

# Getting Started with DS-MDK

Create Applications for Heterogeneous ARM<sup>®</sup> Cortex<sup>®</sup>-A/ Cortex<sup>®</sup>-M Devices

C/C++ - Blinky/Blinky.tteconfig - Eclipse Platform							-		×	
File Edit Source Refactor Navigate	Search Project Run Windo	w Help								
🖆 • 🗷 🕼 🗠   🛞 • 🗞 • 🕼   1	x 🔝 🕜 • 😂 • 🖻 • G	• * •	• 🖸 • 🂁 • 🛛	<b>9</b> 🖉 🗝 🖪 I	日 数 ・ 御	→ ← ← → → Quick According	ess	<b>B</b>	* 💧	<u>s</u>
🍋 Project Explorer 🕴 👘 🗖	🚸 Blinky.rteconfig 🙁 📧 RTX_	Conf_CM.c	c			-		0 🛯 '	1 -	
E 🕏 🔻	Components Resolve					(?)				
🗸 😂 Blinky	•					01		An outline	is not	
> 🔊 Includes	Software Components	Sel. Vi	ariant	Vendor	Version	Description		available.		
> 👝 Debug	MCIMX7D:Cortex-M4			NXP		ARM Cortex-M4, 64 kB RAM, 32 kB ROM				
🗸 🂁 RTE	> Seard Support	м	ICIMX7D-SABRE	Keil	1.0.0	iMX7D SABRE Board	_			
V 🗁 Board_Support				1014		Cortex Microcontroller Software Interface Components	_			
> board.c [Keil.MCIMX7]	CORE			ARM	4.3.0	CMSIS-CORE for Cortex-M, SCOUD, and SCOU	_			
> 🙀 clock_freq.c [Keil.MCII	OSP			ARM	1.4.6	CMSIS-DSP Library for Cortex-M, SC000, and SC300	_			
> is pin_mux.c [Keil.MCIM.	<ul> <li>RIUS (API)</li> </ul>	-		1014	1.0	CMSIS-RIUS API for Cortex-M, SCUUU, and SC300				
> [ retarget_io.c [Keil.MCI	Kell RIA			ANW	4.00.0	Unified Device Driver correliant to CMCIC Driver Section	<u> </u>			
V 🗁 CMSIS	S CMSIS Driver					Unified Device Drivers compliant to CMSIS-Driver Specifications	_			
> C RTX_Conf_CM.c [ARM	> Compiler					ARM Compiler Software Extensions	_			
RTX_CM3.lib [ARM::CF	> Vevice		DK DL	M-3	670	Startup, system setup	_			
> 👝 Compiler	> Prie System	IM N	IDK-Plus	Canada	6.7.0	File Access on various storage devices	_			
> 👝 Device	> Craphics	IVI M	IDK-Plus	, segger	710	Dist National Contract of California Contractor	_			
> in Kit_Components.n	> OnenAMD	IVI	IDK-FIUS	, Kell	1.1.0	In white working using contented of Senar protocols				
> [e] main.c	> OpenAmir	M	IDK Dive	Keil	670	USP Communication with various during classes				
<ul> <li>Diala standin</li> </ul>	7 <b>4</b> 050			, item	0.710		>			
ACIMY7D Center M4 set	-						-			
Hello World	Validation Output			D	escription					
V B Hello World c										
St stdio h										
stdlib.h	<						>			
<ul> <li>main(void) : int</li> </ul>	Components Device Packs									
> 🗠 KPM56 PingPong BM	🕒 Console 🐹 🎜 Tasks 📓 Pi	roblems 1	Properties			R. 6	8   🗖	<b>⊡ - г</b>	• •	
	CMSIS RTE console (RPMSG PingP	ong BM1							-	
	10:06:33 **** Updating pro	iect RPI	MSG PingPong B	M						~
	Loading RTE configuration	1 - C	0 0							
	Updating resources									
	Updating build settings									
	project updated succession	. 19								
										$\sim$
< >	<									>

Information in this document is subject to change without notice and does not represent a commitment on the part of the manufacturer. The software described in this document is furnished under license agreement or nondisclosure agreement and may be used or copied only in accordance with the terms of the agreement. It is against the law to copy the software on any medium except as specifically allowed in the license or nondisclosure agreement. The purchaser may make one copy of the software for backup purposes. No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or information storage and retrieval systems, for any purpose other than for the purchaser's personal use, without written permission.

Copyright © 1997-2016 ARM Germany GmbH All rights reserved.

Keil<sup>®</sup>,  $\mu$ Vision<sup>®</sup>, Cortex<sup>®</sup>, CoreSight<sup>TM</sup> and ULINK<sup>TM</sup> are trademarks or registered trademarks of ARM Germany GmbH and ARM Ltd.

Microsoft<sup>®</sup> and Windows<sup>™</sup> are trademarks or registered trademarks of Microsoft Corporation.

PC<sup>®</sup> is a registered trademark of International Business Machines Corporation.

Eclipse<sup>®</sup> is a registered trademark of the Eclipse Foundation, Inc.

#### NOTE

We assume you are familiar with Microsoft Windows, the hardware, and the instruction set of the Cortex-A and Cortex<sup>®</sup>-M processors.

Every effort was made to ensure accuracy in this manual and to give appropriate credit to persons, companies, and trademarks referenced herein.

### Preface

Thank you for using the DS-MDK Development Studio available from ARM<sup>®</sup>. To provide you with the very best software tools for developing ARM based embedded applications we design our tools to make software engineering easy and productive. ARM also offers therefore complementary products such as the ULINK<sup>TM</sup> debug and trace adapters and a range of evaluation boards. DS-MDK is expandable with various third party tools, starter kits, and debug adapters.

#### **Chapter Overview**

The book starts with the installation of DS-MDK and describes the software components along with complete workflow from starting a project up to debugging on hardware. It contains the following chapters:

**DS-MDK Introduction** provides an overview about the DS-MDK, the Software Packs, and describes the product installation along with the use of example projects.

Eclipse IDE explains the basic concepts of the IDE and the most frequently used perspectives.

**Create Cortex-M4 Applications** guides you through the process of creating and modifying projects using CMSIS and device-related software components for the Cortex-M microcontroller.

**Create Linux Applications** shows you how to create and modify applications for the Cortex-A processor running Linux.

**Debug Applications** describes the process of how to connect to the target hardware and explains debugging applications on the target.

**Store Cortex-M4 Image** gives further details on how to store the application image on the target and how to run it at start up time.

# Contents

Preface	
DS-MDK Introduction	7
Solution for Heterogeneous Systems	7
DS-MDK Licensing	
License Types	8
Installation	9
Software and Hardware Requirements	9
Install DS-MDK	9
Manage Software Packs	
Install the Linux Image	
Hardware Connection	
Verify Installation using Example Projects	
Documentation and Support	17
Eclipse IDE	
Workbench	
Perspectives	
C/C++ Perspective	
CMSIS Pack Manager Perspective	
Remote System Explorer Perspective	
DS-5 Debug Perspective	
Create Cortex-M4 Applications	
Blinky with CMSIS-RTOS RTX	
Setup the Project	
Select Software Components	
Configure CMSIS-RTOS RTX Kernel	
Create the Source Code Files	
Adapt the Scatter File	
Build the Cortex-M4 Image	
Create Linux Applications	
Setup the Project	
Build the Application Image	
Debug Applications	
Preparing the Terminal Views	
Debug Cortex-M4 Application	
Stop in U-Boot	
Configure CMSIS DS-5 Debugger	40
Run Cortex-M4 Application	

Debug Linux Application	42
Setup RSE Connection	43
Boot Linux	43
Configure DS-5 Debugger	44
Run Linux Application	46
Store Cortex-M4 Image	
Create a Cortex-M4 BIN Image File	47
Store Cortex-M4 BIN Image File on SD Card	48
Run Cortex-M4 BIN Image File from U-Boot	49

#### NOTE

*This user's guide describes how to create applications with the Eclipse-based DS-MDK IDE and Debugger for ARM Cortex-A/Cortex-M based NXP i.MX 6 and 7 series.* 

Refer to the **Getting Started with MDK** user's guide for information how to create projects for ARM Cortex-M microcontrollers using the  $\mu$ Vision<sup>®</sup> IDE/Debugger.

# **DS-MDK** Introduction

DS-MDK combines the Eclipse-based DS-5 IDE and Debugger with CMSIS-Pack technology and uses Software Packs to extend device support for devices based on 32-bit ARM Cortex-A processors or heterogeneous systems based on 32-bit ARM Cortex-A and ARM Cortex-M processors.

Initially, only NXP i.MX 6 and 7 series devices are supported that combine computing power for application-rich systems with real-time responsiveness. For such embedded systems, the DS-5 Debugger gives visibility to multi-processor execution and allows optimization of the overall software architecture.

### **Solution for Heterogeneous Systems**



Heterogeneous systems usually consist of a powerful ARM Cortex-A class application processor and a deterministic ARM Cortex-M based microcontroller. These systems combine the best of both worlds: the Cortex-A class processor can run a feature-rich operating system such as Linux and enables the user to program complex applications with sophisticated human-machine interfaces (HMI). The Cortex-M class controller offers low I/O latency, superior power efficiency and a fast system start-up time for embedded systems.

Usually, both processors have access to a set of communication peripherals and shared memory. The biggest challenge with heterogeneous systems is the synchronization and inter-processor communication.

DS-MDK offers a complete software development solution for such systems:

 It allows managing Cortex-A Linux and Cortex-M RTOS projects in the same development environment.

- It fully supports the Cortex Microcontroller Software Interface Standard (CMSIS) development flow for efficient Cortex-M programming. Software Packs may be added any time to DS-MDK making new device support and middleware updates independent from the toolchain. They contain device support, CMSIS libraries, middleware, board support, code templates, and example projects. The IDE manages the provided software components that are available for the application as building blocks.
- The DS-MDK Debugger offers full visibility for multicore software development.

# **DS-MDK Licensing**

DS-MDK is part of the <u>Keil MDK-Professional Edition</u> and the product requires a valid license for MDK-Professional Edition.

# License Types

The following licenses types are available:

**Single-User License (Node-Locked)** grants the right to use the product by one developer on two computers at the same time.

Floating-User License or FlexLM License grants the right to use the product on several computers by a number of developers at the same time.

For further details, refer to the *Licensing User's Guide* at www.keil.com/support/man/docs/license.

### Installation

#### Software and Hardware Requirements

DS-MDK has the following minimum hardware and software requirements:

- A PC running a Microsoft Windows (32-bit or 64-bit) operating system
- Dual-Core Processor with > 2 GHz
- 4 GB RAM and 8 GB hard-disk space
- 1280 x 800 or higher screen resolution

### **Install MDK**

Download **MDK** from <u>www.keil.com/download</u> - Product Downloads and run the installer. It also adds the Software Packs for ARM **CMSIS** and MDK **Middleware**.

Follow the instructions on

www.keil.com/support/man/docs/license/license\_sul\_install.htm to activate a MDK-Professional license, which is required for DS-MDK.

#### Install DS-MDK

Download **DS-MDK** from <u>www.keil.com/mdk5/ds-mdk/install</u> and run the installer. Having finished the installation, start DS-MDK by clicking on **Eclipse** for **DS-MDK** in the Start menu (Windows 10: All apps  $\rightarrow$  ARM DS-MDK  $\rightarrow$  Eclipse for DS-MDK).

When starting the product the first time, you will be presented with a window showing MDK installation detected in your system:

Select MDK-ARM installation				×
Select MDK-ARM installation Select the location of your MDK-ARM installation.				
MDK-ARM installation directory: C:\Keil_v5			Brov	vse
0	Apply & Restart	(	Cancel	

If required, change the installation destination.

Now, you need to specify a directory for your workspace (the area where your projects will be stored). For most users, the default suggested directory is the best option.

Workspace Launcher	×
Select a workspace	
Eclipse Platform stores your projects in a folder called a workspace. Choose a workspace folder to use for this session.	
Workspace: C:\Users\USER\Documents\DS-MDK Workspace V Browse	
Use this as the default and do not ask again	
OK Cancel	

The Eclipse-based IDE opens in the C/C++ Perspective:



#### NOTE

*Refer to chapter* **Eclipse IDE** *on page 18 for more information on Eclipse workbench concepts.* 

### **Manage Software Packs**

Use the **CMSIS Pack Manager** perspective for managing Software Packs on the local computer.

Open this perspective using P Window  $\rightarrow$  Open Perspective  $\rightarrow$  CMSIS Pack Manager. You should now install the Software Pack related to your target device or evaluation board.

#### NOTE

*Currently, only Software Packs for the NXP i.MX 6 and 7 series are qualified for DS-MDK.* 

Devices 2.5 🔮 boards 🕂	= 🕐 💥 🗸 🗖 🗖	🐞 Packs 🛛 🎦 Examples		🕀 🖃 🕐 😂 🔶 🔍 🗖 🖬	💷 Pack Properties 🖂	⊞ ⊟	0	1
earch Device		Search Pack			✓ ⊕ Keil.iMX7D_DFP.1.0.0			
Device	Summary	Pack	Action	Description ^	> Boards			
All Devices	3660 Devices	M Device Specific	32 Packs	NXP selected	> Components			
ABOV	6 Devices	Berrice opecane     Wireless	Se Install	Clarinox Bluetooth Classic Bluetooth Low Eper	> Devices			
Ambia Micro	8 Devices	Keil MX7D DEP	Lin to date	NXP i MX 7Dual Family Device Support and Eva	> Examples			
Analog Devices	16 Devices	Heil Kinetic KOO DEP	Install	Freescale Kinetis KOO Series Device Support and Exa				
	26 Devices	h Keil Kinetic K10 DEP	Constall	Freescale Kinetis K10 Series Device Support				
> Atmal	260 Devices	Hail Kinetis K20 DEP	Constall	Freescale Kinetis K10 Series Device Support				
> Aunei	425 Devices	the Keil Kinetis K20 DED	S Install	Freescale Kinetis K20 Series Device Support and				
Gira Davias	42.5 Devices	B Kell Kinetis K40 DED	la stall	Freescale Kinetis K40 Series Device Support				
> Gigabevice	40 Devices	Bu Keil Kinetis K60 DED	Install	Freescale Kinetis K40 Series Device Support				
> Photek	19 Devices	Be K-it Kinetis_K50_DFP	la stall	Freescale Kinetis K50 Series Device Support				
> Infineon	A Devices	> "# Kell Kinetis_K00_DFP	Install	Freescale Kinetis Koo Series Device Support and				
> Vilaxim	4 Devices	Weinkinetis_K/0_DFP	W Install	Preescale Kinetis K/o Series Device Support and				
> Viedlatek	2 Devices	> THE REIL KINETS K80_DFP	Install	Freescale Kinetis Kau Series Device Support				
> Vincrosemi	0 Devices	* Rell.Kinetis_KEAXX_DFP	Install	Freescale Kinetis KEAXX Series Device Support a				
> MindMotion	2 Devices	> The Keil Kinetis KExx_DFP	Install	Freescale Kinetis Kbx Series Device Support				
> Vordic Semiconductors	8 Devices	> "# Keil.Kinetis_KLxx_DFP	<ul> <li>Install</li> </ul>	Freescale Kinetis KLxx Series Device Support an				
> Vivoton	433 Devices	> "# Keil.Kinetis_KMox_DFP	Install	Freescale Kinetis KMox Series Device Support ar				
> VNXP	527 Devices	> "# Keil.Kinetis_KSxx_DFP	<ul> <li>Install</li> </ul>	Freescale Kinetis KSxx Series Device Support				
> 🖉 Renesas	3 Devices	> H Keil.Kinetis_KVxx_DFP	Install	Freescale Kinetis KVxx Series Device Support				
Silicon Labs	397 Devices	> H Keil.Kinetis_KWPR1516_DFP	🔅 Install	Freescale Kinetis WPR1516 Series Device Suppo				
> SONiX	49 Devices	> H Keil.Kinetis_KWxx_DFP	Install	Freescale Kinetis KWxx Series Device Support				
> STMicroelectronics	846 Devices	> # Keil.Kinetis_SDK_DFP	<ul> <li>Install</li> </ul>	Freescale Kinetis SDK v1.2.0 including MK64FN				
> Prexas Instruments	342 Devices	> Heil.LPC1100_DFP	🔅 Install	NXP LPC1100 Series Device Support				
> 🔗 Toshiba	90 Devices	> H Keil.LPC1200_DFP	📀 Install	NXP LPC1200 Series Device Support				
		> 🖷 Keil.LPC1300_DFP	📀 Install	NXP LPC1300 Series Device Support				
		> Heil.LPC1500_DFP	🔅 Install	NXP LPC1500 Series Device Support and Examp				
		> Heil.LPC1700_DFP	😔 Install	NXP LPC1700 Series Device Support, Drivers an				
		> 🖶 Keil.LPC1800_DFP	📀 Install	NXP LPC1800 Series Device Support, Drivers an				
		> Heil.LPC4000_DFP	📀 Install	NXP LPC4000 Series Device Support				
		S teil.LPC4300 DFP	🐵 Install	NXP LPC4300 Series Device Support. Drivers an *				

The **Console** window shows information about the Internet connection and the installation progress.

TIP: The device database at <u>www.keil.com/dd2</u> lists all available devices and provides download access to the related Software Packs. If the Pack Manager cannot access the Internet, you can manually install Software Packs using the **Import existing packs** icon ≇ or by double-click on \*.PACK files.

### Install the Linux Image

Currently, DS-MDK supports the following development board:

NXP i.MX 7 SABRE development board: MCIMX7SABRE

For this development board, a pre-configured Linux image with DS-MDK specific debug settings is available. Please download the zipped image file here: <a href="http://www.keil.com/mdk5/ds-mdk/imx7reference">www.keil.com/mdk5/ds-mdk/imx7reference</a>

All steps to create a Linux image for the MX7DSABRESD board are explained in the documentation. Please refer to the **Freescale Yocto Project User's Guide** and the **i.MX Linux User's Guide**. The above website also links to these documents and explains the changes that are required for DS-MDK debugging using ULINK*pro*.

#### Copy the Linux Image to an SD-Card

Once you have downloaded the zipped Linux Kernel image, you need unzip it before you can flash it onto an SD-Card. Windows users can use the open source tool **Win32 Disk Imager** from <u>http://win32diskimager.sourceforge.net/</u>.

Install and run the tool. To write the image to the memory card, specify the location of the image file, select the **Device** letter of the SD card and press the **Write** button:

👒 Win32 Disk Im	ager		_	$\Box$ $\times$
Image File				Device
are mage base mult	halowed 21 kill	root	tfs.sdcard	à 🔹
Copy MD5 Has	h:			
Progress				
	1	- 1		
Version: 0.9.5	Cancel	Read	Write	Exit

### **Hardware Connection**

#### i.MX 7 SABRE Board

- Insert the SD-Card with the Linux image into the slot labelled **SD1 BOOT**.
- Connect the ULINK*pro* debug adapter using the 10-pin ribbon cable to **J12 JTAG**.
- Connect your computer using a Micro-USB cable to the USB connector labelled **DEBUG UART**. Your Windows PC will automatically detect a dual USB serial port component and will install the required drivers.
- Connect the 5V power supply to **J1**.



### Verify Installation using Example Projects

Once you have selected, downloaded, and installed a Software Pack for your device, you can verify your installation using one of the examples provided in the Software Pack.

#### **Remote Processor Messaging Protocol Example**

The i.MX 7 Device Family Pack contains two example projects that show how the two processors communicate with each other using the remote processor messaging protocol (RPMSG) via a TTY serial device. The TTY device is installed on the Linux system using a Linux kernel module (imx\_rpmsg\_tty.ko).



The *Linux Application TTY* runs on the Cortex-A7 processor and writes a message to the TTY device. This message is shown in the terminal of the *RPMSG TTY RTX* application running on the Cortex-M4 processor. This application responds on the TTY device, which is read by the Linux application and shown in its console.

#### Copy the RPMSG TTY RTX Example Project

In the CMSIS Pack Manager perspective, select the Examples tab. Use filters in the toolbar to narrow the list of examples.

📕 Devices 😫 📓 Boards 🛛 🕀	E 🕐  🛣 🗖 🗖	🎒 Packs 📑 Examples 😒	Only show examples	from installed packs   🕐   🍣 🤌 🗢 🗖				
Search Device		Search Example						
Device	Summary ^	Example	Action	Description				
V 🖗 NXP	527 Devices	CMSIS-RTOS Blinky (MCIMX7D-SAB	RE) 🚸 Copy	CMSIS-RTOS based Blinky example for Cortex-M4				
🗸 🔧 i.MX 7 Series	1 Device	Linux Application TTY (MCIMX7D-S	ABRE) 🚸 Copy	Linux Application TTY example				
🗸 🔩 i.MX 7Dual	1 Device	RPMSG PingPong BM (MCIMX7D-S	ABRE) 🚸 Copy	Bare-Metal RPMSG PingPong example for Cortex-M4				
MCIMX7D	ARM Cortex-A7, ARM	RPMSG PingPong RTX (MCIMX7D-S	ABRE) 🚸 Copy	CMSIS-RTOS RTX and Bare-Metal RPMSG PingPong ex-				
> 🍕 K Series	1 Device	RPMSG TTY RTX (MCIMX7D-SABRE)	🚸 Сору	CMSIS-RTOS RTX TTY example for Cortex-M4				
> 🏤 K00 Series	2 Devices							

Click **Copy** next to the **RPMSG TTY RTX** example. A new window opens asking you to verify the selected example project:

Copy Example	e to Eclipse Workspace X
Example:	RPMSG TTY RTX
Pack:	Keil.iMX7D_DFP.0.1.10
Project Name:	RPMSG_TTY_RTX_M4
Project Location:	: C:\Users\USER\Documents\DS-MDK Workspace\RPMSG_TTY_RTX_M4
	Copy Cancel

CMSIS Pack Manager copies the example into your workspace and switches automatically to the C/C++ perspective:

C/C++ - RPMSG_TTY_RTX_M4/RPMSG_TTY_RTX_M4.rteconfig - Eclipse Platform							- 🗆 X
File Edit Source Refactor Navigate Search Proje	ct Run Window Help						
📑 • 🗒 🕼 🖻   🕸 • 🗞 • 📾   🗙   🔯   🗃	°° • °° • °° • † † • O •	•	• 😂 🖋 • 🗉	n (2 - ¥	• • ÷ ÷ ÷	•	Quick Access 🔡 🔛 🕸 🏶
Project Explorer 🙁 📄 😓 🤝 🖻 🗖	RPMSG_TTY_RTX_M4.rteconfig	×				- 0	🗄 Outline 😫 🛞 Make Ta 📟 🗖
Comparison Reproduction     Reproductin     Reproduction     Reproduction     Reproduction     Reproduc	💠 Components 🗹 Resolve					0	An and fan is net an illeble
V 🏂 RTE	Software Components	Sel.	Variant	Vendor	Version	Description	An outline is not available.
> 🗁 Board_Support	MCIMX7D:Cortex-M4			NXP		ARM Cortex-M4, 64 kB RAM, 32 k	
> 🗁 CMSIS	> 🚸 Board Support		MCIMX7D-SABRE	Keil	1.0.0	iMX7D SABRE Board	
> 🗁 Compiler	> 💠 CMSIS					Cortex Microcontroller Software I	
> 🗁 Device	> 💠 CMSIS Driver					Unified Device Drivers compliant	
> 🗁 OpenAMP	> 💠 Compiler					ARM Compiler Software Extension	
> h RTE_Components.h	> 🚸 Device					Startup, System Setup	
> 🖻 hardware_init.c	> 🚸 File System		MDK-Plus	Keil	6.8.0	File Access on various storage dev	
> 🖻 tty_rbc	> In the second seco		MDK-Plus	Segger	5.32.2	User Interface on graphical LCD d	
MCIMX7D_Cortex-M4.sct	> 💠 Network		MDK-Plus	Keil	7.2.0	IPv4 Networking using Ethernet o	
RPMSG_TTY_RTX_M4.rteconfig	> 💠 OpenAMP						
	> 🚸 USB		MDK-Plus	, Keil	6.8.0	USB Communication with various	

#### **Build the Application**

Build the project from the context menu in the **Project Explorer**:

陷 Project Explorer 🔀		RPMSG_TTY_RTX_M4.rteconfig	1 23
V 🖉 RPMSG_TTY_RT	'X_M4	🔺 Components 🗹 Resolve	
> ⋒) Inclu ✓ 🏠 RTE > 🍋 B –	New Go Into	> omponents IMX7D:Cortex-M4	Sel.
> 👝 C	Open in New Window	rd Support	
> 👝 C	Copy Paste	SIS SIS Driver	
> 🔓 R 🗶	Delete	ice	
> .c hard\ > .c tty_rt	Move Rename	System phics	
MCIN _	Nendrice	work	
🚸 RPM: 🔤	Import	enAMP	
2	Export		
	CMSIS C/C++ Project	>	
	Build Project		
	Clean Project		

The **Console** window shows information about the build process:

#### Copy and Build the Linux Application TTY

Switch back to the **CMSIS Pack Manager** perspective and copy the **Linux Application TTY** example project to your workspace.

Build the project from the context menu in the **Project Explorer.** The **Console** should show an error-free build:



The chapter **Debug Applications** on page 36 explains how to debug both applications using the DS-5 Debugger.

# **Documentation and Support**

DS-MDK provides online manuals and context-sensitive help. The **Help** menu opens the main help system that includes the *CMSIS C/C++ Development User's Guide*, the *ARM DS-MDK Documentation*, the *RSE User Guide*, and other reference guides.

Many dialogs have context-sensitive Help buttons that access the documentation and explain dialog options and settings.

If you have suggestions or you have discovered an issue with the software, please report them to us. Support and information channels are accessible at <a href="http://www.keil.com/support">www.keil.com/support</a>.

# **Eclipse IDE**

DS-MDK uses Eclipse for DS-5, an Integrated Development Environment (IDE) that combines the Eclipse IDE from the Eclipse Foundation with the compilation and debug technology of the ARM tools.

You can use Eclipse for DS-5 as a project manager to create, build, debug, monitor, and manage projects for ARM targets. It uses a single folder called a workspace to store files and folders related to specific projects.

Users can extend its abilities by installing plug-ins written for the Eclipse platform, such as the **CMSIS Pack Manager** and **Remote System Explorer**, included in DS-MDK.

### Workbench

The workbench is the main development environment where you can manage individual projects, associated sub-folders, and source files.

Each workbench window links to one workspace. If you want to use different workspaces at the same time, you can launch several workbench windows and link each one to a different workspace.

# Perspectives

A workbench contains multiple perspectives. Each perspective contains an initial set and layout of views. It aims at accomplishing a specific type of task, such as project creation and build, debugging, and Pack management. While working with DS-MDK, you will switch perspectives frequently. It is always possible to change a perspective layout and to add new views to it.

DS-MDK uses mainly these perspectives:

- C/C++ Perspective
- CMSIS Pack Manager Perspective
- Remote System Explorer Perspective
- DS-5 Debug Perspective

### **C/C++ Perspective**

This perspective is designed for working with  $C/C^{++}$  projects. By default, it consists of an editor area and views for project management, tasks, properties, and a console for messages.

The editor area of the C/C++ perspective in DS-MDK includes the Manage Run-Time Environment window that lets you select software components, target devices, and Software Packs for the current project.

It also features a graphical editor for files that have CMSIS Configuration Wizard annotations.



For more information, refer to the C/C++ Development User's Guide and the CMSIS C/C++ Development User's Guide available from the Eclipse help system (Help  $\rightarrow$  Help Contents).

The C/C++ perspective contains views that are tailored for specific needs.

#### **AXF File Viewer**

An AXF file is the executable image generated by the ARM linker that contains object code and debug information. Open it from the Project Explorer to inspect the contents of the image.

RPMSG_TTY_RTX_M4.axf 🐹				
Header		See.		
Machine class	ELFCLASS32 (32-bit)			
Data encoding	ELFDATA2LSB (Little endian)			
Header version	EV_CURRENT (Current version)			
Operating System ABI	none			
ABI version	0			
File type	<pre>ET_EXEC (Executable file) (2)</pre>			
Machine	EM_ARM (Advanced RISC Machines ARM)			
Image entry point	0x1FFFB299			
Flags	<pre>EF_ARM_HASENTRY + EF_ARM_ABI_FLOAT_SOFT (0x05000202)</pre>			
Header Size	52 bytes (0x34)			
Segment header entry size	32 bytes (0x20)			
Section header entry size	40 bytes (0x28)			
Program header entries	1			
Section header entries	16			
Program header offset	5497220			
Section header offset	5497252			
Section header string table index	15			
Header Sections Segments Symbol Table I	Disassembly			
reader sections segments symbol rable i	JISUSSCHIDIN			

#### **CMSIS Configuration Wizard**

Files containing configuration annotations may be modified using a graphical editor. Right-click on a file in the Project Explorer and select **Open With**  $\rightarrow$  **CMSIS Configuration Wizard**. Verify and adapt the contents directly in the graphical representation of the text file.

I RTX_Conf_CM.c ⊠						
≔ CMSIS Configuration Wizard		± = ?				
Option	Value	^				
✓ Thread Configuration						
Number of concurrent running user threads	6					
Default Thread stack size [bytes]	1024					
Main Thread stack size [bytes]	1024					
Number of threads with user-provided stack size	0					
Total stack size [bytes] for threads with user-provided stack size	0					
Stack overflow checking						
Stack usage watermark						
Processor mode for thread execution	Privileged mode					
<ul> <li>RTX Kernel Timer Tick Configuration</li> </ul>						
Use Cortex-M SysTick timer as RTX Kernel Timer	$\checkmark$					
RTOS Kernel Timer input clock frequency [Hz]	24000000					
RTX Timer tick interval value [us]	1000					
✓ System Configuration						
<ul> <li>Round-Robin Thread switching</li> </ul>	$\checkmark$					
Round-Robin Timeout (ticks)	5	~				
<		>				
Number of concurrent running user threads Defines max. number of user threads that will run at the same time. Default: 6						
Source Editor CMSIS Configuration Wizard						

#### **Scatter File Viewer**

Scatter files (\*.sct) are used to specify the memory map of an image to the linker. The **Scatter File Viewer** lets you inspect this text file in a graphical representation. Edit the file contents using the *filename.sct* tab (refer to **Adapt the Scatter File** on page 32).



#### **CMSIS Pack Manager Perspective**

The Pack Manager perspective offers the following functionality:

- Install or update Software Packs.
- List devices and boards that are supported by Software Packs.
- List example projects that can be copied into the Eclipse workspace.

To open this perspective, use the E icon and select CMSIS Pack Manager

Device Dat	abase	Availal	ole Pac	ks/Examples	Pack Properties
CMSIS Pack Manager - RPMSG_ Eile Edit Navigate Saach Br	TTY_RTX_M4/RPMSG_TTY_	RTX_M4.rteconfig - Eclipse Platform			- c ×
The cut Navigate Seann Pr	oject Ran Window He	ap			
: 🖬 = 🖾 🐨 🖾 👘 🖉	📽 e 🔯 e 🍓 🔹 🔗 🔹 e	21 × 22 × %+ <-> ×			Quick Access
📓 Devices 💥 🕃 Boards 💽	e 🛛 🦹 🗸 🗖 🗖	🛞 Packs 🌱 Examples 😫 🛛 Or	ily show exam	les from installed packs 🛛 🕐 😂 🧽 🌣 🗢 🗖	😑 Pack Properties 💥 📃 🗖
Search Device		Search Evample			
		ocare campie			Keil iMX7D DEP 0 1 10
Device	Summary ^	Example	Action	Description	× B Boards
V 🖗 NXP	527 Devices	CMSIS-RTOS Blinky (MCIMX7D-SABRE)	🚸 Сору	CMSIS-RTOS based Blinky example for Cortex-M4	NXP::MCIMX7D-SABRE
43 i.MX 7 Series	1 Device	Linux Application TTY (MCIMX7D-SABRE	🚸 Сору	Linux Application TTY example	V 🚸 Components
🗸 🔧 i.MX 7Dual	1 Device	RPMSG PingPong BM (MCIMX7D-SABRE	🚸 Сору	Bare-Metal RPMSG PingPong example for Cortex-M4	×      Device
MCIMX7D	ARM Cortex-A7, ARM	RPMSG PingPong RTX (MCIMX7D-SABRE	) 🚸 Copy	CMSIS-RTOS RTX and Bare-Metal RPMSG PingPong exan	Startup
> 🔧 K Series	1 Device	RPMSG TTY RTX (MCIMX7D-SABRE)	🚸 Сору	CMSIS-RTOS RTX TTY example for Cortex-M4	V Q IMX7D HAL
> 🎋 K00 Series	2 Devices				UART
> 🍕 K10 Series	23 Devices				MU
> 🍕 K20 Series	41 Devices				CCM
> 1\$ K30 Series	6 Devices				RDC
> 🍕 K40 Series	6 Devices				> @ Board Support
> 1\$ K50 Series	12 Devices				> 💡 OpenAMP
> 1\$ K60 Series	18 Devices				V Devices
> 1 K70 Series	4 Devices				🗸 🍕 I.MX 7 Series
> 13 K80 Series	2 Devices				🗸 🔩 I.MX 7Dual
> 1\$ KEAxx Series	6 Devices				MCIMX7D
> 1\$ KExx Series	21 Devices				✓ ➡ Examples
> 15 KLxx Series	55 Devices				> CMSIS-RTOS Blinky (MCIMX7D-S
> 15 KMbox Series	14 Devices				> Linux Application TTY (MCIMX70
> 1\$ KSxx Series	2 Devices				> RPMSG PingPong BM (MCIMX7E
> 15 KVxx Series	23 Devices				> RPMSG PingPong RTX (MCIMX7I)
> 15 KWx Series	14 Devices				<ul> <li>RPMSG TTY RTX (MCIMX7D-SAB</li> </ul>
> TS LPC800 Series	10 Devices				NXP::MCIMX7D-SABRE
> TS EPCTIOU Series	128 Devices				
> 15 LPC 1200 Series	12 Devices				
> 10 EPC 1500 Series	12 Devices				
> I DC 100 Series	21 Devices				
S CECTION Series	21 Devices				
S P C 1000 Series	16 Devices				
<	>	<		>	< >
Console 23 - Progress CDT Build Console [RPMSG_TTY_RT Total R0 Size (Code + Total RW Size (RW Data	X_M4] RO Data) a + ZI Data)	28292 ( 27.63k8) 31856 ( 31.11k8)		0 0 S	
Total ROM Size (Code +	RO Data + RW Data)	28412 ( 27.75kB)			*

For more information, refer to the *CMSIS C/C++ Development User's Guide* available from the Eclipse help system (**Help**  $\rightarrow$  **Help Contents**).

### **Remote System Explorer Perspective**

The Remote System Explorer (RSE) is a workbench perspective that allows you to connect and work with a variety of remote systems. With predefined plug-ins, you can look at remote file systems, transfer files between hosts, do remote search, execute commands and work with processes.

	e Systems Fil	e/S	System Prop	perties	Sourc	e Code Editor	Remo	te System I 	Details
Remote System E	Explorer - RemoteSystemsTempl	iles/1	10.41.1.130/home/root/kee	epbusy.sh - Eclipse Platform	1			-	
File Edit Navigate	e Search Project Run Wi	dow	Help						
🖻 • 🛛 🕤 🗠	B I ► II ■ N 3. つ	.e	≅ 🕱 🔯 🎋 र 🖸	- 🏊 - 🛷 - 🗉 🗉	철 - 월 - 두 수	• 🔿 •		Quick Access	Fil 🌞 🌰 🔢
I Remote Systems	😂 😪 Tram 👘		📄 keepbusy.sh 🖂					🗖 🔡 Outline 🖾	- 0
<ul> <li>Storal</li> <li>Storal</li> <li>Storal</li> <li>Storal</li> <li>Local File</li> <li>Local File</li> <li>Local File</li> <li>Storal</li> <li>Storal<!--</td--><td>s s bin bot dev bot dev come come bin bot dev come come come come bin bot dev come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come</td><td>×</td><td>1 while :; do ;; 2 3</td><td>done &amp;</td><td></td><td></td><td></td><td>An outline is not</td><td>available.</td></li></ul>	s s bin bot dev bot dev come come bin bot dev come come come come bin bot dev come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come come	×	1 while :; do ;; 2 3	done &				An outline is not	available.
> 🗀 > 🗀 ¥ 🍅	keepbusy.sh lib lost+found media								
> (= > (= * (=	keepbusy.sh     lib     lost+found     media     widia     mnt	Ļ						~	
> () > () ~ () ()	keepbusy.sh lib lost+found media b sd0 t mnt	• ~	¢				>	~	
<	keepbusy.sh b lost+found media sd0 met	, ,	<	aile 12 💭 Tacke			>	× هاد د د ا	
<ul> <li>&gt; </li> <li>&gt; </li> <li>&gt; </li> <li>&gt; </li> <li>Properties (2)</li> <li>6</li> </ul>	Remote Scratchpad	,	<	ails 🛛 🔊 Tasks			>	¥2 ⇔⇔@ 3	₩ ▽ □ □
> Control Cont	Recepturysh     Ib     Ib     Jost-found     media     adia     sd0     mnt     Remote Scratchpad	× = •	Remote System Det Root Connections Resource	ails 22 @ Tasks Parent orofile	Remote system type	Connection status	> Host name	ک کو کو ک	Description
> C > C Properties 23 G	k Ceepbusysh bib bib bib ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh ceepbusysh cee	× □ ↓ <	ج Remote System Det Resource کار محما	ails 23 🕢 Tasks Parent profile DESUTION - SETURON	Remote system type	Connection status	Host name	کی کے کہ کو کی کہ کو کر کے کہ کو کر کے کہ کو کر کے کہ	Description
> Canonical Path	Receptorysh Rob Stat-Found Stat-Found State State State Remote Scratchpad State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State State Stat	× □ ↓ <	Remote System Det Rot Connections Resource	ails 23 Tasks Parent profile DESKTOP-SFTOQPL DESKTOP-SFTOQPL DESKTOP-SFTOP	Remote system type Local	Connection status Some subsystems connected	Host name LOCALHOST	문 환 다 다 유 I 3 Default User ID USER	Description
<ul> <li>&gt; </li> <li>&gt; </li> <li>&gt; </li> <li>&gt; </li> <li>&gt; </li> <li>Propertig</li> <li>Canonical Path</li> <li>Classification</li> </ul>	Teepburysh     Bo     Iob	× □ × <	Remote System Det Root Connections Resource Cool Local IMK7_SABRE	ails 22 Tasks Parent profile DESKTOP-S6TOQPL DESKTOP-S6TOQPL	Remote system type Local SSH Only	Connection status Some subsystems connected Some subsystems connected	Host name LOCALHOST 10.41.1.130	Default User ID USER root	Description
<ul> <li>&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;</li></ul>	Exceptusysh     Indefinition	× □ ↓ <	Remote System Det Root Connections Resource Cocal Local	ails 22 Tasks Parent profile DESKTOP-S6TOQPL DESKTOP-S6TOQPL	Remote system type Local SSH Only	Connection status Some subsystems connected Some subsystems connected	Host name LOCALHOST 10.41.1.130	Default User D USER root	Description
<ul> <li>Construction</li> <li>Construction</li> <li>Property</li> <li>Canonical Path</li> <li>Classification</li> <li>Extension</li> <li>Filter string</li> </ul>	IC Repbusysh Io tot-found media Io tot-found media Io tot-found media Io tot-found Io tot-f	× = • *	Remote System Det Root Connections Resource Cocal Local LiMK7_SABRE	ails 23 Tasks Parent profile DESKTOP-S6TOQPL DESKTOP-S6TOQPL	Remote system type Local SSH Only	Connection status Some subsystems connected Some subsystems connected	Host name LOCALHOST 10.41.1.130	Default User ID USER root	Description
<ul> <li>Canonical Path Classification Extension Filter string Group</li> </ul>	Pice Reepbury.th Dis	> □ ▽ ▲	Remote System Det Root Connections Resource Cocal Local Lintx7_SABRE	alls 22 Tasks Parent profile DESKTOP-SETOQPL DESKTOP-SETOQPL	Remote system type Local SSH Only	Connection status Some subsystems connected Some subsystems connected	Host name LOCALHOST 10.41.1.130	Default User ID USER root	Description
<ul> <li>Canonical Path Classification</li> <li>Extension</li> <li>Extension</li> <li>Filter string</li> <li>Group</li> <li>Hidden</li> </ul>	ip: Reepbury.th in: Tother of the service of the	> □	Remote System Det Root Connections Resource Cocal MXT_SABRE	alis 23 Tasks Parent profile DESKTOP-SETOQPL DESKTOP-SETOQPL	Remote system type Local SSH Only	Connection status Some subsystems connected Some subsystems connected	Hest name LOCALHOST 10.41.1.130	Default User ID	Description
Properties 22 3     Property     Canonical Path     Classification     Extension     Filter string     Group     Hidden     Last modified	Constraints Const	~	Renote System Det Raot Connections Resource Clocal Clocal Clocal Clocal	ails 22 Tasks Parent profile DESKTOP-SATOQPL	Remote system type Local SSH Only	Connection status Some subsystems connected Some subsystems connected	Host name LOCALHOST 10.41.1.130	を Default User ID USER root	Description
Properties 22 6 Properties 22 6 Property Canonical Path Classification Extension Filter string Group Hidden Hidden Last modified Location	ip: Respbury.th in to control of the service of		Remote System Det Root Connections Resource Local Chi.MAT, SABRE	alis 22 Parsts Parstt profile DESITOR-SATOOPL DESITOR-SATOOPL	Remote system type Local SSH Only	Connection status Some subsystems connected Some subsystems connected	Host name LOCALHOST 10.41.1.130	Default User ID USER root	Description
<ul> <li>Canonical Path</li> <li>Casonical Path</li> <li>Cassification</li> <li>Filter string</li> <li>Group</li> <li>Hidden</li> <li>Last modified</li> <li>Location</li> <li>Name</li> </ul>	Exceptusy.th		Renote System Det Rect Connections Resource Catal Catal Catal Catal	alis 22 Parets Parent profile DESKTOP-SATOQPL DESKTOP-SATOQPL	Remote system type Local SSH Only	Connection status Some subsystems connected Some subsystems connected	Host name LOCALHOST 10.41.1.130	を Default User ID USER root	Description
<ul> <li>Construction</li> <li>Property</li> <li>Canonical Path</li> <li>Classification</li> <li>Extension</li> <li>Filter string</li> <li>Group</li> <li>Hidden</li> <li>Location</li> <li>Name</li> <li>Number of child</li> </ul>	IDE Resplaysh IDE Tothond media IDE Tothond media IDE Tothond IDE Tothond I	> 0 ▼ ▲	≪ Remote System Det Resource E Local C+1.MA7_SABRE	alis 22 Tasks Parent profile DESITOP-SATOOPL DESITOP-SATOOPL	Remote system type Local SSH Only	Connection status Some subsystems connected Some subsystems connected	Host name LOCALHOST 10.41.1.130	Default User ID USER root	Description
<ul> <li>Canonical Path</li> <li>Canonical Path</li> <li>Cassification</li> <li>Filter string</li> <li>Group</li> <li>Hidden</li> <li>Last modified</li> <li>Location</li> <li>Name of hild</li> <li>Owner</li> </ul>	Bepusyah     Bepusyah     Bet	~	Renote System Det Root Connections Resource Cola Local	als 22 Parts Parent profile DESKTOP-SSTOQPL DESKTOP-SSTOQPL	Remote system type Local SSH Only	Connection status Some subsystems connected Some subsystems connected	Host name LOCALHOST 10.41.1.130	を Default User ID USER root	Description
<ul> <li>&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;&gt;</li></ul>	IDE Resplaysh IDE Tothond media IDE Tothond media IDE Tothond Tothond IDE Tothond IDE To	>	≪ Remote System Det Resource © Local Call.MAT_SABRE	alis 22 Tasks Parent profile DESKTOP-S6TOOPL DESKTOP-S6TOOPL	Remote system type Local SSH Only	Connection status Some subsystems connected Some subsystems connected	Host name LOCALHOST 10.41.1130	Default User ID USER root	Description

For more information, refer to the *RSE User Guide* in the Eclipse help system (Help  $\rightarrow$  Help Contents).

#### **DS-5** Debug Perspective

Using DS-5 Debugger, you can debug bare-metal, RTOS, and Linux applications with comprehensive and intuitive views, including synchronized source and disassembly, call stack, memory, registers, expressions, variables, threads, breakpoints, and trace.



For more information, refer to the *ARM DS-5 Debugger Documentation* in the *ARM DS-MDK Documentation* available from the Eclipse help system (Help  $\rightarrow$  Help Contents).

# **Create Cortex-M4 Applications**

This chapter guides you through the steps required to create and modify projects for the Cortex-M target in a heterogeneous system. The tutorial creates a project called *Blinky* using the real-time operating system CMSIS-RTOS RTX.

# **Blinky with CMSIS-RTOS RTX**

The section explains the creation of the project using the following steps:

- Setup the Project: create a project and select the microcontroller device along with the relevant CMSIS components.
- Select Software Components: choose the required software components for the application.
- Customize the CMSIS-RTOS RTX Kernel: adapt the RTOS kernel.
- Create the Source Code Files: add and create the application files.
- Build the Application Image: compile and link the application.

For the *Blinky* project, you will create and modify the *main.c* source file which contains the *main()* function that initializes the RTOS kernel, the peripherals, and starts thread execution. In addition, you will configure the system clock and the CMSIS-RTOS RTX.

### **Setup the Project**

From the Eclipse menu bar, choose File  $\rightarrow$  New  $\rightarrow$  C Project:

C++ Project	– 🗆 X
C++ Project Create C++ project of selected type	
Project name: Blinky ✓ Use default location Location: C:\Users\USER\Documents' Choose file system: default	\DS-MDK Workspace\Blinky Browse
Project type:	Toolchains: ARM Compiler 5.06u2 ARM Compiler 6.4 GCC 4.x larm-linux-onueabihfl (built-
<ul> <li>CMSIS C/C++ Project</li> <li>Show project types and toolchains of</li> </ul>	v < >>

Select **CMSIS RTE C/C++ Project**, enter a project name (for example **Blinky**) and click **Next**. In the following window, you can select to create a default *main.c* file. Do not use this option. We will add a *main.c* template file later from a Software Pack, so click again **Next**.

In the following step, select your target device:

🖨 C++	Project				-			×
Select D	evice						- TUTUTE	00000
Device:	MCIMX7D:Cortex-M4			CPU:	ARM Cortex-M4			
Vendor:	NXP			Max. Clock:				
Pack:	Keil.iMX7D_DFP.0.1.7			Memory:	64 kB RAM, 32 kB	ROM		
URL:	http://www.keil.com/c	Id2/nxp/mcimx7	<u>d</u>	FPU:	none			$\sim$
Search:				Endian:	Little-endian			$\sim$
	NXP i.MX 7 Series i.MX 7Dual i.MX 7Dual MCIMX7D MCIMX7 MCIMX7	D:Cortex-A7 D:Cortex-M4	*	The i.MX 7 an advance Cortex-A7 up to 1 GH core. - He Architectur	Dual family of proc ed implementation core, which operate z, as well as the ARI terogeneous Multia re, up to Dual Corte	essors fe of the A es at spe M Corte Core Pro x-A7 an	eatures RM eds of x-M4 ocessing d	< >
?		< Back		Next >	Finish		Cancel	

Select the NXP  $\rightarrow$  i.MX 7 Series  $\rightarrow$  i.MX Dual  $\rightarrow$  MCIMX7D:Cortex-M4 device and click Finish. The C/C++ Perspective will open and show the current project:

C/C++ - Blinky/Blinky.rteconfig - Eclipse Platform								-		×
Eile Edit Source Refactor Navigate Search Project	ct <u>R</u> un <u>W</u> indow <u>H</u> elp									
📑 🕶 📰 🕼 🗁   🏵 🕶 🗞 🕶 🗟 i 🗙 i 🔂 i 🔂 🕶	62 • 🖻 • 69 • 🕸 • 💽	- 🎍	• 🙋 🖋 • 🗉	n i a + 9		<b>*</b>	Quick Access	🖻	<b>1</b> 2	۵
Project Explorer 😫 📄 🗳 👻 🖻 🗖	💠 Blinky.rteconfig 😒					- 0	📴 Outline 🔀 🛞 N	fake Ta.		· 🗆
S Blinky     Solucian	💠 Components 🕑 Resolve					0 📓	An endline is not enable			
RTE     RTE     RTE	Software Components	Sel.	Variant	Vendor	Version	Description	All outline is not availa	ioic.		
Blinky.rteconfig	> Seard Support		MCIMX7D-SABRE	Keil	1.0.0	iMX7D SABRE Board				
MCIMX7D_Cortex-M4.sct						Cortex Microcontroller Software I Unified Device Drivers compliant				
	> Compiler					ARM Compiler Software Extension				
	> Series		MDK-Plus	Keil	6.7.0	File Access on various storage de				
	> 💠 Graphics > 💠 Network		MDK-Plus MDK-Plus	Segger Keil	5.32.2	User Interface on graphical LCD d IPv4 Networking using Ethernet o				
	> I OpenAMP > I USB		MDK-Plus	Keil	6.7.0	USB Communication with various				
				1						
	<					>				
	Validation Output				Description					
	<					>				
	Components Device Packs									_
	Problems 🔊 Tasks 🗳 Cor	sole (	3 Properties				🗟 🔝 🛛 🖻	- 📑	•	· 🗆
	11:00:06 **** Updating pro	oject	Blinky							^
	Updating resources Updating build settings Replact updated successful	11								
	Project updated successio.									
<	<									>

### **Select Software Components**

For the Blinky project based on CMSIS-RTOS RTX, you need to select the following components:

- CMSIS:RTOS (API):Keil RTX.
- Device:i.MX7D HAL:CCM
- Device:i.MX7D HAL:RDC
- Device:i.MX7D HAL:UART
- Compiler: I/O:STDERR configured as variant User
- Compiler:I/O:STDIN configured as variant User
- Compiler:I/O:STDOUT configured as variant User
- Board Support:iMX7D SABRE Board:HW INIT
- Board Support:iMX7D SABRE Board:User I/O Redirect

Use the **Resolve** button to add other required components automatically. Finally, save your selection:

🚸 *Blinky.rteconfig 🔀							
Components* Resolv	e Re	solve validatio	n messa	iges	Save when done) 🔚	)	
Software Components	Sel.	Variant	Vendor	Versio	on Description	^	
MCIMX7D:Cortex-M4	- 1		NXP		ARM Cortex-M4, 64 kB RAM, 32 kB ROM		
V 🚸 Board Support		MCIMX7D-SABRE	Keil	1.0.0	iMX7D SABRE Board		
iMX7D SABRE Board					1		
HW INIT					Board specific settings for hardware initialization		
User I/O Redirect	t 🖸				User I/O Redirect to UART		
🗸 🚸 CMSIS	T				Cortex Microcontroller Software Interface Components		
CORE	b		ARM	4.3.0	CMSIS-CORE for Cortex-M, SC000, and SC300		
DSP			ARM	1.4.6	CMSIS-DSP Library for Cortex-M, SC000, and SC300		
V 🔶 RTOS (API)				1.0	CMSIS-RTOS API for Cortex-M, SC000, and SC300		
Keil RTX			ARM	4.80.0	CMSIS-RTOS RTX implementation for Cortex-M, SC000, and		
> 🚸 CMSIS Driver					Unified Device Drivers compliant to CMSIS-Driver Specifica		
🗸 🚸 Compiler					ARM Compiler Software Extensions		
Event Messaging		DAP	Keil	1.0.0	Event Messaging using Debug Access Port (DAP)		
✓ ♦ I/O							
🖉 File		File System	Keil	1.1.0	Use retargeting together with the File System component		
STDERR	$\checkmark$	User	Keil	1.1.0	Redirect STDERR to a user defined output target (USART, Gr		
STDIN	$\checkmark$	User	Keil	1.1.0	Retrieve STDIN from a user specified input source (USART,		
STDOUT	$\checkmark$	User	Keil	1.1.0	Redirect STDOUT to a user defined output target (USART, G		
🖉 TTY		Breakpoint	Keil	1.1.0	Stop program execution at a breakpoint when using TTY		
🗸 🚸 Device					Startup, System Setup		
🗸 🔶 iMX7D HAL							
🖉 ССМ			Keil	1.0.0	Clock Control Module		
🖉 MU			Keil	1.0.0	Messaging Unit		
RDC			Keil	1.0.0	Resource Domain Controller		
🖉 UART			Keil	1.0.0	Universal Asynchronous Receiver/Transmitter	۷	
<					>		
Validation Output					Description	^	
ARM CMSIS BTOS Keil F	тх				Additional software components required		
v A require Colass="Dev	ice" Co	roun-"Startun"			Select component from list		
Keil-Device Start	un	ioup= startup			NXP iMX7D CM4 devices		
Keil MCIMX7D-SABRE	Soard Si	innort.iMX7D SABRE	Board HV		Additional software components required		
v A require Cclass="CM	SIS" Ca	roun="CORF"	. Dourdan in		Select component from list		
		.cop- conc				~	
<					>		
Components Device Packs							

#### NOTE

Saving the RTE configuration triggers a project update and the selected software components become instantly visible in the Project Explorer.

### **Configure CMSIS-RTOS RTX Kernel**

In the project, expand the group **RTE:CMSIS**, right-click on the file  $RTX\_Conf\_CM.c$ , and select **Open With**  $\rightarrow$  **CMSIS Configuration Wizard**. Change the following settings:

- Default Thread stack size [bytes]
   512
- Main Thread stack size [bytes]
   512

RTOS Kernel Timer input clock frequency [Hz] 240000000

Intion	Value		
	value		
I hread Configuration	<i>c</i>		
Number of concurrent running user threads	0		
Default Thread stack size [bytes]	512	-	
Main Thread stack size [bytes]	512	-	
Number of threads with user-provided stack size	0		
Total stack size [bytes] for threads with user-provide	0	-	
Stack overflow checking	$\checkmark$		
Stack usage watermark			
Processor mode for thread execution	Privileged mode	e ,	
<ul> <li>RTX Kernel Timer Tick Configuration</li> </ul>			
Use Cortex-M SysTick timer as RTX Kernel Timer	$\checkmark$		
RTOS Kernel Timer input clock frequency [Hz]	24000000		
RTX Timer tick interval value [us]	1000		
<ul> <li>System Configuration</li> </ul>			
> Round-Robin Thread switching	$\checkmark$		
> User Timers	$\checkmark$		
ISR FIFO Queue size	16 entries		
<b>FOS Kernel Timer input clock frequency [Hz]</b> efines the input frequency of the RTOS Kernel Timer. hen the Cortex-M SysTick timer is used, the input clock on most systems identical with the core clock.	·		

Save the file.

#### NOTE

If you have opened a file with the CMSIS Configuration Wizard once, your choice is stored and the file will be opened in this view automatically next time.

#### **Create the Source Code Files**

Add your application code using pre-configured User Code Templates containing routines that resemble the functionality of the software component. Right-click on the project and select New  $\rightarrow$  Files from CMSIS Template.

🖨 New Fil			×					
CMSIS Use								
This wizar								
Project:	Blinky			Brov	/se			
Compone	ent	Name						
🗸 🚸 CN	VISIS							
•	RTOS.Keil RTX	CMSIS-RTOS 'main' function						
*	RTOS.Keil RTX	CMSIS-RTOS Mail Queue						
*	RTOS.Keil RTX	CMSIS-RTOS Memory Pool						
*	RTOS.Keil RTX	CMSIS-RTOS Message Queue						
*	RTOS.Keil RTX	CMSIS-RTOS Mutex						
*	RTOS.Keil RTX	CMSIS-RTOS Semaphore						
*	RTOS.Keil RTX	CMSIS-RTOS Thread						
*	RTOS.Keil RTX	CMSIS-RTOS Timer						
*	RTOS.Keil RTX	CMSIS-RTOS User SVC			_			
Location:	/Blinky			Brov	vse			
File name:	osObjects.h main.c							
?		Finis	h	Cancel				

Expand the software component **CMSIS** and select the template **CMSIS-RTOS 'main' function**. Click **Finish**. Add application specific code to the file *main.c*:

```
/*-----
* CMSIS-RTOS 'main' function template
*-----*/
#define osObjectsPublic
                                 // define objects in main module
#include "osObjects.h"
                                 // RTOS object definitions
#ifdef RTE
 #include "RTE Components.h"
                                 // Component selection
 rdef RTE_CMSIS_RTOS // when RTE component CMSIS RTOS is used
#include "cmsis_os.h" // CMSIS_RTOS is
#endif
#ifdef RTE CMSIS RTOS
#endif
#include "system iMX7D M4.h"
#include "retarget io.h"
#include "board.h"
#include <stdio.h>
```

```
osThreadId tid threadA;
                                 /* Thread id of thread A */
/*-----
*
     Thread A
*-----*/
void threadA (void const *argument) {
 volatile int a = 0;
 for (;;) {
        osDelay(750);
        printf("Blinky threadA: Hello World!\n");
 }
ł
osThreadDef(threadA, osPriorityNormal, 1, 0);
/*
* main: initialize and start the system
*/
int main (void) {
 /* Board specific RDC settings */
 BOARD RdcInit();
 /* Board specific clock settings */
 BOARD ClockInit();
 SystemCoreClockUpdate();
 InitRetargetIOUSART();
 tid threadA = osThreadCreate(osThread(threadA), NULL);
#ifdef RTE CMSIS RTOS
                                 // when using CMSIS RTOS
 osKernelInitialize ();
                                      // initialize CMSIS-RTOS
#endif
 /* Initialize device HAL here */
                                 // when using CMSIS RTOS
#ifdef RTE CMSIS RTOS
 osKernelStart ();
                                  // start thread execution
#endif
 /* Infinite loop */
 while (1)
 £
   /* Add application code here */
   osDelay(1000);
   printf("Blinky main loop: Hello World!\n");
 // initialize peripherals here
 // create 'thread' functions that start executing,
 // example: tid name = osThreadCreate (osThread(name), NULL);
 osKernelStart ();
                                      // start thread execution
 }
```

### Adapt the Scatter File

On the i.MX 7 devices, several types of memory are available. For deterministic, real-time behavior, the Cortex-M4 provides local Tightly Coupled Memory (TCM), which provides low-latency access. Multiple on-chip RAM areas (OCRAM) are available, which are larger, but not as fast.

The following table shows the memories and their load addresses for the different processors:

Region	Size	Cortex-A7	Cortex-M4 (Code Bus)
OCRAM	128KB	0x00900000-0x0091FFFF	0x00900000-0x0091FFFF
TCMU	32KB	0x00800000-0x00807FFF	
TCML	32KB	0x007F8000-0x007FFFFF	0x1FFF8000-0x1FFFFFFF
OCRAM_S	32KB	0x00180000-0x00187FFF	0x00000000-0x00007FFF/ 0x00180000-0x00187FFF

By default, the scatter file template uses the start address 0x0 for the load region command. To put the Cortex-M4 code into the TCM, change the address of the load region to 0x1FFF8000:

### **Build the Cortex-M4 Image**

Right-click on the project name and select **Build Project** to build the application. This step compiles and links all related source files. The **Console** shows information about the build process. An error-free build displays program size information:

**Debug Cortex-M4 Application** on page 37 guides you through the required steps to connect your evaluation board to the workstation and to debug the application on the target hardware.

# **Create Linux Applications**

This chapter guides you through the steps required to create and modify projects for an ARM Cortex-A class device running Linux:

- Setup the Project: create a project.
- Build the Application Image: compile and link the application.

# **Setup the Project**

From the Eclipse menu bar, choose File  $\rightarrow$  New  $\rightarrow$  C Project. In the upcoming window, select the Hello World ANCI C Project:

🖨 C Project		- 🗆 X
C Project Create C project of selected type		
Project name:       Hello_World         ✓       Use default location         Location:       C:\Users\USER\Documents\DS         Choose file system:       default	-MDK	Workspace\Hello_World Browse
Project type:		Toolchains:
Executable     Empty Project     Hello World ANSI C Project     GMSIS C/C++ Project	^	ARM Compiler 5.06u2 ARM Compiler 6.4 GCC 4.x [arm-linux-gnueabihf] (built-in)
Show project types and toolchains only	/ if the	y are supported on the platform

Enter a project name (for example Hello\_World) and make sure that the GCC [...] (built-in) toolchain is selected before clicking Finish.

The C/C++ Perspective will open and show the current project:



### **Build the Application Image**

Right-click on the project name and select **Build Project** to build the application. This step compiles and links all related source files. The **Console** shows information about the build process



The chapter **Debug Linux Application** on page 42 guides you through the required steps to connect your evaluation board to the workstation and to download the application to the target hardware.

# **Debug Applications**

The DS-5 Debugger can verify all software applications that execute on a heterogeneous computer system. Complete system visibility is enabled using multiple simultaneous debug connections:



DS-5 Debugger

Heterogeneous System

- The Cortex-M application is debugged using a ULINK*pro* debug unit (refer to <u>www.keil.com/ulink</u> for more information). Users can analyze the microcontroller application using RTOS aware-debugging and peripheral views.
- The **Cortex-A Linux kernel** is also debugged using a ULINK*pro* debug unit. The debugger lists kernel threads and processes.
- The Cortex-A Linux application is debugged via <u>gdbserver</u>. The debugger supports multi-threaded application debugging and shows pending breakpoints on loadable modules and shared libraries.

# **Preparing the Terminal Views**

Many applications use a serial connection to display messages. To be able to view these messages, use a **Terminal** window that shows data coming from serial COM ports.

The i.MX 7 SABRE development board contains a dual USB serial port device that offers two independent COM ports. Connect the board to your computer and Windows will install the drivers that will add two new USB Serial Ports to your system. Check the exact numbers in the Windows **Device Manager** (to open it, type "device manager" in the Windows search bar):



The smaller number is the COM port of the Cortex-A processor, while the larger number is the COM port of the Cortex-M processor. To open a Terminal view, go to Window  $\rightarrow$  Show View  $\rightarrow$  Other... Select Terminal  $\rightarrow$  Terminal and click OK.

Open the settings dialog from the toolbar of the Terminal 1 window:



Change the settings to the following:

- View Title: Terminal Linux
- Connection Type: Serial
- Port: Use the first of the new COM ports
- Baud Rate: 115200

Click OK. Press the RST button on the development board to observe the boot process in the Terminal window. Send any keyboard key to the terminal window to interrupt the boot process:



#### NOTE

You must halt the boot loader at this point to be able to launch the Cortex-M4 debug session. Before starting to debug, copy and build the Linux application.

Add another Terminal view to display the output of the Cortex-M4 processor. Simply use the drop-down selector next to the **New Terminal Connection in Current View...** icon 🚰 🖛 and select **New terminal View**:

```
New Terminal Connection in Current View...
New Terminal View
```

Select the larger COM port number and leave the other settings as they are. Name the Terminal view **Terminal M4**.

### **Debug Cortex-M4 Application**

This section explains how to debug the microcontroller application running on the Cortex-M4. If you are debugging the *Blinky* application from the previous chapter, execute the following steps using that project. Here, we will continue with the *RPMSG\_TTY\_RTX\_M4* project from the **Verify Installation using Example Project** chapter.

### Stop in U-Boot

To be able to connect to the target, you need to stop U-Boot before it is actually loading the Linux kernel. Restart/reset the device and observe the bootloader output on the **Terminal Linux**. Press any key before the autoboot countdown expires:

```
- N 💦 🚍 🏣 🚮 🖉 🕶 🔛 🗖 🕬
🦹 Problems 🧔 Tasks 📃 Console 🔲 Properties 🦧 Terminal CM4 🦧 Terminal Linux 🔀 🗌
Serial: (COM14, 115200, 8, 1, None, None - CONNECTED) - Encoding: (ISO-8859-1)
U-Boot 2015.04-imx_v2015.04_4.1.15_1.0.0_ga+gd7d7c43 (Jul 01 2016 - 11:47:23)
CPU: Freescale i.MX7D rev1.2 at 792 MHz
CPU: Temperature 47 C
Reset cause: POR
Board: i.MX7D SABRESD RevB
I2C: ready
DRAM: 1 GiB
PMIC: PFUZE300 DEV_ID=0x30 REV_ID=0x11
MMC: FSL_SDHC: 0, FSL_SDHC: 1
No panel detected: default to TFT43AB
Display: TFT43AB (480x272)
Video: 480x272x24
In: serial
Out: serial
Ecc
     serial
switch to partitions #0, OK
mmc0 is current device
Net:
      FEC0
Normal Boot
Hit any key to stop autoboot: 0
=>
```

### **Configure CMSIS DS-5 Debugger**

Launch the DS-5 Debugger using the Cortex-M4 project context menu.

Right-click the **RPMSG\_TTY\_RTX\_M4** project and select **Debug As** → **CMSIS DS-5 Debugger**:

Debug Configurations							
Create, manage, and run configuration Launch a DS-5 debugging session using a	<b>ons</b> CMSIS DS-5 Debugger project.	To-					
<ul> <li>We filter text</li> <li>C/C++ Application</li> <li>C/C++ Attach to Application</li> <li>C/C++ Postmortem Debugger</li> <li>C/C++ Remote Application</li> <li>C/C++ Remote Application</li> <li>C/C++ Remote Application</li> <li>RPMSG_TTY_RTX_M4</li> <li>DS-5 Debugger</li> <li>IronPython Run</li> <li>IronPython unittest</li> <li>Java Applet</li> <li>Java Applet</li> <li>Java Application</li> <li>Ju JUnit</li> <li>Jython run</li> <li>Jython run</li> <li>PyDev Django</li> <li>PyDev Google App Run</li> <li>Python Run<!--</th--><th>Name: RPMSG_TTY_RTX_M4  Connection Advanced OS Awareness  Project Selection  RPMSG_TTY_RTX_M4  Connection Settings Connection Type ULINKpro Connection Address P1445217:Keil ULINKpro  Target Configuration</th><th>Browse Apply Revert</th></li></ul>	Name: RPMSG_TTY_RTX_M4  Connection Advanced OS Awareness  Project Selection  RPMSG_TTY_RTX_M4  Connection Settings Connection Type ULINKpro Connection Address P1445217:Keil ULINKpro  Target Configuration	Browse Apply Revert					
?		Debug Close					

#### Connection

Verify the **Connection Settings** and ensure that ULINK*pro* is correctly detected. If in doubt, use **Browse...** to list available debug adapters.

Click on **Target Configuration...** to setup the Debug and Trace Services Layer (DTSL).

Debug and Trace Services Layer (DTS)	SL) Configuration for ULINKpro	_	×
Debug and Trace Services Layer (D Add, edit or choose a DTSL configuration	TSL) Configuration for ULINKpro		
offault	Name of configuration: default Trace Capture Cortex-A7 Cortex-M4 ETR ITM CTI Synchronization Enable Cortex-M4 core trace Enable Cortex-M4 trace Enable ETM Timestamps		

- On the Cortex-A7 tab, disable all trace options to avoid buffer overflows.
- On the Cortex-M4 tab, Enable Cortex-M4 core trace.

#### **OS Awareness**

In the **OS Awareness** tab chose the real-time operating system used in your application using the drop-down menu.

Debug Configurations		×
Create, manage, and run configurati Launch a DS-5 debugging session using	i <b>ons</b> a CMSIS DS-5 Debugger project.	Ť.
type filter text         c C/C++ Attach to Application         c C/C++ Remote Application         c C/C++ Remote Application         c C/S-+ Remote Application         c CMSIS DS-5 Debugger         c RPMSG_TTY_RTX_M4         b DS-5 Debugger         c TronPython Run         d' IronPython nunittest         j Java Applet            Filter matched 21 of 21 items	Name: RPMSG_TTY_RTX_M4	Revert
?	Debug	Close

To start debugging, click **Debug**.

#### NOTE

The error message "Failed to launch debug server" most likely indicates that an incorrect ULINKpro connection address is selected.

### **Run Cortex-M4 Application**

Starting the debugger, changes to the DS-5 Debug perspective.

The application loads and runs until main. To start the Cortex-M4 application click **Run** in the **Debug Control** view.



Observe the output of the application in the Terminal M4 window.

#### NOTE

You can add another Terminal view to the debug perspective by simply opening it using **Window**  $\rightarrow$  Show View  $\rightarrow$  Terminal.

# **Debug Linux Application**

This section explains how to debug the Linux application running on the Cortex-A7. If you are debugging the *Hello\_World* application from the previous chapter, execute the following steps using that project. Here, we will continue with the *Linux Application TTY* project from the **Verify Installation using Example Project** chapter.

For Linux application debugging, the DS-5 Debugger supports connections to the target using gdbserver.

Before connecting, you must:

- Set up the target with Linux installed and booted. Refer to **Install the Linux Image** on page 12.
- Obtain the target IP address or name for the connection between the debugger and the debug hardware adapter. If the target is in your local subnet, click Browse and select your target.

Next, you should set up a Remote Systems Explorer (RSE) connection to the target to download the application onto the target's file system.

### **Setup RSE Connection**

Go to Window  $\rightarrow$  Open Perspective  $\rightarrow$  Other..., then select Remote System Explorer. Create a new connection using the 4° button. Select SSH Only and click Next.

RSE is using Ethernet to communicate with the target, thus you need to enter the target's IP address into the **Host Name** field. Enter a meaningful name in the **Connection name** box:

New Connection		_		×	
Remote SSH Only Syste	em Connection				
Define connection information					
Parent profile:	DESKTOF UTTER			~	
Host name:	10.41.			~	
Connection name:	iMX7_SABRE				
Description:					
Verify host name <u>Configure proxy settings</u>					
? < <u>B</u> ack	<u>N</u> ext >	<u>F</u> inish	Canc	el	

Click Finish. Your connection shows up in the **Remote Systems** window:



### **Boot Linux**

#### NOTE

If you are debugging a microcontroller application simultaneously, you need to run the Cortex-M4 application, otherwise the Linux Terminal will not be accessible and you will not be able to boot Linux.

In the Terminal Linux enter "boot" to start the Linux system:

💼 App Console 💼 Target Console   Error Log 🧬 Terminal Linux 🕺 💦 🛤 🛅 🌆 🖉 + 😭 🖛 🗶 🖛	
Serial: (COM14, 115200, 8, 1, None, None - CONNECTED) - Encoding: (ISO-8859-1)	
Warning: FEC0 MAC addresses don't match: Address in SROM is 00:04:9f:04:49:80 Address in environment is 00:04:9f:04:01:d3	*
Normal Boot Hit any key to stop autoboot: 0 => boot	

Once the Kernel is running, log in as root (no password required).

#### **Configure DS-5 Debugger**

Right-click on the project *Linux Application TTY* and select **Debug As**  $\rightarrow$  **Debug Configurations...** In the Debug Configurations window, select DS-5 Debugger and then press the  $\square$  icon to create a new debug configuration. Name it *GDB Debug* and select in the **Connection** tab **Linux Application Debug**  $\rightarrow$ **Application Debug**  $\rightarrow$  **Connections via gdbserver**  $\rightarrow$  **Download and debug application**. The RSE connection from the previous step shows up:



On the **Files** tab, in **Target Configuration**, select the workspace build target for **Application on host to download**. Select an **existing** directory on the target file system, e.g. /home/root/tmp as the **Target download directory**.

Select an existing directory on the target file system, e.g. /home/root/tmp as the Target working directory (use the same directory as for Target download directory).

ne:	GDB Debug
Cor	nnection 🔚 Files 🛛 🏘 Debugger 🆓 OS Awareness 🕺 Arguments 🗷 Environment
Tar	aet Configuration
Ар	plication on host to download:
<b>\$</b> {	workspace_loc:/Linux Application TTY/Debug/Linux Application TTY}
Fi	le System) Workspace) 🕼 Load symbols
Tar	get download directory:
/h	ome/root/tmp
Tar	get working directory:
/h	ome/root/tmp

On the **Debugger** tab, under **Run Control** select **Debug from symbol** "main". Click **Debug**.

#### **Run Linux Application**

In the **Terminal Linux**, load the kernel module that communicates with the Cortex-M4 application using the following command:

```
root@imv7dsabresd:~# modprobe -v imx rpmsg tty
```

The kernel module should be loaded as shown below:

```
insmod /lib/modules/4.1.15-
1.1.0+ga4d2a08/kernel/drivers/rpmsg/imx_rpmsg_tty.ko
imx_rpmsg_tty rpmsg0: new channel: 0x400 -> 0x0!
Install rpmsg tty driver!
```

Now you can run the Linux application by clicking the **Continue** button. The **App Console** shows the application's messages:



Similarly, the **Terminal M4** shows the output of the microcontroller application:



#### NOTE

You can add another Terminal view to the debug perspective by simply opening it using **Window**  $\rightarrow$  Show View  $\rightarrow$  Terminal.

# **Store Cortex-M4 Image**

To store the Cortex-M4 Image for execution at start up use the following steps:

- 1. Create BIN image file using the fromelf utility application.
- 2. Store this BIN image on SD card in the boot partition
- 3. Setup the U-Boot environment to start-up the BIN image file.

### Create a Cortex-M4 BIN Image File

fromelf --bin --output "Blinky.bin" "Blinky.axf"

Right-click the project and select **Properties**  $\rightarrow$  C/C++ Build  $\rightarrow$  Settings. In the the Build Steps enter under Post-build steps the Command:

Properties for Blinky		– – ×	(
type filter text	Settings		•
<ul> <li>Resource</li> <li>Builders</li> <li>C/C++ Build</li> <li>Build Variables</li> <li>Environment</li> </ul>	Configuration: Debug [Active] V Manage C	Configurations	^
Logging Settings Tool Chain Editor > C/C++ General Project References Run/Debug Settings	Tool Settings       Perebuild Steps       Binary Parsers       Error Parsers         Pre-build steps       Command:       Description:       Description:	~	
	Post-build steps Command: fromelfbinoutput "Blinky.bin" "Blinky.axf"	~	
	Description:	~	v
?	OK	Cancel	

#### NOTE

This example shows the steps for the Blinky application from section **Blinky with** CMSIS-RTOS RTX on page 22.

Click OK and rebuild the project to get the BIN file generated.

### Store Cortex-M4 BIN Image File on SD Card

The SD Card has two partitions:

- The Linux file system partition.
- The **FAT32** boot partition.

List them using the fdisk command:

```
~# fdisk -1
...
Device Boot Start End Sectors Size Id Type
/dev/mmcblk0p1 8192 24575 16384 8M c W95 FAT32 (LBA)
/dev/mmcblk0p2 24576 1236991 1212416 592M 83 Linux
```

To execute the binary Cortex-M4 image at system startup, it must be stored in the **FAT32** boot partition using the following steps:

1. Create a sub-directory on the Linux file system, for example:

```
~# mkdir /media/sd0
```

2. Mount the Linux file system partition for access with RSE.

```
~# mount -t vfat /dev/mmcblk0p1 /media/sd0
```

- 3. Use RSE to copy the binary image file from your workspace to the /media/sd0 directory.
- 4. Unmount the partition to ensure that the file is written correctly:

~# umount /media/sd0

5. Reboot the system and halt in U-Boot.

#### Run Cortex-M4 BIN Image File from U-Boot

At this point, the Cortex-M4 BIN image file is stored in the boot partition.

Use the setenv command at the U-Boot prompt to change the U-Boot setup the new image file:

```
=> setenv m4image Blinky.bin; save
```

Verify the boot setup using the printenv command:

```
=> printenv
...
loadm4image=fatload mmc ${mmcdev}:${mmcpart} 0x7F8000 ${m4image}
m4boot=run loadm4image; bootaux 0x7F8000
m4image=Blinky.bin
```

Run the m4boot process to start the Blinky application:

=> run m4boot

#### NOTE

For more information refer to the U-Boot Command Line Interface in the U-Boot user's manual (<u>www.denx.de/wiki/DULG/UBoot</u>).

# Index

# **A**

Applications	
Add Source Code	31
Blinky with CMSIS-RTOS RTX	25
Build	33
Build M4 Image	33
Create	25
Create BIN File	47
Create Source Files	30
Customize RTOS	29
Debug	36
Run from U-Boot	49
Select Software Components	28
Setup the Project	26
Store BIN File	48

# **C**

Console
Console

#### D

Debug	
OS Awareness	41
Device Database	11
Documentation	17
DS-MDK	
Install	9
Installation Requirements	9
Introduction	
License Types	8
Licensing	8
-	

### Ε

Eclipse	
IDE	
Perspectives	
Workbench	
Example Project	
Install	14
Workbench Example Project Install	

# F

Linux	
Create Image	12
Install Image	12
Linux Applications	
Build Application Image	
Development	
Project Set Up	

#### Ρ

Perspective	
C/C++	19
CMSIS Pack Manager	
DS-5 Debug	
Remote System Explorer	

### R

Remote	System	Explorer	43
	~	1	

#### S

Software Packs		
Install manually	11	l
Manage	11	l

### Т

Terminal	View		37	7
----------	------	--	----	---