Freescale Semiconductor
Application Note

Document Number: AN5048

Using an External GCC Toolchain with CodeWarrior for ARMv7

1. Introduction

This document explains how to use an external GNU compiler collection (GCC) toolchain with CodeWarrior for QorIQ LS series – ARM V7 ISA. This process is only applicable to the Linux version of CodeWarrior.

This document provides steps to:

- Build the toolchain supplied with Freescale Linux SDK
- Customize a stationary Linux project to work with an SDK standalone toolchain
- Build the project using an external toolchain

Contents

2. Preliminary background

CodeWarrior for QorIQ LS series - ARM V7 ISA includes the GCC Linaro binary toolchain. If you are developing a Linux user space application with CodeWarrior, then you are recommended to use the toolchain supplied with the Freescale Linux SDK.

3. Using SDK standalone toolchain

You can use the standalone toolchain provided in SDK to build a Linux user space application with CodeWarrior. To build and install the standalone toolchain with Yocto, perform these steps:

```
$ cd build_<machine>_release
$ bitbake fsl-toolchain
$ cd build_<machine>_release/tmp/deploy/sdk
$ ./fsl-networking-eglibc-<host-system>-<core>-toolchain-
<release>.sh
```

NOTE The default installation path for the standalone toolchain is: `/opt/fsl-networking/`. You need to specify this path while installing the standalone toolchain.
For additional information about building and installing the standalone toolchain with Yocto, see **SDK Knowledge Center**.

See Change toolchain for using SDK standalone toolchain as the default build tool in CodeWarrior.

4. Switching between different CodeWarrior toolchains

CodeWarior toolchain versions are provided as service packs. This helps you to choose the desired toolchain version.

To switch from Linaro GCC toolchain 4.9.3 to 4.8.3, follow these steps:

1. Choose **Help > About CodeWarrior Development Studio** from the CodeWarrior IDE menu bar. The **About CodeWarrior Development Studio** dialog appears.
2. Click **Installation Details**. The **CodeWarrior Development Studio Installation Details** dialog appears.
3. Select **GNU ARM C/C++ Development Support**, **GNU ARM LinuxApp C/C++ Development Support, Linaro GCC Baremetal**, and **Linaro GCC Linux Application** on the **Installed Software** page and click **Uninstall**.
4. Restart CodeWarrior as indicated by the installer.
5. Choose **Help > Install New Software** from the CodeWarrior IDE menu bar and select the service pack corresponding to **Linaro GCC toolchain 4.8.3**.
6. Install the new service pack.

7. Restart CodeWarrior as indicated by the installer.

To switch back to the original Linaro GCC toolchain 4.9.3, follow these steps:

1. Choose **Help > About CodeWarrior Development Studio** from the CodeWarrior IDE menu bar. The **About CodeWarrior Development Studio** dialog appears.
2. Click **Installation Details**. The **CodeWarrior Development Studio Installation Details** dialog appears.
3. Select **GNU ARM C/C++ Development Support**, **GNU ARM LinuxApp C/C++ Development Support**, and **Wizard feature for GCC** on the **Installed Software** page and click **Uninstall**.
4. Restart CodeWarrior as indicated by the installer.
5. Go to **Help > Install New Software** and reinstall the service pack corresponding to **Linaro GCC toolchain 4.9.3**.

NOTE Due to the differences between toolchains 4.8.3 and 4.9.3, when a project created using toolchain 4.8.3 service pack is built using toolchain 4.9.3 service pack, the correct architecture needs to be chosen from the **Architecture** list on the **Project Properties > C/C++ Build > Settings > Tool Settings > Target Processor** page.

In addition, when a project created using toolchain 4.9.3 service pack is built using toolchain 4.8.3 service pack, the **rdimon** library needs to be added to the **Libraries** pane on the **Project Properties > C/C++ Build > Settings > Tool Settings > ARM Sourcery GCC C Linker > Libraries** page and the **--specs=rdimon.specs** flag needs to be added to the **Other flags** field on the **Project Properties > C/C++ Build > Settings > Tool Settings > ARM Sourcery GCC C Linker > Miscellaneous** page.
Changes in other settings may be required depending on the processor type or compiler options.

5. Working with an ARMv7 Linux application project

This section contains the following subsections:

- Create a stationary project for Linux application
- Change toolchain
- Verify build settings
- Build project using an external toolchain

5.1. Create a stationary project for Linux application

To create an ARMv7 stationary project for Linux application, follow these steps:

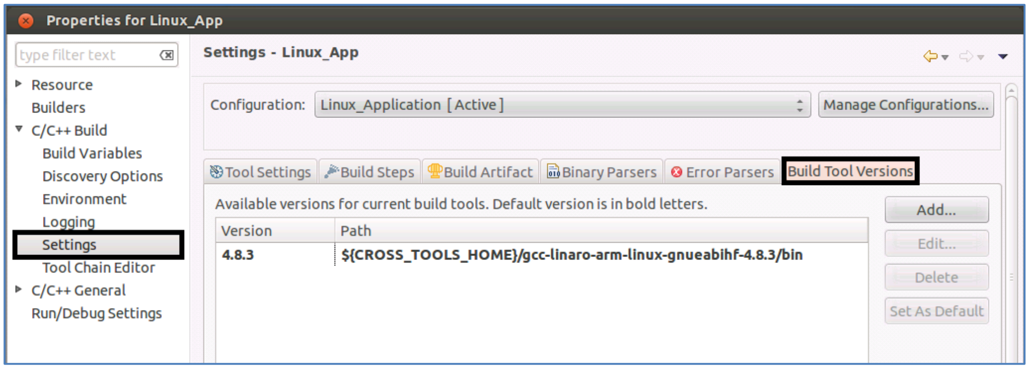
1. Start CodeWarrior for QorIQ LS series - ARM V7 ISA.
2. Choose **File > New > CodeWarrior Linux Project Wizard** from the CodeWarrior IDE menu bar. **CodeWarrior Linux Project Wizard** starts.
3. Specify the project name and location.
4. Select the processor and project output.
5. Configure the build settings.
6. Configure the connection details and click **Finish** to create the Linux application project.

5.2. Change toolchain

By default, the stationary project for Linux application includes the GCC Linaro binary toolchain. To change the default toolchain used by CodeWarrior, follow these steps:

1. Choose **Project > Properties** from the CodeWarrior IDE menu bar. The **Properties** dialog appears.
2. Choose **C/C++ Build > Settings** in the left pane and click the **Build Tool Versions** tab in the right pane, as shown in the figure below.

Figure 1. Project properties



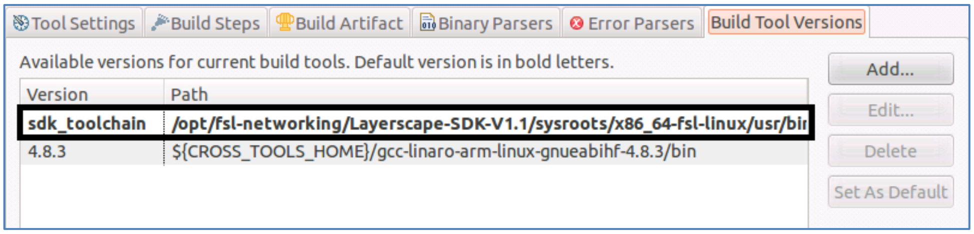
Configuration: Linux_Application [Active]

Available versions for current build tools. Default version is in bold letters.

Version	Path
4.8.3	**${CROSS_TOOLS_HOME}/gcc-linaro-arm-linux-gnueabihf-4.8.3/bin**

3. Click **Add** and browse for the new toolchain location. The default installation path for Freescale Linux SDK standalone toolchain is: `/opt/fsl-networking/Layerscape-<release>/sysroot/<host-system>/usr/bin/arm-fsl-linux-gnueabi/`
4. Click **OK** to make the new toolchain available.
5. Select the new toolchain and click **Set As Default**, as shown in the figure below. This will make the new toolchain as the default toolchain for the project.

Figure 2. Change default toolchain



Available versions for current build tools. Default version is in bold letters.

Version	Path
sdk_toolchain	**/opt/fsl-networking/Layerscape-SDK-V1.1/sysroots/x86_64-fsl-linux/usr/bi**
4.8.3	${CROSS_TOOLS_HOME}/gcc-linaro-arm-linux-gnueabihf-4.8.3/bin

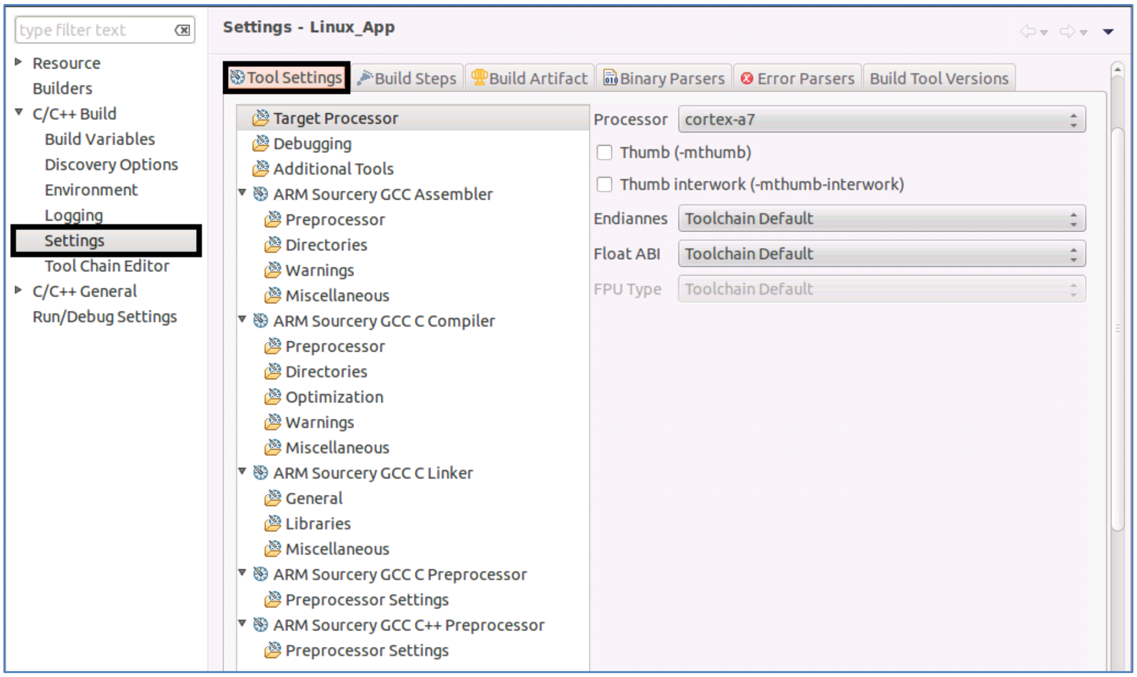
NOTE In the CodeWarrior for QorIQ LS series - ARM V7 ISA 10.0.3 release, the **Add** option is not available and you need to implement a workaround to use an external toolchain as the default toolchain. See Changing toolchain path for details.

5.3. Verify build settings

After setting the external toolchain as the default toolchain and before building your project, you should verify the build settings of the project. To verify build settings, follow these steps:

1. Choose **Project** > **Properties** from the CodeWarrior IDE menu bar. The **Properties** dialog appears.
2. Choose **C/C++ Build** > **Settings** in the left pane and click the **Tool Settings** tab in the right pane, as shown in the figure below.

Figure 3. Tool settings



Processor: cortex-a7

□ Thumb (-mthumb)

□ Thumb interwork (-mthumb-interwork)

Endiannes: Toolchain Default

Float ABI: Toolchain Default

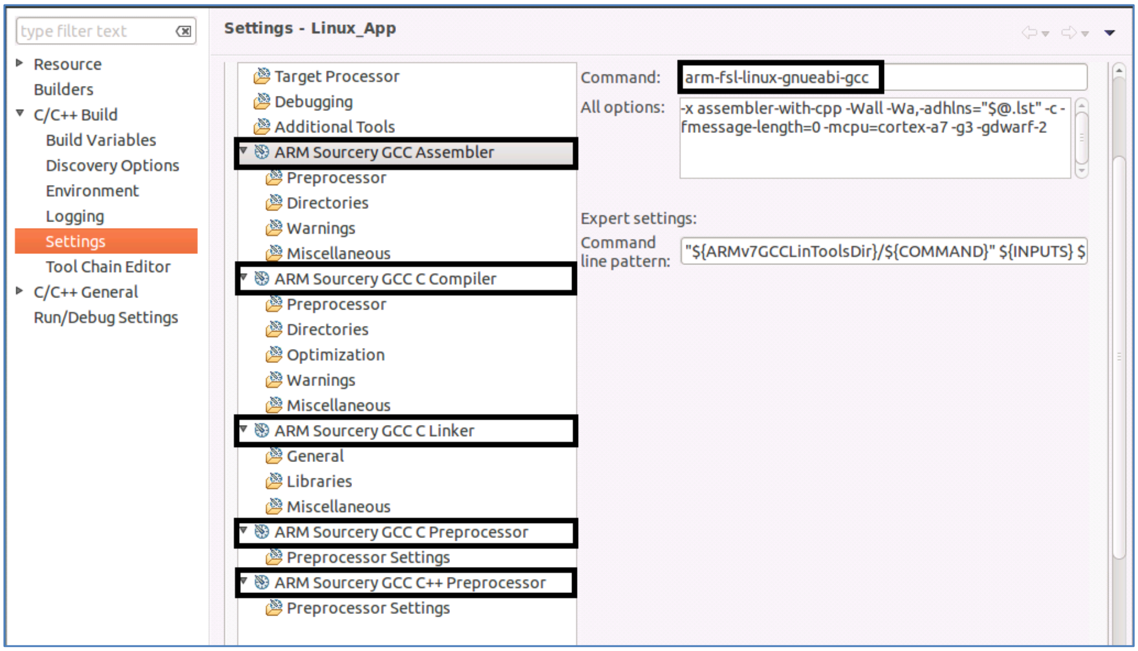
FPU Type: Toolchain Default

3. For ARM Sourcery GCC Assembler, C Compiler, C Linker, C Preprocessor, C++ Preprocessor,

verify if the command is same as in the external toolchain (see the figure below).

NOTE In the toolchain that is shipped with CodeWarrior, the command is: *`arm-linux-gnueabihf-`*, and in the Freescale SDK Linux toolchain, the command is: *`arm-fsl-linux-gnueabi-`*.

Figure 4. Project settings



Command: arm-fsl-linux-gnueabi-gcc

All options: -x assembler-with-cpp -Wall -Wa,-adhlns="$@.lst" -c -fmessage-length=0 -mcpu=cortex-a7 -g3 -gdwarf-2

Expert settings:

Command line pattern: "${ARMv7GCCLinToolsDir}/${COMMAND}" ${INPUTS} $

4. Click **OK** to save the project settings.

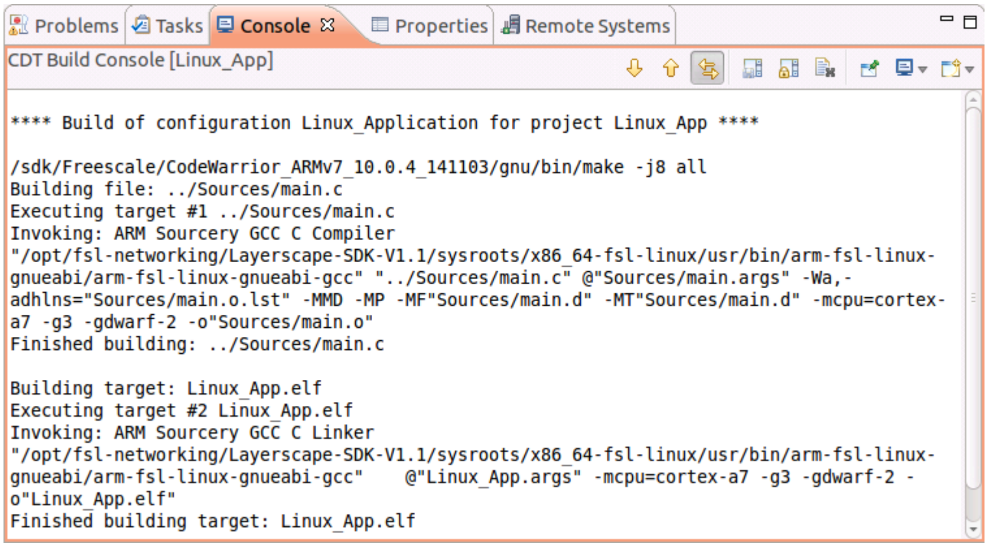
SDK toolchain is a sysrooted toolchain. This means that GCC will start to look for target fragments and libraries starting from the path specified by the sysroot option. To have a working build configuration, follow these steps:

1. For ARM Sourcery GCC C Compiler, go to **Miscellaneous** > **Other flags** and add *`--sysroot=''<path_to_target_sysroot>''`* as an option.
2. For ARM Sourcery GCC C Linker, go to **Miscellaneous** > **Other flags** and add *`--sysroot=''<path_to_target_sysroot>''`* as an option.

5.4. Build project using an external toolchain

To build the project, choose **Project** > **Build Project** from the CodeWarrior IDE menu bar. The project should be built with no errors, as shown in the figure below.

Figure 5. Console view



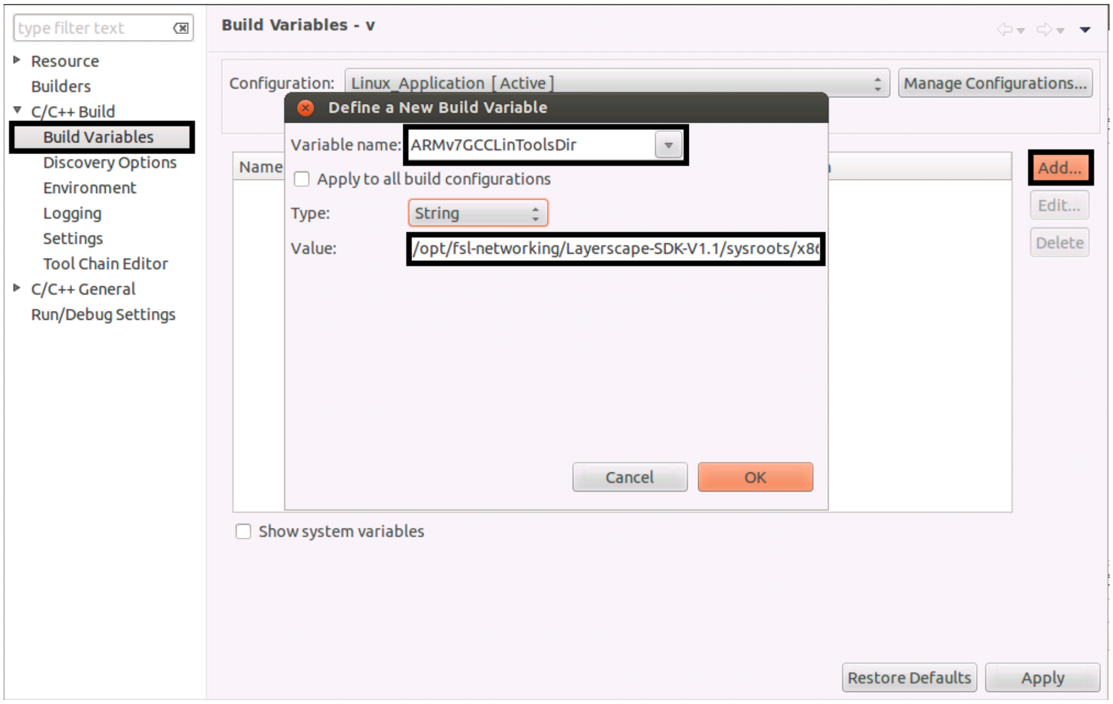
```
**** Build of configuration Linux_Application for project Linux_App ****

/sdk/Freescale/CodeWarrior_ARMv7_10.0.4_141103/gnu/bin/make -j8 all
Building file: ../Sources/main.c
Executing target #1 ../Sources/main.c
Invoking: ARM Sourcery GCC C Compiler
"/opt/fsl-networking/Layerscape-SDK-V1.1/sysroots/x86_64-fsl-linux/usr/bin/arm-fsl-linux-
gnueabi/arm-fsl-linux-gnueabi-gcc" "../Sources/main.c" @"Sources/main.args" -Wa,-
adhlns="Sources/main.o.lst" -MMD -MP -MF"Sources/main.d" -MT"Sources/main.d" -mcpu=cortex-
a7 -g3 -gdwarf-2 -o"Sources/main.o"
Finished building: ../Sources/main.c

Building target: Linux_App.elf
Executing target #2 Linux_App.elf
Invoking: ARM Sourcery GCC C Linker
"/opt/fsl-networking/Layerscape-SDK-V1.1/sysroots/x86_64-fsl-linux/usr/bin/arm-fsl-linux-
gnueabi/arm-fsl-linux-gnueabi-gcc"    @"Linux_App.args" -mcpu=cortex-a7 -g3 -gdwarf-2 -
o"Linux_App.elf"
Finished building target: Linux_App.elf
```

6. Changing toolchain path

To specify the external toolchain path as the path for the *ARMv7GCCToolsDir* and *ARMv7GCCLinToolsDir* variables, follow these steps:

1. Choose **Project > Properties** from the CodeWarrior IDE menu bar. The **Properties** dialog appears.
2. Choose **C/C++ Build > Build Variables** in the left pane and click **Add** in the right pane. The **Define a New Build Variable** dialog appears.
3. Specify external toolchain path as the value (path) for each of the variables, *ARMv7GCCToolsDir* and *ARMv7GCCLinToolsDir*, as shown in the figure below.

Figure 6. Change toolchain path



Configuration: Linux_Application [Active]

Variable name: ARMv7GCCLinToolsDir

Type: String

Value: /opt/fsl-networking/Layerscape-SDK-V1.1/sysroots/x8

4. Click **OK**.
5. Click **OK**.

How to Reach Us:

Home Page:
freescale.com

E-mail:
freescale.com/support

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, CodeWarrior, and QorIQ are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners. ARM, Cortex, and TrustZone are trademarks or registered trademarks of ARM Ltd or its subsidiaries in the EU and/or elsewhere. All rights reserved.

Document Number: AN5048

15 January 2016