AVR ONE!

Quick-start Guide

.....

EVK1101 + Windows®



.....

.....

Section 1

Introducti	on	1-1
1.1	General	1-1
1.2	Requirements	1-1

Section 2

Quick-	start guide (short version)	2-1
	1 Install Hardware and software	
2.	2 Create a demonstration project	
2.	3 Configure target MCU for a debug session using trace	
2.	4 Start the debug session and configure AVR32 Studio 2.5 for trace	
2.	5 Start the trace debug session	

Section 3

Software	Installation	4-1
3.1	Download the software	4-1
3.2	Download the two installation files to your disk	4-2
3.3	Install AVR32 GNU Toolchain	4-2
3.4	Install AVR32 Studio 2.5	4-8
3.5	Connect the AVR ONE! to power and USB host	4-11
3.6	Install AVR ONE! Driver	4-12

Section 4

4.1	Connect the AVR ONE! to the EVK1101 5-1
4.2	Connect the EVK1101 to power and RS2325-2

Section 5

Create d	lemo ap	plication	6-1
5.1	Start A	VR32 Studio	6-1
5.2	Configu	ure adapter and target	
	5.2.1	Add and configure the adapter (AVR ONE!)	6-3
	5.2.2	Configure target board and MCU	6-4
	5.2.3	Target MCU Chip erase	6-5
5.3	Create	a demonstration project	6-6
5.4	Configu	ure AVR32 Studio for a debug session using trace	6-9
	5.4.1	Create a new debug launch configuration	6-10
	5.4.2	Configure the target trace module for program trace	6-11
	5.4.3	Configure the target trace module for data trace	6-14
5.5	Start a	debug session and configure the debugger for trace	6-16
5.6	Add sta	art and stop trace-points	6-17
5.7	Start th	e trace debug session	6-20
5.8	View tra	ace data	6-24



i

Table of Contents (Continued)

5.9	Modify the application
Section 6	6
Firmware	Upgrade7-1
6.1	Firmware upgrade overview7-1
6.2	Firmware version test and upgrade7-1
6.3	Adapter in use



ii



Section 1

Introduction

1.1 General

This document contains a quick-start guide describing how to get up and running using the AVR[®] ONE! debugger with AVR32 Studio. In addition to the AVR ONE! debugger, you need the following items:

- AVR32 Studio 2.5 software
- AVR32 GNU Toolchain 2.4
- EVK110x Evaluation board

Software and documents can be found at www.atmel.com/avrone

1.2 Requirements

This example was created on a PC running Microsoft[®] Windows[®] XP Professional. For other versions of Windows, the behaviour when installing software and drivers may be slightly different.

Please read the AVR32 Studio 2.5 release notes for information about support for other versions of Windows.



Quick-start guide (short version)

2.1 Install Hardware and software

- Download and install avr32-gnu-toolchain-2.4.x and AVR32Studio-2.5.x.
- Connect AVR ONE! to power and USB and turn it on.
- Install AVR ONE! USB driver.
- Connect AVR ONE! to the EVK1101 using the 10pin JTAG connector.
- Connect the EVK1101 to power and turn it on.
- Start AVR32 Studio.
- Select a suitable workspace folder to contain your projects.
- Exit from the welcome screen to workbench.
- Right-click in the *AVR32 Targets* view and select **Scan Targets**.
- Select the AVR ONE! and click on the *Properties*-tab.
- Select Board-tab. Set Board to EVK1101, MCU to UC3B0256 or UC3B256ES, depending on what MCU is mounted on your EVK1101.
- Right-click on the AVR ONE! in the AVR32 Target view and select Chip Erase. This operation is only needed one time (when the EVK1101 is new).

2.2 Create a demonstration project

- Select File>New>Example.
- Select EVK1101>Components>Accelerometer example, then Next.
- Enter a name for the project, and click **Finish**.
- Right-click on the project in *Project Explorer* view and select **Build Project** (or use Ctrl+B).

2.3 Configure target MCU for a debug session using trace

- When the build process is finished, right-click on the project in the *Project Explorer*-view and select *Debug As>Debug Configurations*.
- In the Debug-view, select AVR32 Application and click New. A new launch configuration will be created and default values will be filled into all fields.
- Select the *Trace*-tab and click **Enable Trace**.
- Select the preferred trace method. In this case we want **Nano Trace**.
- Select the preferred action when buffer is full. In this case we choose Break, read out and halt.
- Deselect the option Break on application buffer access
- Set Buffer Size. Select **Specify size and location**, then click **Detect**.

2.4 Start the debug session and configure AVR32 Studio 2.5 for trace

- Click the **Debug**-button. Now the program will be loaded into the target, and run until main().
- When the program halts, add at least a trace start-point (Right-click to the left of the source code line in the source code view).

2.5 Start the trace debug session

- Click **Resume** (green *Play* button in Debug view) and wait until the program halts.
- You can now look at the trace data in the *Trace*-view.





Section 3

Software Installation

3.1 Download the software

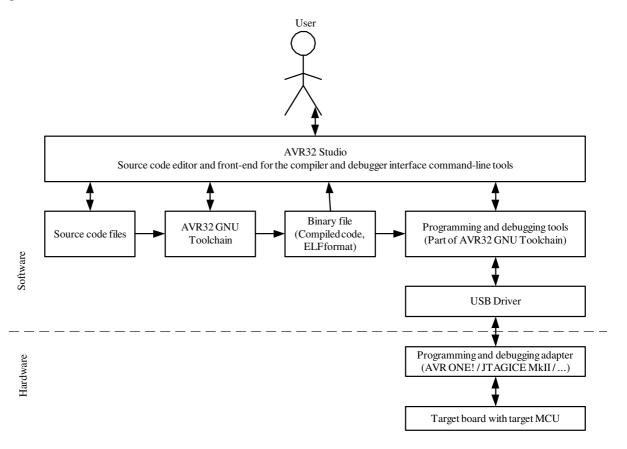
To use the AVR ONE!, you must download and install two software packages:

- avr32-gnu-toolchain-2.4.x.exe
- AVR32Studio-2.5.x.exe

The AVR32 Toolchain is a collection of tools that are required to be able to work with the AVR ONE! It contains command-line tools for controlling the AVR ONE!, and tools to compile code for the AVR32 MCUs.

AVR32 Studio is the front end that uses the AVR32 GNU Toolchain to generate binary code for the target, program the target, and control the debug sessions.

Figure 3-1. Tools structure



3.2 Download the two installation files to your disk.

The installation files can be found at this location: www.atmel.com/avrone

3.3 Install AVR32 GNU Toolchain

If you have any AVR tools connected to the USB hub, turn them off now. Otherwise the USB driver installation may fail.

Double-click on avr32-gnu-toolchain-2.4.x to start the installation process.

Figure 3-2. AVR32 GNU Toolchain installation welcome

AVR32 Toolchain - InstallShie	eld Wizard	×
	Welcome to the InstallShield Wizard for AVR32 Toolchain Version: 2.4.2	
	< Back Next > Cancel	

Click Next.



Figure 3-3. AVR32 GNU Toolchain License Agreement form

Please read the following license agreement carefully. The AVR32 GNU Toolchain Package contains open source software packages with individual license agreements. You must ensure that you comply to the individual software package's licenses before using or distributing it. All open source software packages are distributed in source form on the AVR32 Board Support Package CD-ROM in the source directory. THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE EXPRESSITY AND SPECIFICALLY DISCLAIMED, IN NO EVENT SHALL ATMELE BE LIABLE FOR ANY DIRECT, INDIRECT, INDIRECT, INDIRECT, OR CONSTRUCTIONED UNTED TO, THE MAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DISCLAIMED INTOR DISCLAIMED AND ON ANY TURGED AND ANY TURGED A	icense Agreement Plasse read the following license agree	ement e zrefullu	
with individual license agreements. You must ensure that you comply to the individual software package's licenses before using or distributing it. All open source software packages are distributed in source form on the AVR32 Board Support Package CD-ROM in the source directory. THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OP DISPUSSOR HE terms of the license agreement	Fledse ledu (Ne followning incense agree	ement caleidily.	
Individual software package's licenses before using or distributing it. All open source software packages are distributed in source form on the AVR32 Board Support Package CD-ROM in the source directory. THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR DISUBSCENTED BUTCH DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR DISUBSCENTED BUTCH DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR DISUBSCENTED BUTCH DAMAGES ARE EXPRESSED AND DISTANCE UNDITIED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR DISUBSCENTES OF THE INFORMATION OF DAMAGES (INCLUDING DAMAGES) (INCLUDING) (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR DISUBSCENTES (INCLUDING)	The AVR32 GNU Toolchain Package	e contains open source software packages	
open source software packages are distributed in Source form on the AVR32 Board Support Package CD-ROM in the source directory. THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL, DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OD DISCUMPTOR INFORMATION OF CONSECUENCES, DATA, OR PROFITS; DE DISCUMPTOR INFORMATION OF CONSECUENCES, DATA, OR PROFITS; DE DISCUMPTOR INFORMATION OF CONSECUENCES, DATA, OR PROFITS; DISCUMPTOR INFORMATION OF CONSECUENCES, DATA, OR PROFITS; DISCUMPTOR INFORMATION OF THE INFORMATION OF CONSECUENCES, DATA, OR PROFITS; DISCUMPTOR INFORMATION OF CONSECUENCES, DATA, DISCUMPTOR INFORMATION OF CONSECUE			
THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DO DISCUSSED INTERCOMPTON DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DO DISCUSSED INTERCOMPTON DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DO DISCUSSED INTERCOMPTON DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DO DISCUSSED INTERCOMPTON DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DO DISCUSSED INTERCOMPTON DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DO DISCUSSED INTERCOMPTON DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DO DISCUSSED INTERCOMPTON DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DO DISCUSSED INTERCOMPTON DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROFILE INTERCOMPTON DAMAGES (INCLUDING, BUT NOT LIMITED	open source software packages are o	distributed in source form on the AVR32	
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE EXPRESSIV AND SPECIFICALLY DISCLAMED. IN NO EVENT SHALL ATMEL BE LABLE FOR ANY DIRECT, INDIRECT, IND	Board Support Package CD-ROM in t	he source directory.	
FITNESS FOR A PARTICULAR PURPOSE ARE EXPRESSLY AND SPECIFICALLY DISCLAMED. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DB DISCUSSED INTO THE CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DD DISCUSSED INTO THE INFORMATION OF ANY DISCOVERY OF UNDER THE DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; DD DISCUSSED INTO THE INFORMATION OF ANY DISCOVERY OF UNDER THE DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROFILE PROFILE O I go not accept the terms of the license agreement Print			
IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, INDIRE			
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; Image: service of the license agreement Image: service of the license agreement	IN NO EVENT SHALL ATMEL BE LIA	ABLE FOR ANY DIRECT, INDIRECT, INCID	DENTAL,
I go not accept the terms of the license agreement I go not accept the terms of the license agreement			
I go not accept the terms of the license agreement		NUCLED CALLERS AND ON ANY THEODY	
OI do not accept the terms of the license agreement	I accept the terms of the license ac	reement	Print
allChield	I do not accept the terms of the lice	ense agreement	
	IIShield		
		< Back	Next > Cancel

Select I accept the terms of the licence agreement, then click Next.

Figure 3-4. AVR32 GNU Toolchain installation folder select

AVR32 Too	olchain - InstallShield Wizard	X
Installatio	on folder	
<u></u>	Install AVR32 Toolchain to: C:\\Atmel\AVR Tools\AVR32 Toolchain	Change
InstallShield -	< <u>B</u> ack Next >	Cancel

Check that the installation folder is correct and click Next.



Figure 3-5. AVR32 GNU Toolchain installer configuration finished

AVR32 Toolchain - InstallShield Wizard	
Ready to Install the Program	
The wizard is ready to begin installation.	
Click Install to begin the installation.	
If you want to review or change any of your installation settings the wizard.	, click Back. Click Cancel to exit
nstallShield	
< Back	Install Cancel

Click Install.

Figure 3-6. AVR32 GNU Toolchain installation progress indicator

AVR32 Toolchain - InstallShield Wizard	
Setup Status	
AVR32 Toolchain is configuring your new software installation.	
Installing	
C:\\AVR32 Toolchain\avr32\lib\ldscripts\avr32elf_uc3b0256es.xwr	
InstallShield	
	Cancel

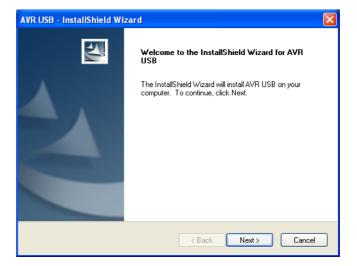
The AVR32 GNU Toolchain is now being installed. As a part of the installation process, USB drivers for all supported programming and debugging adapters are installed.



Figure 3-7. USB Drivers installation start

VR32 Toolchain - Ins Setup Status	tallShield Wizard	×
AVE 22 T	· · · · · · · · · · · · · · · · · · ·	
AVH32 Toolchain is cor	figuring your new software installation.	
Installing	Installing USB drivers	
stallShield		Cancel

Figure 3-8. USB Driver installer welcome



Click Next.



Figure 3-9. USB Drivers licence agreement form

AVR USB - InstallShield Wizard	×
License Agreement Please read the following license agreement carefully.	
Welcome to AVR USB drivers from Atmel Corporation. The tools are free of charge and may be freely copied and distributed in its original form. The tools runs under Microsoft Windows 98, Microsoft Windows 2000, Microsoft Windows XP , Microsoft Windows XP 64, Microsoft Windows Vista and Microsoft Windows Vista 64. Copyright © ATMEL Corporation. All rights reserved. AVR is trademark of ATMEL Corporation Windows is a trademark of Microsoft Corporation	
Print Cancel)

Select I accept the terms of the licence agreement, then click Next.

Figure 3-10. USB drivers installer configuration finished

AVR USB - InstallShield Wizard	
Ready to Install the Program The wizard is ready to begin installation.	2
Click Install to begin the installation.	
If you want to review or change any of your installation settings, click Back. Click the wizard.	Cancel to exit
nstallShield	
< Back Instruction	Cancel

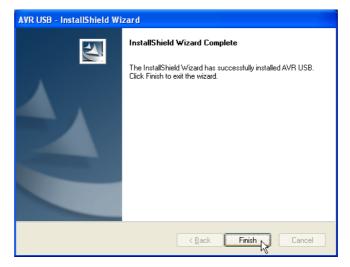
Click Install.



Figure 3-11. USB Drivers installation progress indicator

Setup Status	
The InstallShield Wizard is installing AVR USB	
Installing	
C:\Program Files\Atmel\AVR Tools\usb\windrvr6.inf	
stallShield	
	Cancel

Figure 3-12. USB Drivers installation complete



Click Finish.





Figure 3-13. AVR32 GNU Toolchain installation complete

Click Finish to complete the AVR32 Toolchain installation process.

3.4 Install AVR32 Studio 2.5

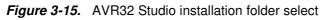
Double-click on the AVR32Studio-2.5.x.exe file to start the installation process.

Figure 3-14. AVR32 Studio 2.5 installer welcome



Click Next.





	dio - InstallShield Wizard Pestination Location		×
	Ider where setup will install files.		/
	Install AVR32 Studio to: C:\Program Files\Atmel\AVR Tools		Change
InstallShield –		K Back Next	Cancel

Check that the installation folder is correct and click Next.

Figure 3-16. AVR32 Studio installer configuration finished

AVR32 Studio - InstallShield Wizard 🛛 🛛 🗙
Ready to Install the Program The wizard is ready to begin installation.
Click Install to begin the installation.
If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.
InstallShield < Back Cancel

Click Install to start the installation.



AVR32 Studio - InstallShield Wizard	
Setup Status	
The InstallShield Wizard is installing AVR32 Stud	io
InstallShield	Cancel

Wait for the installation process to complete.

If a suitable Java[™] runtime is not installed, a Java installer wizard will guide you through the installation procedure.

Figure 3-18. AVR32 Studio installation process complete

AVR32 Studio - InstallShield Wizard		
	InstallShield Wizard Complete The InstallShield Wizard has successfully installed AVR32 Studio. Click Finish to exit the wizard.	
-	< Back Finish Cancel	

Tick Create shortcut on desktop if you want a shortcut to be created. Then click Finish.



3.5 Connect the AVR ONE! to power and USB host

- Connect the AVR ONE! to power using the supplied power supply.
- Connect the AVR ONE! to the USB host (PC) using the supplied USB cable
- Turn on the AVR ONE! using the power switch next to the power connector

Figure 3-19. AVR ONE! connected to power and USB





3.6 Install AVR ONE! Driver

When the AVR ONE! is powered up and connected to the PC for the first time, the proper USB driver must be installed. Since the PC is keeping track of the serial number of each USB device, this will happen every time a new AVR ONE! is connected to the PC, even if the driver is the same as for all other AVR ONE!s that have been connected previously. This is a property of the operating system, and is not controlled by any Atmel software installed.

Figure 3-20. "New hardware" notification pop-up

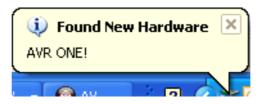


Figure 3-21. AVR ONE! Hardware installation wizard

Found New Hardware Wizard		
	Welcome to the Found New Hardware Wizard Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission). <u>Read our privacy policy</u>	
Can Windows connect to Windows Update to search for software? Yes, this time only Yes, now and every time I connect a device No, not this time Click Next to continue.		
	< Back Next > Cancel	

When the hardware installation wizard pops up, select No, not this time and click Next.



Figure 3-22. Hardware installation wizard configuration



Select Install the software automatically and click Next.

Figure 3-23. Hardware installation in progress

Found New I	Hardware Wizard
Please wa	it while the wizard searches
E	AVR ONE!
	<u>S</u>
	K Back Next > Cancel

Wait for the installation process to complete.







Click Finish.

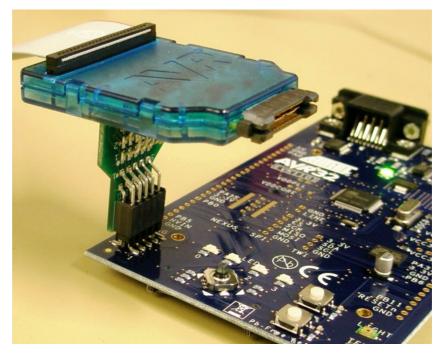




4.1 Connect the AVR ONE! to the EVK1101

Connect the AVR ONE! debugger to the EVK1101 evaluation board using the 10 pin JTAG connector. To make it possible to use the joystick while the AVR ONE! is connected to the JTAG connector, the 100mil - 100mil JTAG stand-off adapted can be used.

Figure 4-1. AVR ONE! connected to the EVK1101



4.2 Connect the EVK1101 to power and RS232

Connect the EVK1101 to power and turn it on. The easiest way to provide power is to use the supplied USB cable. Also connect the RS232 port to your PC using the supplied RS232 cable.

Switch it on by setting the power switch to **VBUS**.

Figure 4-2. Powering the EVK1101 using the USB cable



Note: If the EVK1101 contains the Control Panel Demo Application, you may be requested to install drivers for it. Just cancel this request (you do not need to install this driver).





Section 5

Create demo application

5.1 Start AVR32 Studio

Start AVR32 Studio. Start-up may take a while (because of all the Java libraries being loaded).

Figure 5-1. AVR32 Studio splash screen



Figure 5-2. AVR32 Studio workspace selection

Workspace Launcher	×
Select a workspace	
AVR32 Studio stores your projects in a folder called a workspace. Choose a workspace folder to use for this session.	
Workspace: C:\AVR32_Demo Brov	vse
Use this as the default and do not ask again	
OK Ca	ncel

Select a suitable workspace folder for your project files. If you want to use the same folder for your workspace every time you start AVR32 Studio, you should tick the box before clicking **OK**.

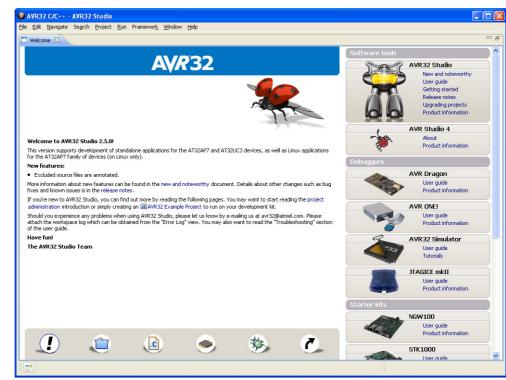


Figure 5-3. AVR32 Studio Welcome view

Exit from the welcome screen to the workbench by clicking on the **Close Page** icon (Arrow).

5.2 Configure adapter and target

Before you can use the AVR ONE! and the EVK1101, you have to tell AVR32 Studio what type of equipment is connected to your PC.

"Target" refers to the MCU on the EVK1101 evaluation board, and "Adapter" refers to the tool connecting the target to the PC (in this case, the AVR ONE!).



5.2.1 Add and configure the adapter (AVR ONE!)

Figure 5-4. Scan Targets

📣 📣	VR32 Targets 🛛	3		1	- 8
	Name 🔻	Adapter	Board		1
		Scan Tarne	ets		
<					>

Right-click in the AVR32 Target-view and select Scan Targets.

Figure 5-5. Available targets

🥔 A'	VR32 Targets 🛛		a 🖉 🚽	
	Name 💌	Adapter	Board	1 P
	AVR ONE!	AVR ONE!		
۲	AVR32 Simulator	AVR32 Simulator	AVR32 Simulator	
	R			
<				>

Select the AVR ONE!

Figure 5-6. Selecting the properties view

L								
🖹 Problems 🔲	🖹 Problems 🔲 Properties 🛛 📮 Console							
AVR ONE	!							
General	Name:	AVR ONE!						
Details								
Decails	▼ Binaries							
Daisy Chain								
	AVR32 Studio keep	os track of the last file used to program a target. The name and date is show below.						
Information								
	Binary path:							
	,							
	Binary date:	Thu Jan 01 01:00:00 CET 1970						

Click on the **Properties** tab.

You are now looking at the *Target* properties. If you have several adapters connected at the same time, this is the place where you can give them unique names. Just type the name you want to use in the **Name** field.



5.2.2 Configure target board and MCU



🖹 Problems 🔲 Properties 🛛 📮 Console								
AVR ONE!								
General	Details							
Details	Debugger/programm	er: AVR ONE! Device: UC3A0512	Select					
Daisy Chain Information	Clock source:	Internal RC oscillator 🕑 Board: 🕱 🕅	~					
	▼ Connection							
	Serial number:	0000000015						
	Connection:	usb	*					
	COM Port:							
	▼ Clock		?					
	JTAG Clock	····	4MHz					
		🖥						
	aWire Clock		256kHz					
	Read Apply d	Value is out of recommended range, please check data sheet!						

Set Device to UC3B0256 or UC3B0256ES, depending on what MCU is installed on your EVK1101.

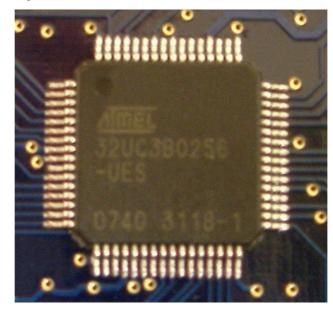


Figure 5-8. MCU Markings

To check which type of MCU is mounted on your EVK1101 evaluation board, you can read the part number printed on the MCU. The picture shows the part number printed on an -ES part (-UES suffix).

Set Board to EVK1101.

Set **MCU Clock source** to **Crystal** and adjust the JTAG Clock to a suitable value (Usually 33MHz or less. Max speed depends on target board signal quality). Click **Apply**.

The target and adapter configuration process is now complete.



5.2.3 Target MCU Chip erase

If the EVK1101 evaluation board is brand new, or if it still contains the original demo application (Control Panel Demo), the FLASH lock-bits need to be cleared. Right-click on the AVR ONE! In the *AVR32 Target* view and select **Chip Erase**.

WARNING! This process will erase the original demo application programmed at the factory. After this operation the EVK1101 evaluation board will be completely empty. If you need to keep the original application, you should not perform this operation.

If you would like to use your EVK1101 for this example, it is not difficult to restore the original "Control Panel Demo application". All you have to do is to build the "Control Panel Demo example" enclosed with AVR32 Studio.

You should now perform the Chip Erase operation.

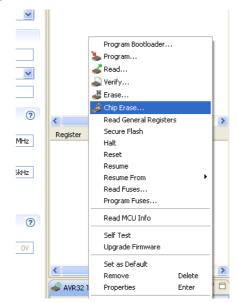


Figure 5-9. Chip erase operation

Right click on the target (AVR ONE!), and select Chip Erase.



5.3 Create a demonstration project

Figure 5-10. Create new project

_			J								
0	AVR3	2 C/C++ -	AVR32	Studio							
File	Edit	Navigate	Search	Project	Run	Framework	Window	Help			
	New		i	Alt+Shift+	N 🕨	📸 AVR32 C	Project			- Q	i 🔊 🕫
	Open I	=ile				📸 AVR32 0	:++ Projec	t			
	Close			⊂trl+W		📸 AVR32 C	Project Fr	om Templa	ate		
	Close /	41	(+W	😭 AVR32 E	xample Pro	oject			
						📑 Project.					
	Save			Ctrl+S		Source F	older				
	Save A					Folder	01061				
R	Save A		(Ctrl+Shift-	+S	Source F	ilo				
	Revert					Header I					
	Move.						rile				
	Renam	ne	I	-2		File					
8	Refres	h	I	-5		警 Fuse Sel	ttings				
	Conve	rt Line Delin	niters To		•	📸 Other			Ctrl+N		
<u>ن</u>	Print		(Ctrl+P						_	
=	Cushala				•	oblems 🔲	Properties	1 23	📃 Console		
	Restar	Workspace +				AVR ONE!					
—	Restar					- UNL:					
è	Import					aral	Details				

Create a new project by clicking File>New>AVR32 Example Project.

Figure 5-11. Select project example

🙆 New Ex	camp le	
Select a r Creates a		
Wizards:		
type filter	text	
	EVK1100 - SERVICES - USB Mass Storage EVK1100 - SERVICES - USB Mass Storage with FreeRtos example C3BES Software Framework EVK1101 - APPLICATIONS - Control Panel Demo example EVK1101 - COMPONENTS - Accelerometer example EVK1101 - COMPONENTS - Data Flash Memory example EVK1101 - COMPONENTS - Data Flash Memory example EVK1101 - COMPONENTS - SD/MMC Card example EVK1101 - DRIVERS - Analog-to-Digital Converter (ADC) example EVK1101 - DRIVERS - CPU Cycle counter example EVK1101 - DRIVERS - CPU Cycle counter example EVK1101 - DRIVERS - External Interrupt Controller (EIC) example 1 EVK1101 - DRIVERS - External Interrupt Controller (EIC) example 3 EVK1101 - DRIVERS - Flash controller (FLASHC) example EVK1101 - DRIVERS - Flash controller (FLASHC) example	
0	< Back Next > Finish	Cancel

Select EVK1101 - Components - Accelerometer example, then click Next



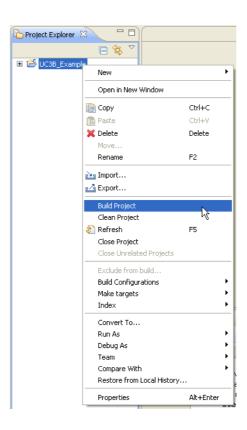
Figure 5-12. New project name

🖲 New Example Project								
EVK1101 - COMPONENTS - Accelerometer example Creates a Accelerometer example for the EVK1101.								
Project name: UC38_Example Ţ								
Location: C:/AVR32_Demo/UC3B_Example Choose file system: default	Browse							
 	Cancel							

Enter a name for the project, and click **Finish**.



Figure 5-13. Build project



Right-click on the project in **Project Explorer**-view and select **Build Project** (or press CTRL+B).

Figure 5-14. Project build progress

Build Project	
Building project	
Run	in Background Cancel Details >>

Wait for the project build process to finish.



Figure 5-15. Console view

🖹 Problems 🔲 Properties 📮 Console 🛛 🦉 Progress	🔒 🚉 📑 🖬 🖬 📬	•
C-Build [UC3B_Example]		
STC\SOFIWARE_FRAMEWORK\DRIVERS\FM\pm.o STC\SOFIWARE_FRAMEWORK\DRIV	ERS(INIC/INCC.)	~
src\SOFTWARE_FRAMEWORK\DRIVERS\INTC\exception.o		_
src\SOFTWARE_FRAMEWORK\DRIVERS\GPIO\gpio.o src\SOFTWARE_FRAMEWORK\	DRIVERS\ADC\adc.	0
src\SOFTWARE_FRAMEWORK\COMPONENTS\ACCELEROMETER\LIS3LO6AL\1is3106a	al.o	
src\SOFTWARE FRAMEWORK\BOARDS\EVK1101\led.o src\SOFTWARE FRAMEWORF	<pre>X\ASM\trampoline.</pre>	0
-1m I		
Build complete for project UC3B_Example		
Time consumed: 14421 ms.		_
		~

The console shows output from the compiler. Make sure that this ends with a "Build complete ..." message (Except for the "Time consumed" message). If something is not working, you will see error messages in this view.

5.4 Configure AVR32 Studio for a debug session using trace

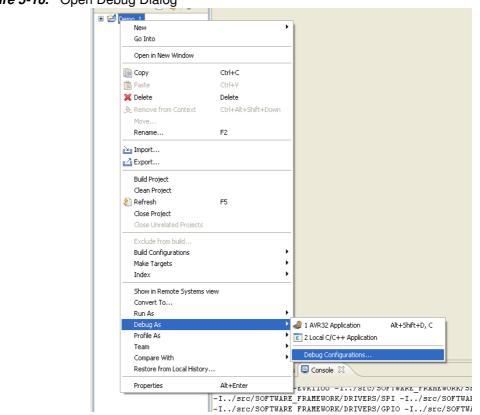


Figure 5-16. Open Debug Dialog

When the build process is finished, right-click on the project in the *Project Explorer* view and select *Debug As>Debug Configurations.*



5.4.1 Create a new debug launch configuration

In the *Debug Configurations* view, select **AVR32 Application** and right click and select **New**. A new launch configuration will be created and default values will be filled into all applicable fields.

Select the *Debugger* tab and tick the **Stop on startup at: main** option.



Debug Configurations		
Create, manage, and run cor	nfgurations 🥤	ġ.
Image: Second	Name: Debugger Trace Common GDB Proxy: Start GDB proxy GDB proxy: Start GDB proxy GDB proxy toommad: awr32gdsproxy GDB proxy trace port: 4711 GDB proxy trace port: 4712 V Stop on startup st: main Verbose mode Verbose mode	
Filter matched 8 of 8 items	Apply_ Rey	ert
?		Close



5.4.2 Configure the target trace module for program trace

Figure 5-18. Debug configurations, Main tab

Debug Configurations Create, manage, and run con	figurations		×
AVR32 Application AVR32 Application G(C++ Application C(C++ Application C(C++ Atpact to Application C(C++ Remote Application C(C+++ Remote Application C(C+++ Remote Application C(C+++ Remote Application C(C+++ Remote Application C(C++++ Remote Application C(C++++++ Remote Application C(C++++++++++++++++++++++++++++++++	Name: UC38_Example Main Debugger [Jace] Project: UC38_Example Ele: Debug/UC38_Example.elf Tgrget: AVR ONEI Launch provider: AVR32 Program/GD8-Proxy Erase sectors O TAG chip, erase Run options Prase sectors V Run from reset vector Dr.00000000 V Reset MCU V Start executing Secure code Enable FlashVault after programming Disable debugging in secured area Desable debugging in secured area	Programming options Suppress programming Verify memory Fuse settings Before:	Browse Browse
Filter matched 8 of 9 items		Apply	Re <u>v</u> ert
?		Debug	Close

In the Main tab, make sure that Target is set to AVR ONE!

Figure 5-19.	Debug	configurations,	Debugger tab
--------------	-------	-----------------	--------------

😉 Debug Configurations			×
Create, manage, and run con	figurations		Ť.
Image: Second	Name: UC38_Example Main_Debugger Trac CDB Proxy GDB proxy command: GDB proxy command: GDB proxy trace port: GDB proxy trace port: Stop on startup at: Verbose mode	ace Common t: avr32gdbproxy locahost 4711 t: 4712	
Filter matched 8 of 8 items		Apply	Revert
?		Debug	Close

Select the **Debugger** tab and check the checkbox at the option **Stop on startup at: main**.



Figure 5-20. Enable Trace

Create, manage, and run conf	gurations	1
Yope filter toxt Yope filter toxt ● QR32 Application ● UC38 Example ○ (C++ Attach to Application ○ (C++ Attach to Applica	Name: UC38_Example Main (Debugger Trace Common General: Penable trace Inable ownership trace Trace method: NanoTrace Data trace range 1 Data trace range 2 Memory access type: access Upter boundary: Lower boundary: Upper boundary: Data trace, continue running OBreak, read out and I OBreak, read out and continue Wrap buffer	AUX trace options: Differ size: Override auxiliary port selection Auxiliary port pin configuration: Break on application buffer access Use NANOTRACE vanable © Specify size and location Buffer start: Size: 32 bytes nak
Filter matched 8 of 8 items		Apply Revert

Select the Trace tab and check Enable Trace.

Figure 5-21. Preferred Trace method

Create, manage, and run conf	igurations	Ĩ
Yppe filter taxt Yppe filter taxt US38_Example C(C++ Attach to Application C (C++ Attach to Application C (C++ Remote Application	Name: UC38_Example Main Debugger Trace Common General: Debugger Trace Enable ownership trace Trace method: BisroTrace Debut trace range 1 Data trace range 2 Memory access type: access Uwer boundary: Address: 0x0 Euffer full actions: O Disable trace, continue running O Break, read out and f Break, read out and continue O Wrap buffer	ALX trace options: Differ size: Override auxiliary port selection Auxiliary port pin configuration: Break on application buffer access Use NANOTRACE variable © Specify size and location Buffer start: Size: 32 bytes het
Filter matched 8 of 8 items		Apply Revert

Select the preferred trace method. In this case we want Nano Trace.

Deselect Break on application buffer access.



Figure 5-22. Trace buffer size

Create, manage, and run conf	igurations	- Contraction of the second se
AVR32 Application AVR32 Application COSB_Example C/C++ Application C/C++ Attach to Application C/C++ Remote Application C/C++ Remote Application C GDB Hardware Debugging Launch Group	Name: UC38_Example Main Debugger Trace Common General: Plable trace Enable ownership trace Trace method: NanoTrace Data trace onplons: Data trace range 1 Data trace range 1 Data trace range 2 Memory access type: access Uower boundary: 0x0 Address: 0x0 Buffer full actions: Ox0 O blaske trace, continue running Direak, read out and tontinue	AUX trace options: Buffer size: Override auxiliary port selection Auxiliary port pin configuration: Peeal on application buffer access O Use NANOTRACE variable O Specify size and location Buffer start: Dide AUXOTRACE variable Size: 1024 bytes ak
Filter matched 8 of 8 items		Apply Revert

Select **Specify size and location** option. Then click **Detect** to configure trace buffer size and location.

Figure 5-23. Buffer full action

Create, manage, and run conf	igurations	- To
AVR32 Application AVR32 Application Complexemple Complexemple Complexemple Complexemple Complexemple Complexemple Complexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexemplexe	Name: LUC38_Example Main Debugger Trace Common General:	AUX trace options: Differ size: Override auxiliary port selection Auxiliary port pin configuration: Break on application buffer access Use NANOTRACE variable © Specify size and location Buffer start: Size: 1024 bytes Net
Filter matched 8 of 8 items		Apply Revert

Selected the preferred action when buffer is full. In this case we choose Break, read out and halt.



5.4.3 Configure the target trace module for data trace

We would like to trace all data written to the debug UART. We do a quick lookup in the datasheet and find that the UART registers are located between 0xffff1400 and 0xffff1d00. Although we only use one UART in this application, we configure the data trace range to cover all UARTs.

Figure 5-24. Memory access type

Data trace options:		
Data trace range 1 🛛	ata trace range 2	
Memory access type:	access	~
Lower boundary:	access read	
Address:	write	
Upper boundary:		
Address:		0x0
<u>`</u>		

Set Memory access type to write.

Figure 5-25. Data trace lower boundary

Data trace r	ange 1 Data trace range 2	
Memory acc	ess type: access	*
-Lower bou	ndary:	
Address:	0×ffff1400	
Upper bou Address:	ndary: 0x0	

Set lower boundary to 0xffff1400.

Figure 5-26. Data trace upper boundary

Data trace options:	
Data trace range 1 D	ata trace range 2
Memory access type:	access
CLower boundary:	
Address:	0xffff1400
Upper boundary:	
Address:	0xffff1d00

Set upper boundary to 0xffff1d00.



Figure 5-27. Configured trace

C (C++ Aplatadul) C (C++ Acth to Appletation C (C++ Postmotram Debug) C (C++ Rende Aplatadul) C (C++	Create, manage, and run conf	igurations	
Apply Revert		Main Debugger Trace Common General: Common Usable trace Enable ownership trace Trace method: NanoTrace Data trace options: Data trace range 1 Data trace range 1 Data trace range 2 Memory access type: access Lower boundary: 0xfff11400 Address: 0xfff11400 Buffer ful actions: Otheshe trace, read out and Obside trace; continue running	Buffer stze: 1288 Override auxiliary port selection Auxiliary port pin configuration: Oreak on application buffer access Oreak NANOTRACE variable Operatory size and location Buffer start: 0x5c00 Detect Size: 1024 bytes
Filter Induited o Ur o items	Filter matched 8 of 8 items		Apply Revert



5.5 Start a debug session and configure the debugger for trace

Click the **Debug** button in the *Debug Confugurations* view. Now the program will be loaded into the target, and run until main().

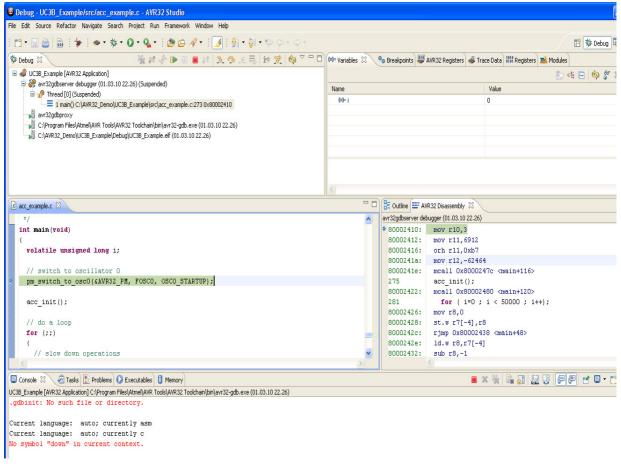
Figure 5-28. Switching perspective

🙆 Con	firm Perspective Switch					
2	This kind of launch is configured to open the Debug perspective when it suspends.					
	This Debug perspective is designed to support application debugging. It incorporates views for displaying the debug stack, variables and breakpoint management.					
	Do you want to open this perspective now?					
Rem	ember my decision Yes No					

When the debug session starts, AVR32 Studio 2.5 will change to the *Debug* perspective (desktop layout designed for use during debug sessions). You should click **Yes**. To avoid being asked every time you start a debug session, you should also click the **Remember my decision** box before answering **Yes**.

Wait until the target has stopped at the first instruction in the main() routine.

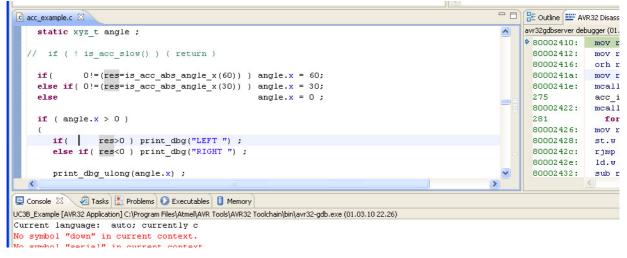
Figure 5-29. Program halted at main()





5.6 Add start and stop trace-points

Figure 5-30. Source code editor



Scroll up to line 156 in the file acc_example.c and right-click at the left edge of the editor. Select **Add Tracepoint...** from the pop-up menu.



Figure 5-31. Tracepoint (Start)

Properties for		
····· Tracepoint	Line Tracepoint Prop	perties 🗘 🗘 - 💌
	Trigger event: Program counter Data read Data write Data access	Trigger location: location Range inclusive Trigger value compare: No value comparison Mask bytes: Value equals: 0 Disregard size
	Tracepoint type: Program trace Data trace Boundaries Trigger start location: -	Trace operation: Emit trace message Start trace Stop trace
	File location:	C:\AVR32_Demo\UC3B_Example\src\acc_example.c,156
?		OK Cancel

Set Tracepoint Configuration values:

- Set Trigger Event to Program Counter
- Set Trace Operation to Start Trace
- Set Tracepoint type to both Program trace and Data trace
- Click OK

This will create a tracepoint that starts both program and data trace when the program counter hits this code line.

Scroll down to line 160 in the file acc_example.c and right-click at the left edge of the editor. Select **Add Tracepoint...** from the pop-up menu.



Figure 5-32. Tracepoint (Stop)

Properties for	
····· Tracepoint	Line Tracepoint Properties $\diamond \bullet \bullet \bullet \bullet \bullet \bullet$
	✓ Tracepoint is enabled Trigger event: Trigger location: O Program counter Location O Data read Range inclusive O Data write Trigger value compare: O Data access Mask bytes: Value equals: Disregard size
	Tracepoint type: Trace operation: ✓ Program trace ○ Emit trace message ✓ Data trace ○ Start trace Boundaries ○ Stop trace Trigger start location: □
	File location: C:\AVR32_Demo\UC3B_Example\src\acc_example.c,160 Trigger end location: File location:
	Restore Defaults Apply
?	OK Cancel

Set Tracepoint Configuration values:

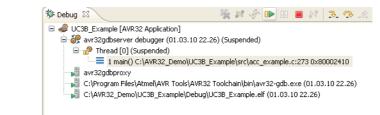
- Set Trigger Event to Program Counter
- Set Trace Operation to Stop Trace
- Set Tracepoint type to both Program trace and Data trace
- Click OK

This will create a tracepoint that stops both program and data trace when the program counter hits this code line.



5.7 Start the trace debug session

Figure 5-33. Resume debug session



Make sure that the main() process is still selected in the *Debug* view before pressing the **Resume** button.

Start a serial port terminal to view the output from the debug UART. To make it simple, we just start Hyperterminal. Click on *Start>All Programs>Accessories>Communications>Hyperterminal*.

Figure 5-34. New Hyperterminal

New Connection - HyperTerminal		
File Edit View Call Transfer Help		
다 🗃 🍯 🖉 💲 😭		
	Connection Description Image: Connection Enter a name and choose an icon for the connection: Name: EVK1101_Demo_1 Icon: Icon: Image: Connection Icon: Image: Connection	
Disconnected Auto detect A	uto detect SCROLL CAPS NUM Capture Print echo	

Enter a name for the session and click **OK**.



Figure 5-35. Hyperterminal port selection

Connect To	? 🛛
	1_Demo_1
Enter details for	the phone number that you want to dial:
Country/region:	Norway (47)
Area code:	
Phone number:	
Connect using:	СОМ1
	OK Cancel

Seletc the com-port that you connected the EVK1101 to (in this case we use **Com1**).

Figure 5-36. Hyperterminal port configuration

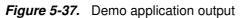
COM1 Properties	? 🛛
Port Settings	
Bits per second:	57600
Data bits:	8
Parity:	None
Stop bits:	1
Flow control:	None
	Restore Defaults
	K Cancel Apply

Set port parameters:

- Bits per second: 57600
- Data bits: 8
- Parity: None
- Stop bits: 1
- Flow control: None

Click OK.





🔐 EVK1101_Demo_1 - HyperTerminal	
File Edit View Call Transfer Help □ 🖙 📾 🕉 💷 🎦 🔛	
Accelerometer Example	
HEX Value for Channel x : 0x00000079 = 0x00000080 HEX Value for Channel y : 0x0000007A = 0x00000080 HEX Value for Channel z : 0x00000049	
HEX Value for Channel z : 0x00000049	
-	
<u></u>	
Connected 0:00:20 Auto detect 57600 8-N-1 SCROLL CAPS NUM Capture Print echo	

Tilt the EVK1101 board carefully as shown in the photograph. Start with the board laying flat on the table, and increase the tilt slowly.

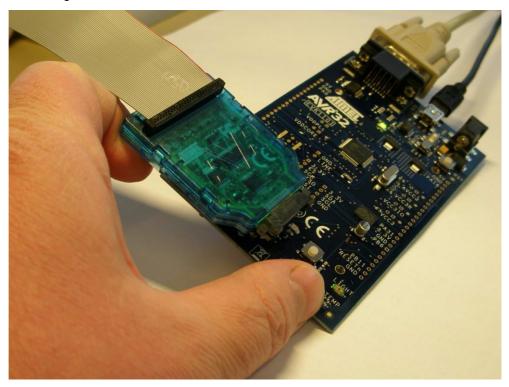


Figure 5-38. Tilting the EVK1101 board



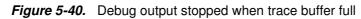
When the tilting angle reaches a certain value, the debug output notifies you about which direction the board is being tilted.

Figure 5-39. Tilt direction indicator

Image: Second state state Image: Second	
Accelerometer Example HEX Value for Channel x : 0x0000006D > 0x0000008D right HEX Value for Channel y : 0x00000078 = 0x00000080 HEX Value for Channel z : 0x0000004A -	
Connected 0:00:51 Auto detect 57600 8-N-1 SCROLL CAPS NUM Capture Print echo	

When the tilt angle reaches 30 degrees, the program wants to print an additional message saying **RIGHT30**. When this happens, the program counter hits the start tracepoint, and trace data will start being collected.





🔬 EV	/K1101_Demo_	1 - HyperTerm	iinal							
	Edit View Call									
	¥ 🐲 🕉 🗈	₽ ₽								
Ac	celeromet	er Examp]	le							
HE	X Value f X Value f X Value f	or Channe	el v : 0x0	1000007	9 = (0×000 0×000	0008D 00080	rig	nt	
R	[GH_									
Conne	cted 0:01:09	Auto detect	57600 8-N-1	SCROLL	CAPS	NUM	Capture	Print echo		·····

When the trace buffer in the target MCU is full, the program will break, and trace data will be uploaded to AVR32 Studio for inspection.

5.8 View trace data

Figure 5-41. Trace data view

(×)= \	ariables 🔍 ariables	🕮 AVR32 Registers	s 🚙 Trace Data 🛛 👬 Registers	A Modules	F٩	20 0	á 🧬	A.	• - E
	_Example								
	Frame #	Address	Frame description						
	0x00000000		Watchpoint hit						
	0x00000001	0x800023f8	Trace started						
1	0x00000002	0x800023f8	print_angles()						
2	0x00000003	0x8000240a	print_angles()						
2	0x00000004	0x80002684	print_dbg()						
8	0x00000005	0x800026f0	print()						
1	0x00000006	0x80002c0c	usart_write_line()						
1	0x00000007	0x80002c3a	usart_write_line()						
8	0x0000008	0x80002c1e	usart_write_line()						
1	0x00000009	0x80002b78	usart_putchar()						
1	0x0000000a	0x80002bca	usart_putchar()						
2	d0000000k	0x80002be4	usart_putchar()	~					
344 t	ace frames								>

Click on the Trace Data tab to view the trace frames.



Figure 5-42.	Reconstructed source code
--------------	---------------------------

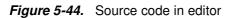
(×)= V	ariables 🤷 Breakpoints	🕮 AVR32 Registers	🚽 Trace Data 🛛 👫 🖓	🛋 Modules	;]	1	æ 💰	🥔 🖉 🚚 🖶 - 🗆 🗖
UC3B	_Example							
	Frame #	Address	Frame description	🚺 🔼 Dire	et call	. into src/SOFTWARE_FRAM	IEWORK	/DRIVERS/USART/usa
2	0x0000008	0x80002c1e	usart_write_line()	- }	while	(usart_write_char(usar	t, c)	!= USART_SUCCESS);
1	0x00000009	0x80002b78	usart_putchar()					
1	0x0000000a	0x80002bca	usart_putchar()					
1	d000000x0	0x80002be4	usart putchar()					
1	0x0000000c	0x80002b28	usart_write_char() 🔓					
	0x0000000d	Oxffff181c	st.w 0x0000052					
Ð	0x0000000e	0x80002b6c	usart_write_char()					
Ð	0x0000000f	0x80002bf0	usart_putchar()					
1	0x00000010	0x80002c3a	usart_write_line()					
1	0x00000011	0x80002c1e	usart_write_line()					
1	0x00000012	0x80002b78	usart_putchar()					
1	0x00000013	0x80002bca	usart_putchar()	~				~
344 ti	ace frames			<				>

Click on a trace frame to view the reconstructed code that was executed by the MCU.

(×)= V.	ariables 🔍 🔍 Breakpoints	I AVR32 Register	rs 📣 Trace Data 🙁 📲 Registers	A Modules	🛃 🎦 💰 🥔	∠ .₽ = □
UC3B	_Example					
	Frame #	Address	Frame description			~
2	0x0000008	0x80002c1e	usart_write_line()	-		
1	0x00000009	0x80002b78	usart_putchar()			
1	0x0000000a	0x80002bca	usart_putchar()			
2	d0000000x0	0x80002be4	usart_putchar()			
1	0x0000000c	0x80002b28	usart_write_char()			
	0x0000000d	Oxffff181c	st.w 0x00000052			
Ð	0x0000000e	0x80002b6c	usart_write_char() 🗟			
	0x0000000f	0x80002bf0	usart_putchar()			
1	0x00000010	0x80002c3a	usart_write_line()			
2	0x00000011	0x80002c1e	usart_write_line()			
1	0x00000012	0x80002b78	usart_putchar()			
1	0x00000013	0x80002bca	usart_putchar()	~		
344 tr	race frames			<u><</u>		>

Figure 5-43. Data trace frame

A data trace frame showing a byte being written to the debug UART transmit register. By enabling data trace only, we can see all characters being sent to the terminal.



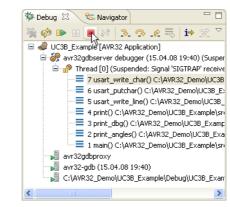
ebug - UC3B_Example/src/SOFTWARE_FRAMEW	ORK/DRIVERS/USART/usar	t.c - AVR32 Studi	o			
Edit Refactor Navigate Search Run Project Frame	ework Window Help					
3 • 🗟 🖆 🗟 🛛 🗢 • 🏇 • 🔕 • 🖓 • 🛛 🔗						😰 🕸 Debug 🏣 AVR32 C/C++
Debug 🛛 😤 Navigator 📃 🗆	🗱 Variables 🤏 Breakpoints	AVR32 Registers	📣 Trace Data 🕺 👌	👫 Registers 🛋	Modules	🛃 🎦 💰 🥔 🖉 💷 📑 • 🧮
🤣 🗈 🗉 🔳 😹 🔍 👁 . e 🔜 🖬 🕱 💙	UC3B_Example					
UC38_Example [AVR32 Application]	Frame #	Address	Frame description			/SOFTWARE_FRAMEWORK/DRIVERS/US
🖶 🔗 avr32gdbserver debugger (15.04.08 19:40) (Susper	🛓 0x0000008	0x80002c1e	usart_write_line	e() 🦰	}	
😑 🝿 Thread [0] (Suspended: Signal 'SIGTRAP' receive	¥ 0x0000009		usart_putchar()			
7 usart_write_char() C:\AVR32_Demo\UC3B	🖡 0x000000a	0x80002bca	usart_putchar()		int yeart init ra??? ty a	nly(volatile avr32 usart t *us
6 usart_putchar() C:\AVR32_Demo\UC3B_E;			usart_putchar()		(miy(volacile avroz_asarc_c as
5 usart_write_line() C:\AVR32_Demo\UC3B_			usart_write_char	: ()	// Reset the USART and	shutdown TX and RX.
4 print() C:\AVR32_Demo\UC3B_Example\sr			st.w Ox000000052		usart reset(usart);	
3 print_dbg() C:\AVR32_Demo\UC3B_Examp			usart_write_char	:()	_	
2 print_angles() C:\AVR32_Demo\UC3B_Exa 1 main() C:\AVR32_Demo\UC3B_Example\sr			usart_putchar()		// Check input values.	
avr32gdbproxy			usart_write_line		if (!opt // Null poi	nter.
avr32-gdb (15.04.08 19:40)	Ox00000011 0x00000012		usart_write_line	e()		<pre> opt->charlength > 9 </pre>
C:\AVR32_Demo\UC38_Example\Debug\UC38_Exam			usart_putchar()		opt->paritytype > 7	
	* 0X0000013	0x80002bca	usart_putchar()	~	opt->stopbits == 1	opt->stopbits > 2 + 255
	344 trace frames				<	
acc_example.c 🕞 usart.h 🔂 usart.c 🖂				🗉 Outline 🖬 Disi	assembly 🖾	=
<pre>int usart_init_rs232_tx_only(volatile (// Reset the USART and shutdown TX usart_reset(usart); // Check input values. if (lopt // Null pointer. opt->charlength < 5 opt->cha opt->charlength < 5 opt->cha opt->stopbits == 1 opt->stop opt->channelmode > 3 usart_set_sync master_baudrate(return USART_INVALID_INPUT; if (opt->charlength == 9) (// Character length set to 9 bits usart->mr = AVR32_USART_MR_MODE9) else</pre>	and RX. wrlength > 9 wbits > 2 + 255 wusart, opt->baudrat	=, pba_hz) ==	rt_options	0x80002b46 0x80002b4c 0x80002b4c 0x80002b4c 0x80002b54 0x80002b55 0x80002b5c 0x80002b5c 0x80002b60 0x80002b64 0x80002b66 0x80002b66 0x80002b60 0x80002b72	<pre><usart_write_char+28>: 1d., <usart_write_char+34>: srm <usart_write_char+34>: srm <usart_write_char+34>: srm <usart_write_char+34>: bre <usart_write_char+34>: bre <usart_write_char+34>: 1d., <usart_write_char+48>: 1d., <usart_write_char+48>: stm, <usart_write_char+54>: stm, <usart_write_char+54>: stm, <usart_write_char+56>: stm, <usart_write_char+56>: stm, <usart_write_char+64>: stm, <usart_write_char+64>: stm, <usart_write_char+64>: stm, <usart_write_char+64>: stm, <usart_write_char+64>: stm, <usart_write_char+64>: stm, <usart_write_char+74>: stm, <usart_write_char+76>: 1dm</usart_write_char+76></usart_write_char+74></usart_write_char+64></usart_write_char+64></usart_write_char+64></usart_write_char+64></usart_write_char+64></usart_write_char+64></usart_write_char+56></usart_write_char+56></usart_write_char+54></usart_write_char+54></usart_write_char+48></usart_write_char+48></usart_write_char+34></usart_write_char+34></usart_write_char+34></usart_write_char+34></usart_write_char+34></usart_write_char+34></usart_write_char+28></pre>	<pre>1 r8,0x2,COH 2 r8 w r8,0 4 0x80002b66 <usart_write_cl 0x80002b6c="" <usart_write_cl="" ktu="" p="" pre="" r12,r8="" r3[xcl],r9="" r6,r7[-16]="" r7[-16],r8="" r8,2="" r8,r7[-12]="" r8[xc],r9="" r8[xr(c],r9="" r9,r8,0x0,0x9="" sp,-16<="" v="" w=""></usart_write_cl></pre>
Console 🛛 🧔 Tasks 🛃 Problems 🖉 Progress 🚺	Memory				🔳 🗶 🖗 🗈 🛛	1 🔜 U 🖻 🖉 🖻 • 📬 • 🗂
B_Example [AVR32 Application] avr32-gdb (15.04.08 19:40)						
rning: Remote failure reply: EO1 cmd_disassemble: Invalid filename.						
						2
*						

Double-click on a trace frame to show the source code in the editor.



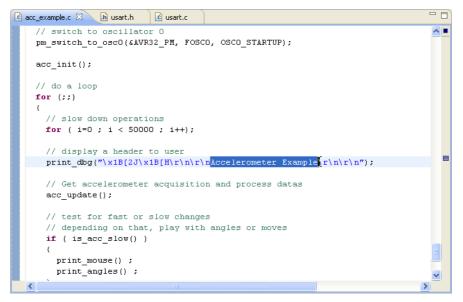
5.9 Modify the application

Figure 5-45. Terminate the debug session



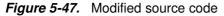
Click the Stop icon to terminate the debug session.

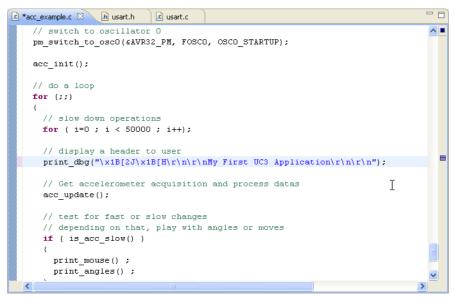
Figure 5-46. Go to source code



Edit the string.







When the source code has been edited, simply restart the previous debug session.

Figure 5-48. Restart debug session

Debug - UC3B_Example/src/acc_example.	c - AVR
File Edit Refactor Navigate Search Run Projec	t Frame
i 🗈 • 🔒 🗁 i 🔷 • 🎠 • 🔍 • 🏊	• : 🔗
🏇 Debug 🖾 😤 Navigator	- 0
🔆 🤣 🕪 🗉 🔳 🕅 🔍 🐟 👘 🖬	T T
🖃 🥔 UC3B_Example [AVR32 Application]	
😑 🎯 avr32gdbserver debugger (15.04.08 19:40)	(Susper
😑 🛷 Thread [0] (Suspended: Signal 'SIGTRAF	^o receive
7 usart_write_char() C:\AVR32_Der	no\UC3B

Click on the **Debug** icon to start a new debug session using the same launch configuration as the previous session.

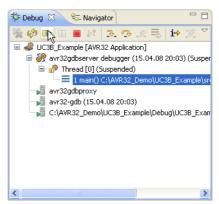
Figure 5-49. Save modifide source code

🙆 Save Resource
'acc_example.c' has been modified. Save changes?
Yes No Cancel

If you did not save the modifide source code, you will be notified now (click **Yes**). After the source code has been saved, AVR32 Studio will re-compile the application and program the target MCU before starting the debug session. The code will run break at main() again.

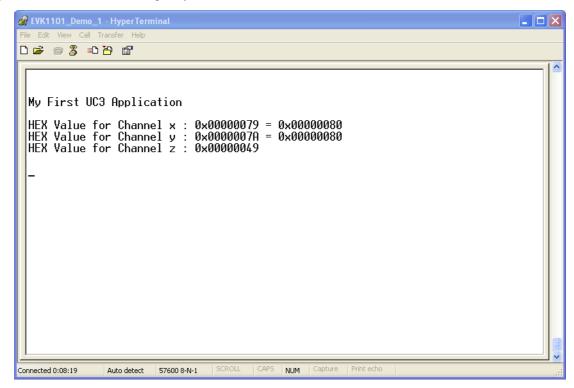


Figure 5-50. Resume debug session



Click on the **Resume** icon to resume the debug session.

Figure 5-51. Modified debug output



Observe the modified output containing your own text.

Congratulations! You have now created your first AVR32 application and collected real time trace data from the target MCU running your program using the AVR ONE!





Section 6

Firmware Upgrade

6.1 Firmware upgrade overview

The tools (adapters) used to provide the physical connection between PC and target MCU contains firmware. This firmware needs to be compatible with the gnu toolchain and AVR32 Studio installed on the PC.

When AVR32 Studio is started, or when a new adapter is detected, AVR32 Studio will perform a firmware version check to determine if the adapter firmware needs to be upgraded.

If AVR32 Studio contains a newer firmware than present in the adapter, the adapter will be upgraded.

6.2 Firmware version test and upgrade

When AVR32 Studio is testing the firmware version of connected adapters, you can see a progress indicator in the status line.

Figure 6-1. Firmware version test

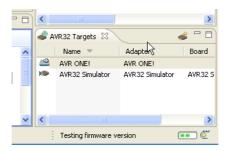
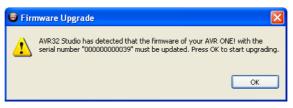


Figure 6-2. Firmware upgrade message



If the adapter firmware must be upgraded, you will be notified by a pop-up. Click **OK** to continue.

Firmware upgrade progress can be monitored by activating the Progress view.

Figure 6-3. Firmware upgrade progress

😰 Problems 🔲 Properties 📮 Console 🙋 Progress 🕱	*	$\overline{\nabla}$	- 6	3
Automatic Update Search				
) 🔳	
Searching: Contacting Target Management Updates				
Upgrading firmware				
			_	

A firmware upgrade report can be found in the *Console* view.

Figure 6-4. Firmware upgrade report

Problems 🔲 Properties	🖳 Console 🛛	🧵 Progress		2	<u></u> .	1	- 0
AVR32 Console							
Upgrading AVR ONE!	FPGA image '	avr32', ple	ease wait				^
Firmware Image	On disk	On tool	Status				
avrone.bin	1.1	0.20	UpgradeRequired				
avr32.bin	1.1	0.e	UpgradeRequired				
							~
<							>

6.3 Adapter in use

The firmware version test is a process that is running in the background. This may cause a situation where the adapter is busy (debug session active) when AVR32 Studio determines that the firmware should be upgraded. In this case, the firmware upgrade process will wait until the adapter is not busy anymore (debug session terminated).

Figure 6-5.	Firmware	upgrade	process	waiting	for a	adapter

🖹 Problems 🔲 Properties 📮 Console 🙋 Progress 🕴	×	\bigtriangledown	
ZzzUpgrading firmware (Sleeping)			





Headquarters

Atmel Corporation 2325 Orchard Parkway San Jose, CA 95131 USA Tel: 1(408) 441-0311 Fax: 1(408) 487-2600

International

Atmel Asia Room 1219 Chinachem Golden Plaza 77 Mody Road Tsimshatsui East Kowloon Hong Kong Tel: (852) 2721-9778 Fax: (852) 2722-1369 Atmel Europe Le Krebs 8, Rue Jean-Pierre Timbaud BP 309 78054 Saint-Quentin-en-Yvelines Cedex France Tel: (33) 1-30-60-70-00 Fax: (33) 1-30-60-71-11

Atmel Japan

9F, Tonetsu Shinkawa Bldg. 1-24-8 Shinkawa Chuo-ku, Tokyo 104-0033 Japan Tel: (81) 3-3523-3551 Fax: (81) 3-3523-7581

Product Contact

Web Site www.atmel.com/avrone Technical Support avr32@atmel.com Sales Contact www.atmel.com/contacts

Literature Requests www.atmel.com/literature

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN ATMEL'S TERMS AND CONDI-TIONS OF SALE LOCATED ON ATMEL'S WEB SITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNTIVE, SPECIAL OR INCIDEN-TAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel's products are not intended, or warranted for use as components in applications intended to support or sustain life.

© 2010 Atmel Corporation. All rights reserved. Atmel®, logo and combinations thereof, AVR® and others, are the registered trademarks or trademarks of Atmel Corporation or its subsidiaries. Windows® and others are registered trademarks or trademarks of Microsoft Corporation in the US and/or other countries. Other terms and product names may be trademarks of others.

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Microchip:

ATAVRONE-PSU