NXP Semiconductors, Inc.
Release Notes

Document Number: FRTOS1016XRN
Rev. 0, 07/2016

FreeRTOS™ BSP v.1.0.1 for i.MX 6SoloX Release Notes

1 Overview

These are the release notes for the FreeRTOS™ BSP 1.0.1 for the i.MX 6SoloX ARM® Cortex®-M4 core. The core of the BSP is a set of peripheral drivers.

The peripheral driver wraps the hardware register accesses into a set of stateless functional primitives that provide the building blocks for high-level applications.

The FreeRTOS BSP includes a set of example applications demonstrating the use of the peripheral drivers and some demonstrations of integrated FreeRTOS Operating System (OS) v8.0.0. The FreeRTOS BSP also integrates RPMsg middleware to provide an easy-to-use multicore communication API for i.MX product families.

Contents

2 What Is New

These are the new features for FreeRTOS BSP 1.0.1 for i.MX 6SoloX:

- Bug Fixes:
  - Changed the LMEM (Local Memory Controller) base address from **0xE0002000** to **0xE0082000** in the SoC header file.
  - Corrected the misspellings in eCSPI, UART and FlexCAN drivers, and related demos/examples:
    - Changed the **ECSPPI_SetDMACmd()** function name to **ECSPI_SetDMACmd()**.
    - Changed the **flexCanNormalMode** member of **enum _flexcan_operating_modes** to **flexcanNormalMode**.
    - Changed **UAER_Putchar9() function name** to **UART_Putchar9()**.
    - Changed **UAER_Getchar9() function name** to **UART_Getchar9()**.
  - Solved the issue that the **FLEXCAN_ClearMsgBufStatusFlag()** function cannot correctly clean the status flags for **MB32–MB63**.
  - Fixed "w1c" status flag cleaning logic in drivers to prevent unexpected flag cleaning.
  - Solved the CAN transceiver wake-up issue on the i.MX 6SoloX SABRE-AI board.
- Improvements:
  - Added de-bounce logic to blinking demo and GPIO examples.
  - Added a new API **WDOG_GetResetSource()** to WDOG driver to get the reset source for the ARM Cortex-M4 core.
  - Disabled **Receive Nocopy Check** in the RPMsg Stack to reduce the memory usage of RPMsg demos.
  - Some code cleaning and documentation improvements.

3 Development Tools

The FreeRTOS BSP 1.0.1 was compiled and tested with these development tools:

- IAR Embedded Workbench for ARM® version 7.40.3
- Makefiles support with GCC revision 4.9-2015-q1-update from ARM Embedded
- ARM Development Studio 5 (DS-5™) version 5.20.1 (32 bit)
- Lauterbach TRACE32 PowerView for ARM version R.2014.09.000058270

4 Supported Development Systems

This release was tested with boards and devices listed in this table.

Table 2 Supported MCU devices and development boards

Development boards	i.MX devices
i.MX 6SoloX SABRE-SD board MCIMX6SX-SDB (Rev. B)	PCIMX6X4AVM08AB
i.MX 6SoloX SABRE-AI Automotive Infotainment board MCIMX6SXAICPU2 (Rev. A) SABRE Automotive Expansion board MCIMXABASEV1 Rev. E for peripherals	PCIMX6X4AVM08AB

NOTE

The release is only tested with the Linux® L3.14.52_1.1.0-GA release. It is not recommended to integrate it with other Linux BSP versions.

5 Release Contents

This table describes the release contents.

Table 3 Release contents

Deliverable	Location
Examples	<install_dir>/examples/...
Demo applications	<install_dir>/examples/<board_name>/demo_apps/…
Driver examples	<install_dir>/examples/<board_name>/driver_examples/...
Documentation	<install_dir>/doc/...
Middleware	<install_dir>/middleware/...
RPMsg stacks	<install_dir>/middleware/multicore/open-amp/...
Driver library, startup code and utilities	<install_dir>/platform/...
Cortex Microcontroller Software Interface Standard (CMSIS) ARM Cortex®-M header files, DSP library source	<install_dir>/platform/CMSIS/…
Linker control files for each supported tool chain	<install_dir>/platform/devices/<device_name>/linker/…
SoC header files	<install_dir>/platform/devices/<device_name>/include
CMSIS-compliant startup code	<install_dir>/platform/ devices/<device_name>/startup/…
Peripheral Drivers	<install_dir>/platform/drivers/…
Utilities such as debug console	<install_dir>/platform/utilities/…
RTOS Kernel folder	<install_dir>/rtos/...
FreeRTOS Kernel Code	<install_dir>/rtos/FreeRTOS/…
External tools for build and debug	<install_dir>/tools/…

6 FreeRTOS BSP Release Overview

The FreeRTOS BSP is intended for use with NXP's i.MX product family on the ARM Cortex-M4 core. The release consists of:

- i.MX 6SoloX ARM Cortex-M4 platform support (ARM Cortex®-A9 core is not supported)
- Demo applications/Driver examples
- Multicore communication stack (NXP modification of RPMsg, version 1.0.0)
- FreeRTOS kernel v8.0.0
- Documentation (*FreeRTOS BSP i.MX 6SoloX API Reference Manual* and various user's guides)

6.1 i.MX 6SoloX Cortex-M4 platform support

The FreeRTOS BSP platform directory contains the startup code, driver libraries for peripherals, header files, linker files, and utilities such as the debug console implementation and RDC settings.

6.1.1 Startup code

The FreeRTOS BSP includes simple CMSIS-compliant startup code for i.MX 6SoloX which efficiently delivers the code execution to the `main()` function. An application can include the startup code directly in the project build environment for a cleaner project build environment.

6.1.2 Driver library

The FreeRTOS BSP provides a set of drivers for the i.MX family on-chip peripherals. The drivers are designed and implemented around the peripheral hardware blocks rather than for a specific i.MX device, and work with or without FreeRTOS OS.

The drivers are designed to wrap hardware register accesses into functional accesses. They are stateless and are intended to cover the most frequently used hardware functionality. They are written in C language, which increases the reuse of code for other products as they are designed to be initialized at runtime based on the custom configuration depending on the product. For more details, see chapter "Architectural Overview" in the *FreeRTOS BSP i.MX 6SoloX API Reference Manual* (FRTOS6XAPIRM).

6.1.3 Header files

The FreeRTOS BSP devices directory contains device-specific header files that provide direct access to the i.MX peripheral registers. Each supported i.MX device in the FreeRTOS BSP has an overall System-on-Chip (SoC) memory-mapped header file. Along with the SoC header files, the FreeRTOS BSP CMSIS directory includes common CMSIS header files for the ARM Cortex-M core and the DSP library from the ARM CMSIS version 4.2 release.

6.1.4 Linker files

The FreeRTOS BSP devices directory contains linker control files (or simply linker files) for each supported tool chain and i.MX device.

6.1.5 Utilities

The utilities directory contains useful software utilities such as a debug console.

6.1.6 RDC settings

i.MX 6SoloX is a multicore device, and the Resource Domain Controller (RDC) can be used to grant peripheral and memory access permissions to each core.

The examples and demo applications in the FreeRTOS BSP use RDC to allocate peripheral access permission. When running the ARM Cortex-A9 application with the FreeRTOS BSP example/demo, it is important to respect the reserved peripheral. The FreeRTOS BSP application has reserved peripherals that are used only by ARM Cortex-M4, and any access from ARM Cortex-A9 core on those peripherals may cause the program to hang.

The default RDC settings are:

- The ARM Cortex-M4 core is assigned to RDC domain 1, and ARM Cortex-A9 core and other bus masters use the default assignment (RDC domain 0)
- Every example/demo has its specific RDC setting in its hardware_init.c. Most of them are set to exclusive access.

The user of this package can remove or change the RDC settings in the example/demo or in his application.

6.2 Demo applications

The demo applications demonstrate the usage of the drivers to create integrated software solutions on i.MX 6SoloX.

6.3 Driver examples

The driver examples demonstrate the usage of drivers on bare metal environment.

6.4 Other integrated software solutions

The FreeRTOS BSP is designed for easy integration with other software solutions such as FreeRTOS OS and RPMsg multicore communication.

6.4.1 RTOS

The FreeRTOS BSP is pre-integrated and tested with the FreeRTOS OS (www.freertos.org) v8.0.0.

As the drivers have no dependency on FreeRTOS OS, it is easy to upgrade FreeRTOS OS. Just download new version of FreeRTOS OS and replace the rtos/FreeRTOS folder.

6.4.2 RPMsg

A multicore communication stack RPMsg is integrated with FreeRTOS BSP and can be used to communicate with the i.MX ARM Cortex-A9 core. It deals with the stack derived from the github OpenAMP project (see github.com/OpenAMP/open-amp) that has been modified by NXP and ported on i.MX 6SoloX. The stack can run in bare metal program and in the FreeRTOS application.

The main changes against the currently available implementation in OpenAMP are:

- Added generic RTOS-aware layer (providing new RTOS API)
- Added no-copy messaging mechanism
- Adjustment of porting layers, separation of environment and platform
- Several bugfixes in RPMsg core and in virtio

Besides the RPMsg API, no other feature from OpenAMP is included in this release. For more information, see the OpenAMP manuals under <install_dir>/middleware/multicore/open-amp/docs and the *NXP RPMSG RTOS Layer User's Guide* (RPMSGRTOSLAYERUG) under <install_dir>/doc.

7 Known Issues

Table 4 Known issues

Issue	Workaround
The U-Boot on the Cortex-A9 core cannot load a Cortex-M4 image from an SD Card to the Cortex-M4 core's Tightly Coupled Memory (TCM) directly. This is a hardware limitation.	Load the Cortex-M4 image from SD Card to OCRAM or DDR first and then copy the image from OCRAM or DDR to the Cortex-M4 core's TCM.

8 Revision History

This table summarizes revisions to this document.

Revision number	Date	Substantial changes
0	07/2016	Initial release.

How to Reach Us:

Home Page:
www.nxp.com

Web Support:
www.nxp.com/support

Document Number: FRTOS1016XRN
Rev. 0
07/2016