

## Freescale Semiconductor Application Note

Document Number: AN4948

# Flash Programmer for CodeWarrior Power Architecture

## 1. Introduction

This application note describes the steps required to program a flash device using the CodeWarrior for Power Architecture.

This document includes the following section:

- Import or create a flash programmer target task
- Configure a flash programmer target task
- Add flash programmer actions on a NOR/NAND/SPI flash device
- Execute a flash programmer target task
- Program a flash device

## 2. Preliminary background

The CodeWarrior flash programmer allows you to program flash memory device of the supported target boards from the CodeWarrior IDE. The flash programmer runs as a target task in the CodeWarrior IDE.

#### Contents

1.	Int	roc	ducti	ion	 	1	
~	-						

- Preliminary background.....1
   Configuring a flash device.....2
- 4. Programming a flash device......9



© Freescale Semiconductor, Inc., 2014 All rights reserved.

2 Configuring a flach day

# 3. Configuring a flash device

To program a flash device using the CodeWarrior Flash Programmer, use the **Target Tasks** view. To open the **Target Tasks** view, choose **Window** > **Show View** > **Target Tasks** from the CodeWarrior IDE menu.

#### Figure 1. Target Tasks view

🖳 Console 🧟 Tasks 🚺 Memory 🔏 Remote Sy	/stems 🔞 Target Tasks 🕄	Problems	Executables		- 6
			4	😑 🛈 🗶 💥 🗋 û	. 다   관 🗸 🎽
Arrange By:Task Groups 🔻	œ e	Tasks			Q
🥟 Root		Name	Task Type	Run Configuration	

Now, you can use the Target Tasks view to:

- Import a target task or Create a target task
- <u>Add flash programmer actions</u>
- Execute flash programmer target task

## 3.1. Import a target task

If you are using a Freescale board, you can use the flash programmer by importing an existing target task file, corresponding to the board type and flash device type (NAND, NOR, or SPI).

To import a target task file, perform the following steps:

- 1. From **Target Tasks** view toolbar, choose **Import** (2) and browse to <CWInstallDir>\PA\bin\plugins\support\TargetTask\Flash\_Programmer
- 2. Select the target task file corresponding to the board type and flash device type.

#### Figure 2. Importing a target task

🖳 Console 🕢 Tasks 🚺 Memory 🗐 Remot	e Systems 🔞 Target Tasks 😣 🔪	🛃 Problems 💽 Executables		- 6
		+ 🕞 (	🕽 🗯 🎇 📄 순 🖓	23
Arrange By:Task Groups 🔻 🖽	📄 Tasks			0
Proof	Name	Task Type	Run Configuration	
	P1010RDB_NAND_FLASH	Flash Programmer for Power Architecture	Active Debug Context	]

3. Double-click the imported target task in the **Target Tasks** view to display the target task details in a new window (see Figure 3).



#### Figure 3. Target task details

lash Devices			Target RAM
Device Name		Base Address	Address: 0x 00000000
MT29F16G08ABAC	A-IFC (2048Mx8x1)	0x0000000	Size: 0x 00040000
			Verify Target Memory Writes
add Device Rem ash Programmer A	ove Device ctions		
Add Device Rem ash Programmer A Enabled Operati	ove Device ctions n Description		Add Action 🔻
Add Device Rem ash Programmer J Enabled Operati	ove Device ctions n Description		Add Action
Add Device Rem Iash Programmer / Enabled Operati	ove Device ctions n Description		Add Action Duplicate Action Remove Action

In the figure above, you can notice that a flash device is added to the imported target task. You can add flash programmer actions to the flash device. See <u>Add flash programmer actions</u> to add flash programmer actions.

### 3.2. Create a target task

When a custom board is used, it is recommended to create a new target task.

1. In the **Target Tasks** view, click **Create New Target Task** (]]), as shown in <u>Figure 4</u>. The **New Target Task** dialog appears (see <u>Figure 5</u>).

#### Figure 4. Creating a target task

		4	🔁 🖸 🗰 🖄 🗎 👌	· · · ·
Arrange By:Task Groups 🔻	Tasks			4
➢ Root	 Name	Task Type	Run Configuration	

- 2. In the New Target Task dialog, enter a target task name in the Task Name text box, choose a launch configuration from the Run Configuration menu, and choose Flash Programmer for Power Architecture from the Task Type menu.
  - Choose Active Debug Context as your launch configuration when flash programmer is used over an active debug session.
  - Choose a project-specific debug context as your launch configuration when flash programmer is used without an active debug session.



Configuring a flash device

#### Figure 5. New Target Task dialog

Tack Name	
Task Group Root Browse	
Run Configuration Active Debug Context 👻	
Task Type         Flash Programmer for Power Architecture         +	

- 3. Click **OK**. The **New Target Task** dialog closes and a new target task is created in the **Target Tasks** view.
- 4. Double-click the newly created target task in the **Target Tasks** view. A new window similar to one shown in <u>Figure 3</u> displays, where you can configure the new target task.
- 5. Click **Add Device** to add a flash device to the target task. The **Add Device** dialog appears (see <u>Figure 6</u>).
- 6. Specify the name and organization of the flash device you want to add to the target task and click **Add Device**, as shown in Figure 6.

#### Figure 6. Add Device dialog

Name	Organization	
MT29F16G08ABACA-IFC		-
MT29F16G08ABACA-eLBC	2048Mx8x1	
MT29F1G08-IFC	128Mx8x1	
MT29F1G08-eLBC	128Mx8x1	
MT29F2G08AAB-eLBC	256Mx8x1	-
MT29F2G08AAC-eLBC	256Mx8x1	
MT29F2G08ABAEA-eLBC	256Mx8x1	
MT29F2G08ABBEA-eLBC	256Mx8x1	
MT29F2G08ABC-eLBC	256Mx8x1	-
MT29F4G08AAA-IFC	512Mx8x1	
MT29F4G08AAA-eLBC	512Mx8x1	
MT29F4G08ABADA-IFC	512Mx8x1	
MT29F4G08ABADA-eLBC	512Mx8x1	
MT29F4G08ABBDA-IFC	512Mx8x1	
MT29F4G08ABBDA-eLBC	512Mx8x1	
MT29F4G08BAC-IFC	512Mx8x1	
MT29F4G08BAC-eLBC	512Mx8x1	
MT29F4G08BAC-IFC	512Mx8x1	

7. Click **Done**. The **Add Device** dialog closes and the specified flash device is added in the target task details window (see Figure 7).



8. Specify a base address for the flash device, as shown in Figure 7.

#### Figure 7. Specifying base address for a flash device

lash Devid	es			Target RAM
Device Na	me		Base Address	Address: 0x 0
MT29F16	G08ABACA-e	LBC (2048Mx8x1)	0x0000000	Size: 0x 0
				Verify Target Memory Writes
Add Devid	Remove	Device		
Add Devid ash Prog Enabled	re Remove rammer Actio Operation	Device ns Description		Add Action 🔻
add Devid ash Prog Enabled	re Remove rammer Actic Operation	Device Ins Description		Add Action
Add Devid ash Prog Enabled	Remove rammer Actic Operation	Device ns Description		Add Action Duplicate Action Remove Action
Add Devid Iash Prog Enabled	Remove rammer Actic Operation	Device ns Description		Add Action

- **NOTE** For NOR flashes, base address indicates a location where the flash is mapped in the memory. For NAND and SPI flashes, the base address is usually  $0 \times 0$ .
- 9. Specify target RAM settings for the flash device, as shown in <u>Figure 8</u>. The target RAM is used by the flash programmer to download its algorithm.

#### Figure 8. Specifying target RAM for a flash device

lash Devices			Target RAM
Device Name		Base Address	Address: 0x 0000000
MT29F16G08	ABACA-eLBC (2048Mx8x1)	0x0000000	Size: 0x 00040000
			Verify Target Memory Writes
Add Device	Remove Device		
Add Device	Remove Device mer Actions		
Add Device lash Program Enabled O	Remove Device mer Actions peration Description		Add Action
Add Device lash Program Enabled O	Remove Device mer Actions peration Description		Add Action   Duplicate Action
Add Device	Remove Device mer Actions peration Description		Add Action Duplicate Action Remove Action
Add Device	Remove Device mer Actions Description		Add Action Duplicate Action Remove Action Move Up

Flash Programmer for CodeWarrior Power Architecture Application Note

Freescale Semiconductor, Inc.



**NOTE** The target RAM memory area is not restored by the flash programmer. If the flash programmer is used with an active debug context, it impacts the debug session.

## 3.3. Add flash programmer actions

Following actions can be added on a flash device:

- Erase/blank check
- Program/verify
- Checksum
- Diagnostics
- Dump flash
- Protect/unprotect
- Secure/unsecure

For details on each of these actions, see *CodeWarrior Development Studio for Power Architecture*® *Processors Version 10.x Tracing and Analysis Tools User Guide*, available in the <CWInstallDir>\PA\Help\PDF folder.

Follow these steps to add a flash programmer action on a flash device:

1. Choose **Program / Verify** from the **Add Action** menu in the target task details window, as shown in Figure 9. The **Add Program / Verify Action** dialog appears (see Figure 10).

Figure 9. Adding a flash programmer action on a flash device

Device Name     Base Address     Address: 0x     0000000       MT29F16G08ABACA-eLBC (2048Mx8x1)     0x0000000     Size: 0x     00040000       Size: 0x     00040000     Verify Target Memory Writes	Idsil Devi	ces			Target RAM	
lash Programmer Actions Enabled Operation Description Add Action 🔻	Device N MT29F16 Add Devi	iame iG08ABACA-e ce Remove	LBC (2048Mx8x1) Device	Base Address A	Address: 0x 0000000 iize: 0x 00040000 Verify Target Memory Writes	m
Erase / Blank Check Program / Verify Checksum	lash Pro <u>c</u> Enabled	operation	ns Description		Add Action 👻 Erase / Blank Check Program / Verify	

2. Select the complete path and type of the file, to be written to the flash device. If a separate erase action was not used before using the program/verify action, then select the **Erase sectors before program** option. Also, ensure that the correct address offset is specified, so that the image is programmed into the correct location.



Figure 10. Add Program / Verify Action dialog

ash Devices		Use File from Launch Configuration	
Device Name	Base Address	File: David La Marta II. 1010 II.	
MT29F16G08ABACA-eLBC (2048Mx	0x00000000	File: D:\tttpboot\1010rdb\p1010rdb_	uboot_nand.bin
		File Type: Auto 🔻	Workspace File System Variables
		Erase sectors before program     Restrict to Addresses in this Range	Verify after progr
		Start: 0x 00000000	Address: 0x 00000000
		End: 0x 7FFFFFF	

**NOTE** The **Verify after program** option is available, but only if the processor supports it.

3. Select Add Program Action, to add the action, and then select Done to close the Add Program / Verify Action dialog.

### 3.4. Execute flash programmer target task

Once you have finished configuring your flash programmer target task, you can execute the target task. To execute a flash programmer target task, perform the following steps:

1. Select the target task in the **Target Tasks** view and click **Execute** (**O**), as shown in <u>Figure 11</u>.

#### Figure 11. Executing target task

	 	+ 6	》 《 《 [] 公 · · · ]	èъ
Arrange By:Task Groups 💌	Tasks			
🕞 Root	 Name	Task Type	Run Configuration	
	P1010RDB_NAND	Flash Programmer for Power Architecture	Active Debug Context	

2. If prompted, select **Yes** in the **Save Resource** dialog, to ensure that changes are saved before executing the target task.

Freescale Semiconductor, Inc.



Configuring a flash device

#### Figure 12. Save Resource dialog



The flash programmer task starts executing, as shown in Figure 13.

#### Figure 13. Executing task dialog

Executing task P1010RDB_NA	ND		
Launching Debugger E	ngine		
🕅 Always run in background			
	Run in Background	Cancel	Details >>
	10		

You can check the results of flash actions in the **Console** view (see Figure 14). The green color indicates success of the task and the red color indicates failure of the task.

#### Figure 14. Console view

😑 Console 🛛 🖉 Tasks 🟮 Memory 🚚 Remote Systems 🔞 Target Tasks 🐰 Problems 🚺 Executables	🔓 🚰 🛛 🛫 🗳 🕶 🗖
Flash Programmer Console	
Erasing Erase Command Succeeded cmdwin::fl::write Beginning Operation	
Programming file D:\tftpboot\1010rdb\p1010rdb_uboot_nand.bin Auto-detection is successful. File is of type Binary/Raw Format.	
Downloading Flash Device Driver Reading flash ID	
Auto-detection is successful. File is of type Binary/Raw Format.	
Downloading 0x00010000 bytes to be programmed at 0x00000000	
Downloading 0x00010000 bytes to be programmed at 0x00010000	
Downloading 0x00010000 bytes to be programmed at 0x00020000	
Executing program Program Command Succeeded	
Downloading 0x00010000 bytes to be programmed at 0x00030000	
Downloading 0x00010000 bytes to be programmed at 0x00040000	
Downloading 0x00010000 bytes to be programmed at 0x00050000	
Executing program Program Command Succeeded	E
Downloading 0x00010000 bytes to be programmed at 0x00060000	
Downloading Avaaalaaa huter to be programmed at Avaaalaaaa	,
·	•



## 4. Programming a flash device

During bring-up and board support procedures, programming using flash programmer usually fails. The reason is that the DDR memory is not initialized or configured incorrectly. Because each new processor has an internal memory, you can use an SRAM initialization file with **Connect** a target and flash programmer, to program a flash devices.

To program a flash device, perform the following steps:

- 1. Choose **Run** > **Debug Configurations** from the CodeWarrior IDE menu bar. The **Debug Configurations** dialog appears.
- 2. Create a new debug configuration or edit an existing configuration, select **Connect** as your debug session type, as shown in Figure 15.

Figure 15. Selecting debug session type in Debug Configurations dialog

Debug session type Choose a predefined	debug session type or custom type for maximum flexibility	
<ul> <li>Download</li> <li>Attach</li> </ul>	<ul> <li>Connect</li> <li>Custom</li> </ul>	
▼ C/C++ application		
Project:	p1010rdb-core00	Browse
Application:	RAM/p1010rdb-core00.elf Search Project Browse	e Variables
Build (if required)	before launching	
<ul> <li>Target settings</li> </ul>		
Connection:	=== p1010rdb-core00_RAM_P1010_Connect T Edit	. New
Execute reset seque	ence	
- Everute initialization	on script(s)	

- 3. Click Edit next to the Connection menu. The Properties window appears (see Figure 16).
- 4. Browse to the SRAM initialization file located inside the <CWInstallDir>\PA\PA\_Support\Initialization\_Files folder, as shown in Figure <u>16</u>.



Programming a flash device

Figure 16. Properties wind	dow
----------------------------	-----

р	arent profile:				
		811883-01			
N	lame:	p1010rdb-core00_RAM_P1010_Connect Target			
D	escription:				
т	emplate:	None 👻	Apply Defau		
т	arget type:	P1010 T	Edit		
	Initialization       Memory       Advanced         Imitialization       Execute reset         Imitialize target:       D:Freescale\CW_PA_v10.3.4 140227\PA\PA_Support\Initialization_Files\QorlQ_P1\P1010RDB_init_sram_flash.tcl				

5. Open a debug session using the above settings and use the flash programmer to program the flash device.





How to Reach Us:

Home Page: www.freescale.com

E-mail: support@freescale.com Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Freescale, the Freescale logo, AltiVec, C-5, CodeTest, CodeWarrior, ColdFire, ColdFire+, C-Ware, Energy Efficient Solutions logo, Kinetis, mobileGT, PowerQUICC, Processor Expert, QorlQ, Qorivva, StarCore, Symphony, and VortiQa are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. Airfast, BeeKit, BeeStack, CoreNet, Flexis, Layerscape, MagniV, MXC, Platform in a Package, QorlQ Qonverge, QUICC Engine, Ready Play, SafeAssure, SafeAssure logo, SMARTMOS, Tower, TurboLink, Vybrid, and Xtrinsic are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2014.

Document Number: AN4948 5 May 2014

