NXP Semiconductors
Release Notes

Document Number: MCUXSDKIMX6ULRN
Rev. 0, 06/2017

MCUXpresso SDK Release Notes for i.MX 6UltraLite Devices

Contents

1 Overview

The MCUXpresso Software Development Kit is a collection of software enablement for Microcontrollers that includes peripheral drivers, high level stacks including USB and other middleware packages, such as FatFs, and integrated RTOS support for FreeRTOS™ OS. In addition to the base enablement, the MCUXpresso SDK is augmented with demo applications and driver example projects, and API documentation to help the customers quickly leverage the support of the MCUXpresso SDK.

For the latest version of this and other MCUXpresso SDK documents, see the MCUXpresso SDK homepage MCUXpresso-SDK: Software Development Kit.

2 Development Tools

The MCUXpresso SDK was compiled and tested with these development tools:

- IAR Embedded Workbench for ARM version 7.80.4
- Makefiles support with GCC revision v5-2016-q3 from ARM Embedded

3 Supported Development Systems

This release supports boards and devices listed in this table. Boards and devices in boldface were tested in this release:

Table 1. Supported MCU devices and development boards

Development boards	i.MX devices
MCIMX6UL-CM, MCIMX6UL-BB	CPU_MCIMX6G0DVM05, CPU_MCIMX6G1AVM05, CPU_MCIMX6G1AVM07, CPU_MCIMX6G1CVM05, CPU_MCIMX6G2AVM05, CPU_MCIMX6G2AVM07, CPU_MCIMX6G2CVK05, CPU_MCIMX6G2CVM05, CPU_MCIMX6G2DVK05, CPU_MCIMX6G2DVM05, CPU_MCIMX6G3CVK05, CPU_MCIMX6G3CVM05, CPU_MCIMX6G3DVK05, **CPU_MCIMX6G3DVM05**

4 Release Contents

This table provides an overview of the MCUXpresso SDK release package contents and locations.

Table 2. Release contents

Deliverable	Location
Boards	<install_dir>/boards
Demo applications	<install_dir>/boards/<board_name>/demo_apps
USB demo applications	<install_dir>/boards/<board_name>/usb_examples
Driver examples	<install_dir>/boards/<board_name>/driver_examples
RTOS examples	<install_dir>/boards/<board_name>/rtos_examples
Documentation	<install_dir>/docs
USB Documentation	<install_dir>/docs/usb
Middleware	<install_dir>/middleware
SDMMC card driver	<install_dir>/middleware/sdmmc_<version>
USB stack	<install_dir>/middleware/usb_<version>
Driver, SoC header files, extension header files and feature header files, utilities	<install_dir>/devices/<device_name>
ARM® Cortex®-A core common header files	<install_dir>/CORTEXA
Peripheral Drivers	<install_dir>/devices/<device_name>/drivers
Utilities such as debug console	<install_dir>/devices/<device_name>/utilities
RTOS Kernel Code	<install_dir>/rtos
Tools	<install_dir>/tools

5 SDK Release Package

The MCUXpresso SDK release package contents are aligned with the silicon subfamily it supports. This includes the boards, CMSIS, devices, documentation, middleware, and RTOS support.

5.1 Device support

The device folder contains all available software enablement for the specific SoC subfamily. This folder includes clock-specific implementation, device register header file, device register feature header file, CMSIS derived device SVD, and the system configuration source files. Included with the standard SoC support are folders containing peripheral drivers, toolchain support, and a simple debug console.

The device-specific header files provide a direct access to the device peripheral registers. The device header file provides an overall System-on-Chip (SoC) memory mapped register definition. In addition to the overall device memory mapped header file, the MCUXpresso SDK also includes the feature header file for each peripheral instantiated on the SoC.

The toolchain folder contains the startup code and linker files for each supported toolchain. The startup code is a CMSIS-compliant startup that efficiently transfers the code execution to the main() function.

5.1.1 Board support

The boards folder provides the board-specific demo applications, driver examples, RTOS, and middleware examples.

5.1.2 Demo applications and other examples

The demo applications demonstrate the usage of the peripheral drivers to achieve a system level solution. Each demo application contains a readme file that describes the operation of the demo and required setup steps.

The driver examples demonstrate the capabilities of the peripheral drivers. Each example implements a common use case to help demonstrate the driver functionality.

The middleware folders each contain examples demonstrating the use of the included source.

5.2 Middleware

5.2.1 RTOS

The MCUXpresso SDK is integrated with FreeRTOS.

5.2.2 USB stack

See the *MCUXpresso SDK USB Stack User's Guide* (document MCUXSDKUSBSUG) for more information.

5.2.2.1 Peripheral devices tested with the USB Host stack

This table provides a list of USB devices tested with the USB Host stack.

Table 3. Peripheral devices

Device type	Device
USB HUB	BELKIN F5U233
	BELKIN F5U304
	BELKIN F5U307
	BELKIN F4U040
	UNITEK Y-2151
	Z-TEK ZK032A
	HYUNDAI HY-HB608
USB flash drive	ADATA C008 32 GB
	ADATA S102 8 G
	ADATA S102 16 G
	Verbatim STORE N GO USB Device 8 G
	Kingston DataTraveler DT101 G2
	SanDisk Cruzer Blade 8 GB
	Unisplendour 1 G
	Imation 2 GB
	V-mux 2 GB
	Sanmina-SCI 128 M
	Corporate Express 1 G
	TOSHIBA THUHYBS-008G 8 G
	Transcend JF700 8 G
	Netac U903 16 G
	SSK SFD205 8 GB
	Rex 4 GB
	SAMSUNG USB3.0 16GB
USB card reader/adapter	SSK TF adapter
	Kawau Multi Card Reader
	Kawau TF adapter
	Kawau SDHC card
USB Mouse	DELL MS111-P
	DELL M066U0A
	DELL MUAVDEL8

Table continues on the next page...

Table 3. Peripheral devices (continued)

	TARGUS AMU76AP DELL MD56U0 DELL MS111-T RAPOO M110
USB Keyboard	DELL SK8135 DELL SK8115

5.3 TCP/IP stack

The lwIP TCP/IP stack is pre-integrated with MCUXpresso SDK and runs on top of the MCUXpresso SDK Ethernet driver with Ethernet-capable devices/boards. For details, see the *lwIP TCPIP Stack and MCUXpresso SDK Integration User's Guide* (document MCUXSDKLWIPUG).

5.4 File System

The FatFs file system is integrated with MCUXpresso SDK and can be used to access either the SD card or the USB memory stick when the SD card driver or the USB Mass Storage Device class implementation is used.

For details, see the FatFs documentation installed at <install_dir>/middleware/fatfs_<version>/doc.

6 MISRA Compliance

All MCUXpresso SDK drivers and USB stack comply to MISRA 2004 rules with the following exceptions.

Exception Rules	Description
1.1	All code shall conform to ISO 9899:1990 Programming languages - C, amended and corrected by ISO/IEC 9899/COR1:1995, ISO/IEC 9899/AMD1:1995, and ISO/IEC
2.4	Sections of code should not be commented out.
5.1	Identifiers (internal and external) shall not rely on the significance of more than 31 characters.
6.3	typedefs that indicate size and signedness should be used in place of the basic types.
6.4	Bitfields shall only be defined to be of type unsigned int or signed int.
8.1	Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call.
8.5	There shall be no definitions of objects or functions in a header file.
8.1	All declarations and definitions of objects or functions at file scope shall have internal linkage unless external linkage is required.
8.12	When an array is declared with external linkage, its size shall be stated explicitly or defined implicitly by initialization.
10.1	The value of an expression of integer type shall not be implicitly converted to a different underlying type if: a. it is not a conversion to a wider integer type of the same signedness, or b. the expression is complex, or c. the expression is not constant and is a function argument, or d. the expression is not constant and is a return expression.
10.3	The value of a complex expression of integer type shall only be cast to a type that is not wider and of the same signedness as the underlying type of the expression.
11.3	A cast should not be performed between a pointer type and an integral type.
11.4	A cast should not be performed between a pointer to object type and a different pointer to object type.
11.5	A cast shall not be performed that removes any const or volatile qualification from the type addressed by a pointer.
12.2	The value of an expression shall be the same under any order of evaluation that the standard permits.
12.4	The right-hand operand of a logical && or | | operator shall not contain side effects.
12.6	The operands of logical operators (&&, | |, and !) should be effectively boolean. Expressions that are effectively boolean should not be used as operands to operators other than (&&, | |, !, =, ==, !=, and ?:).
12.13	The increment (++) and decrement (--) operators should not be mixed with other operators in an expression.
14.3	Before preprocessing, a null statement shall only occur on a line by itself; it may be followed by a comment, provided that the first character following the null statement is a whitespace character.
14.5	The continue statement shall not be used.
14.7	A function shall have a single point of exit at the end of the function.
16.1	Functions shall not be defined with a variable number of arguments.
17.4	Array indexing shall be the only allowed form of pointer arithmetic.
18.4	Unions shall not be used.
19.1	#include statements in a file should only be preceded by other preprocessor directives or comments.
19.1	In the definition of a function-like macro, each instance of a parameter shall be enclosed in parentheses unless it is used as the operand of # or ##.
20.4	Dynamic heap memory allocation shall not be used.
20.9	The input/output library <stdio.h> shall not be used in production code.

Figure 1. MISRA exceptions

7 Known Issues

7.1 Maximum file path length in Windows® 7 Operating System

Windows 7 operating system imposes a 260 character maximum length for file paths. When installing the MCUXpresso SDK, place it in a directory close to the root to prevent file paths from exceeding the maximum character length specified by the Windows operating system. The recommended location is the `C:\nxp` folder.

7.2 USB HUB power supply

The external power supply of the USB HUB must be provided before it can be used. This is the result of the development board which is not designed to power a USB HUB and the devices connected to the HUB. Therefore, the external USB HUB that is connected to the development board should have its own power supply.

7.3 IAR download issue

If QSPI flash already has a bootable application and boot mode is set to QSPI flash boot, or MicroSD card already has a bootable application and boot mode is set to SD boot, an uncertain IAR download error occurs. Therefore, if running an application using IAR use either of these workarounds:

- Set boot mode to reserved mode (BMODE[1:0]=2'b11).
- Run mfgtool2-erase-sdk-mx6ul-evk-qspi-nor-n25q256a.vbs to erase QSPI flash, or run mfgtool2-erase-sdk20-mx6ul-evk-sdcard.vbs to clear image vector table in MicroSD card before using IAR.

7.4 JTAG Port pin network

The JTAG port pin is reused by the Audio codec WM8960 SAI2 pin on the i.MX 6UltraLite and i.MX 6ULL EVK board. In order to correctly connect to the JTAG port of the board with the J-LINK debugger, five resistors, R1432, R1407, R1431, R1433, and R1434, need to be removed. This is shown in the following figure:

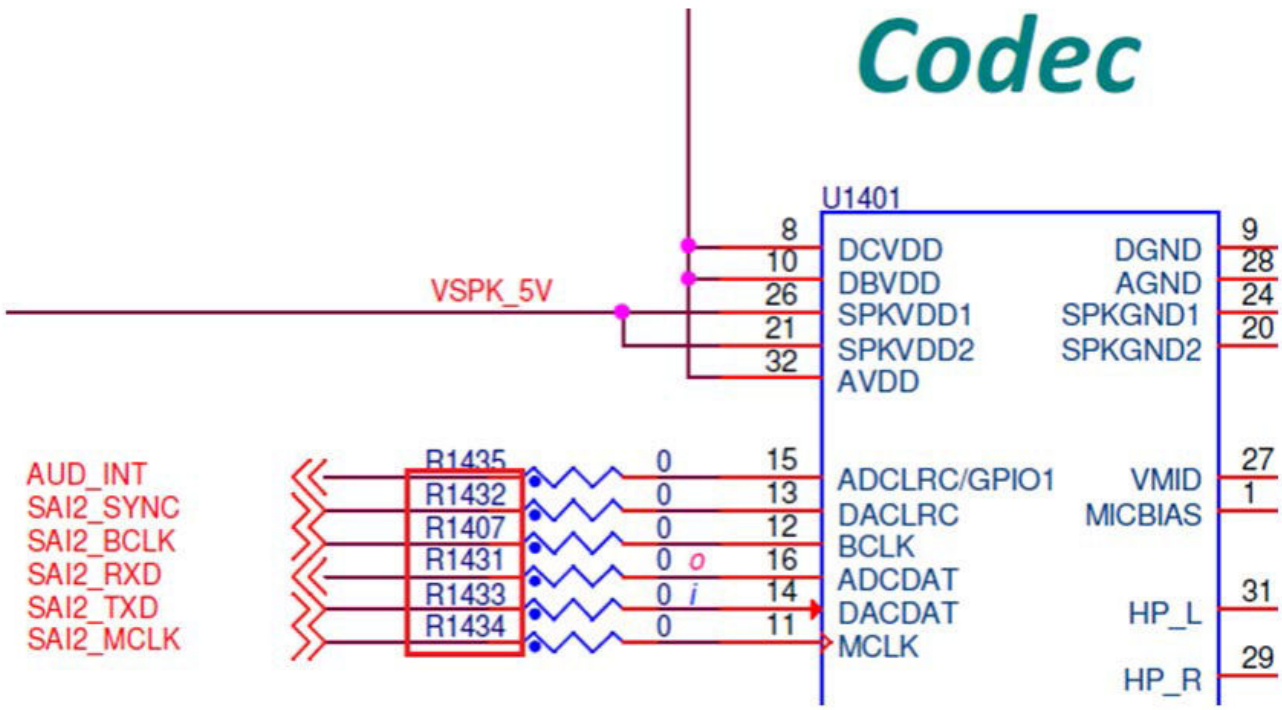

Figure 2. Removing resistors

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number MCUXSDKIMX6ULRN
Revision 0, 06/2017