembly ST-	
	🖻 🖬 🕼 付	X B B N N	
L L	5 🕒 B Z B 3	当 <u>出</u> 罢×	
ETI	1 SWO	Go	
Wo	rkspace	× main.c	I

DocID029205 Rev 1



2.2 MDK-ARM toolchain

1. Open Keil MDK-ARM Microcontroller Kit,

Figure 8 shows the basic names of the "Keil uVision5" windows referred to in this document.

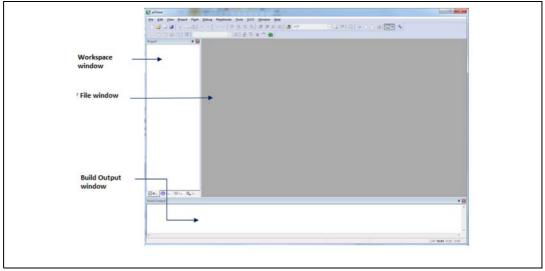
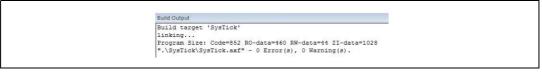


Figure 8. uVision5 IDE

- 2. In the **Project** menu, select **Open Project**. Browse to select either an example or demonstration or 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 window shown in *Figure 9* is displayed.

Figure 9. MDK-ARM project successfully compiled



If the user needs to change the project settings (Include and preprocessor defines), the user should go through the project options:

- For 'Include directories', select **Project>Options for Target > C/C++ > Include Paths**
- For pre-processor definition Project>Options for Target > C/C++ > Preprocessor symbols > Define
- 5. In the MDK-ARM IDE, from the **Debug** menu, select **Start/Stop Debug Session** or, alternatively, click the **Start/Stop Debug Session** button in the tool bar to program the Flash memory and begin debugging.

Figure 10. Start/Stop debug session button

D 🐸 🖬 🎒 💰 🗤 🕯	1 日本 1 日本 1 日本 1 日本	👷 譚 譚 //É //授 🧭 GetBase 💽 🗽 🖗 🔍 🔍 🔿 🔗 🇶 💽 🔍
🧼 🖽 🎬 🧼 📇 🞇	Systick 💽 🔊 🛔 🖷	Start/Stop Debug Session (Ctrl+F5)
Project 🛛 📮 📴		Enter or leave a debug session

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.



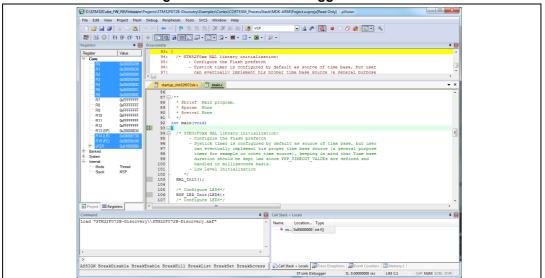


Figure 11. MDK-ARM debugger screen

To run the application, from the **Debug** menu, select **Run**. Alternatively, click the **Run** button in the toolbar to run the application.

Figure 12. Run button

R 🖻 🛞 🔁 🔂 🖓
Registers Registers Register R

2.3 TrueSTUDIO toolchain

1. Open Atollic TrueSTUDIO for ARM product. The program launches and asks for the Workspace location.

Workspace La	Launcher		100	X	1
Select a works Atollic TrueSTU Choose a work	(space TUDIO® for ARM® Pro stores you rkspace folder to use for this sessic	ir projects in a folder called a on.	workspace.		
Workspace: -C Copy Setting	-Discovery_FW\Projects\Periphera gs	ıl_Examples\SysTick_Example\	\TrueSTUDIO ▼	Browse	
?			ок (Cancel	

Figure 13. TrueSTUDIO workspace launcher dialog box

- 2. Browse to select a TrueSTUDIO workspace of either an example or demonstration or 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**.

DocID029205 Rev 1



Select Create new projects from an archive file or directory. Select an import source: type filter text
type filter text

Figure 14. Atollic TrueSTUDIO import source select dialog box

5. Click **Select root directory**, browse to the TrueSTUDIO workspace folder.

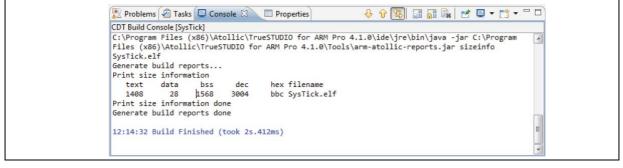
Figure 15. Atollic TrueSTUDIO import projects dialog box

Import		- • ×
Import Projects Select a directory to se	earch for existing Eclipse projects.	
 Select root directory Select archive file: Projects: 	C:\STM32F072B-Discovery_FW_VX.Y.Z\Projects	Browse Browse
SysTick_Examp Copy projects into Working sets Add project to we Working sets		Select All Deselect All Refresh Select
?	< Back Next > Finish	Cancel



- 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.

Figure 16. TrueSTUDIO project successfully compiled



If the user needs to change the project settings (Include directories and preprocessor defines), go through Project>Properties, 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, the user should select the project In the Project Explorer and press F11 to start a debug session (see *Figure 17*).



Debug - STM32F072B-Discovery/Example/User/main.c - Atollic TrueSTUDIO for ARM					
ile Edit View Run Window Help		- Anna Anna Anna Anna Anna Anna Anna Ann			
	- - 1 🖬 - 1	I - X C			
		Quick Acces	s	😭 💀 C/C++	- 🕸 Debug
			·]		processing
🅸 Debug 🛛 🙀 ▽ 🗖 🗖	(×)= Variables 💥 🧕	Breakpoints 1999 Regis			
 STM32F072B-Discovery.elf [Embedded C/C++ Application] R STM32F072B-Discovery.elf 		1_	🐑 📲 📃		📫 🛃 🔍
▲ m [®] Thread [1] < main> (Suspended : Breakpoint)	Name	Туре		Value	
main() at main.c:96 0x8000334					
📙 gdb 🛃 ST-LINK					
					1.0
					÷
	•				•
i main.c ⊠			E Outline	x	
93 * @cetval None		*	69	🖻 🞼 😿 🖋	● # ▽
94 */ 95⊖ int main(void)			🖬 m		*
5 96 { 97⊖ /* STM32F0xx HAL library initialization:				PROM_ADDRESS	
98 - Configure the Flash prefetch				PROM_TIMING	
 99 - Systick timer is configured by default as source of ti 100 can eventually implement his proper time base source (C_XFER_TIMEOUT	-
101 timer for example or other time source), keeping in mi	nd that Time base			PROM_MAX_TRL cHandle : I2C Hai	
102 duration should be kept 1ms since PPP_TIMEOUT_VALUEs a 103 handled in milliseconds basis.	re defined and			[xBuffer : uint8_t[]	2.1
104 - Low Level Initialization 105 */		-		&Buffer : uint8_t[]	
••••		Þ	<	emorv Address : III	
🛢 Console 💥 🔎 Tasks 🖹 Problems 👔 Executables 👔 Memory 🛛 🗖		🕱 📃 SWV Console		× • ×	
	B SVVV Trace Log	Svvv Console		$\Delta = A$	
Image: STM32F072B-Discovery.elf [Embedded C/C++ Application] gdb	Index Ty	pe Data		Cycles	Time(s)
· · · · · · · · · · · · · · · · · · ·					
<pre>Temporary breakpoint 1, main () at\\Src\main.c:96 96 { </pre>	•	III			•
4	Overflow packets:	0			
Read-Only	Smart Insert	100 : 28			

Figure 17. 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.



2.4 SW4STM32 toolchain

1. Open the AC6 SW4STM32 for ARM product. The program launches and prompts for the Workspace location.

Workspace	
Select a wor	The second se
	s your projects in a folder called a workspace. rkspace folder to use for this session.
Workspace:	cts\STM32F072B-Discovery\Examples\ADC\ADC_DMA_Transfer\SW4STM32 + Browse
 Copy Setti 	ngs
?	OK Cancel

Figure 18. SW4STM32 workspace launcher dialog box

- 2. Browse to select a SW4STM32 workspace of either an example or demonstration or 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**.

elect Create new projects from an archive file or director	2 J
Select an import source:	
type filter text	
 General [®] Archive File [®] Existing Projects into Workspace [®] File System [®] Preferences [®] C/C++ [®] C/S [®] C/S [®] Git [®] Archive File [®] C/S [®] Remote Systems [®] Remote Systems [®] Remote Systems [®] Remote Systems [®] Trans [®] Trans [®] Trans [®] C/S [®] C/S	Finish Cancel

Figure 19. SW4STM32 import source select dialog box

5. Click **Select root directory**, browse to the SW4STM32 workspace folder.

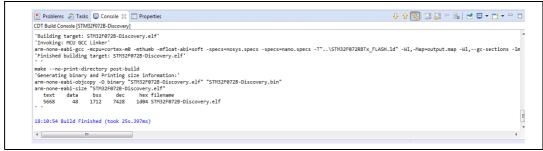


Import Projects	1
6 Some projects cannot be imported because they already exist in the workspace	<u>/</u>
Select root directory: C:\STM32\STM32Cube_FW_F0_V1.2.0\Project\STM32F0728-Discovery\Examples\ADC\ADC_DMA_Transfer\SW4STM32 +	Browse
Select archive file:	Browse
Projects:	
RemoteSystemsTempFiles (C:\STM32\STM32Cube_FW_F0_V1.2.0\Projects\STM32F0728-Discovery\Examples\ADC\ADC_DMA_Transfer\SW4STM32\RemoteSystemsTempI	Select /
STM32F0728-Discovery (C:\STM32\STM32\STM32Cube_FW_F0_V1.2.0\Projects\STM32F0728-Discovery\Examples\ADC\ADC_DMA_Transfer\SW4STM32\STM32F0728-Discovery)	Deselect
	Refres
۲ (
c m popions Search for nested projects	
Options Search for nested projects Copy projects into workspace	
Options Search for nested projects Copy projects into workspace Hide projects that already exist in the workspace	
Options Generation Gene	
Options Copy projects into workspace Inde projects that already exist in the workspace Working sets Add project to working sets	
Options Generation Gene	Select
Options Copy projects into workspace Inde projects that already exist in the workspace Working sets Add project to working sets	Select
Options Copy projects into workspace Inde projects that already exist in the workspace Working sets Add project to working sets	Select
Options Copy projects into workspace Inde projects that already exist in the workspace Working sets Add project to working sets	Select

Figure 20. 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.





If the user needs 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 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 22*.



🗱 Debug 🔀 🦹 🦌 🗢 🗖	(x)= Varia 💊 Brea 🐼 Expr	🔀 👫 Regis 🗰 I/O 🛋 Modu 🖳 🗖
 STM32F072B-Discovery.elf [Ac6 STM32 Debugging] 	Expression	🏠 🍕 🖻 🖶 💥 🦹 🗂 🔽
 	4 Add new expression	
	< [
< •	٠	4
e main.c 🕱 c (gdb[0].proc[42000].threadGroup[i1],gdb[0].proc[42000].OSthread[1]).	thread[1].frame[0]	🗖 🗖 📴 Outline 🛛 🗖 🗖
<pre>86 - Low Level Initialization 87 */ 83 88 HAL_Init(); 89 90 /* Initialize LEDS */ 91 B5P_LED_Init(LED5); 92 4</pre>		Imain.h AdcHandle: ADC_HandleType Sconfig: ADC_ChannelConfly aResultDMA: uint32,t
📮 Console 🔀 🧔 Tasks 🖹 Problems 🜔 Executables 🚦 Memory	■ × %	
STM32F072B-Discovery.eff [Ac6 STM32 Debugging] (:/Ac6/SystemWorkbench/plugins/fr. Find the GDB manual and other documentation resources online at: <htp: documentation="" gdb="" software="" www.gnu.org=""></htp:> . For help, type "help". Type "apropos word" to search for commands related to "word". Warning: the current language does not match this frame.	ac6.mcu.externaltools.arm-none.win3	2_1.3.0.201507241045/tools/compiler/bin/arm-none

Figure 22. 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 the application, from the **Run** menu, select **Resume**, or alternatively click the **Resume** button in the toolbar.



3 Revision history

Table 2. Document revision	history
----------------------------	---------

Date	Revision	Changes
20-May-2016	1	Initial release.



IMPORTANT NOTICE - PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2016 STMicroelectronics – All rights reserved



DocID029205 Rev 1