NXP Semiconductors
Release Notes

Document Number: MCUXSDKKE16RN
Rev. 1, 03/2020

MCUXpresso SDK Release Notes Supporting FRDM-KE16Z

Contents

1 Overview....1
2 MCUXpresso SDK....1
3 Development tools....2
4 Supported development systems....2
5 Release contents....2
6 MCUXpresso SDK release package....3
7 MISRA compliance....4
8 Known issues....6

1 Overview

The MCUXpresso Software Development Kit (SDK) is a collection of software enablement for Microcontrollers that includes peripheral drivers, other middleware packages, 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.

NOTE

See the attached Change Logs section at the end of this document to reference the device-specific driver logs, middleware logs, and RTOS log.

2 MCUXpresso SDK

As part of the MCUXpresso software and tools, MCUXpressoSDK is the evolution of Kinetis SDK v2.5.0, which includes support for both LPC and i.MX System-on-Chips (SoC). The same drivers, APIs, and middleware are still

available with support for Kinetis, LPC, and i.MX silicon. The MCUXpresso SDK adds support for the MCUXpresso IDE, a new Eclipse-based toolchain that works with all MCUXpresso SDKs. Easily import your SDK into the new toolchain to have access to all of the available components, examples, and demos for your target silicon. In addition to the MCUXpresso IDE, support for the MCUXpresso Config Tools allows for easy cloning of existing SDK examples and demos, allowing users to easily leverage the existing software examples provided by the SDK for their own projects.

NOTE

In order to maintain compatibility with legacy FSL code, the filenames and source code in MCUXpresso SDK containing the legacy Freescale prefix 'FSL' has been left as is. The 'FSL' prefix has been redefined as the NXP Foundation Software Library.

3 Development tools

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

- IAR Embedded Workbench for Arm version 8.32.1
- MDK-Arm Microcontroller Development Kit (Keil)® 5.26
- MCUXpresso IDE v10.3.0
- Makefiles support with GCC revision 7-2017-q4-major from Arm Embedded

4 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	MCU devices
FRDM-KE16Z	MKE14Z32VLF4, MKE14Z32VLD4, MKE14Z64VLF4, MKE14Z64VLD4, MKE15Z32VLF4, MKE15Z32VLD4, MKE15Z64VLF4, MKE15Z64VLD4, MKE16Z32VLF4, MKE16Z32VLD4, **MKE16Z64VLF4**, MKE16Z64VLD4, MKE14Z32VFP4, MKE14Z64VFP4, MKE15Z32VFP4, MKE15Z64VFP4

5 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
Driver examples	<install_dir>/boards/<board_name>/driver_examples
CMSIS driver examples	<install_dir>/boards/<board_name>/cmsis_driver_examples
RTOS examples	<install_dir>/boards/<board_name>/rtos_examples

Table continues on the next page...

Table 2. Release contents (continued)

Documentation	<install_dir>/docs
Middleware	<install_dir>/middleware
Motor Control libraries	<install_dir>/middleware/motor_control
NXP Touch libraries	<install_dir>/middleware/touch
RTCESL libraries	<install_dir>/middleware/rtcesl
Driver, SoC header files, extension header files and feature header files, utilities	<install_dir>/devices/<device_name>
Cortex Microcontroller Software Interface Standard (CMSIS) ARM Cortex®-M header files, DSP library source	<install_dir>/CMSIS
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

6 MCUXpresso 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.

6.1 Device support

The device folder contains all available software enablement for the specific System-on-Chip (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 MCU peripheral registers. The device header file provides an overall 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.

6.1.1 Board support

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

6.1.2 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 RTOS and middleware folders each contain examples demonstrating the use of the included source.

6.2 Middleware

6.2.1 RTOS

The MCUXpresso SDK is integrated with FreeRTOS OS.

6.2.2 CMSIS

The MCUXpresso SDK is shipped with the standard CMSIS development pack, including the prebuilt libraries.

7 MISRA compliance

All MCUXpresso SDK drivers and USB stack comply to MISRA 2012 rules with the following exceptions.

Table 3. MISRA exceptions

Exception Rules	Description
Directive 4.4	Sections of code should not be commented out.
Directive 4.5	Identifiers in the same name space with overlapping visibility should be typographically unambiguous.
Directive 4.6	Typedef that indicate size and signedness should be used in place of the basic numerical type.
Directive 4.8	If a pointer to a structure or union is never dereferenced within a transaction unit then the implementation of the object should hidden.
Directive 4.9	A function should be used in preference to a function like macro where they are interchangeable.
Directive 4.10	Precautions shall be taken in order to prevent the contents of a header file being included more than once.
Directive 4.11	The validity of values passed to library functions shall be checked.
Rule 2.3	A project should not contain unused type declarations.
Rule 2.4	A project should not contain unused tag declarations.
Rule 2.5	A project should not contain unused macro declarations.
Rule 2.7	There should be no unused parameters in functions.

Table continues on the next page...

Table 3. MISRA exceptions (continued)

Rule 3.1	The character sequences /* and // shall not be used within a comment.
Rule 5.1	External identifiers shall distinct.
Rule 5.3	A identifier declared in an inner scope shall not hide an identifier declared in an outer scope.
Rule 5.7	A tag name shall be a unique identifier.
Rule 5.9	Identifiers that define objects or functions with external linkage shall be unique.
Rule 8.13	A pointer should point to a const-qualified type whenever possible.
Rule 8.3	All declarations of an object or function shall use the same names and type qualifiers.
Rule 8.6	An identifier with external linage shall have exactly one external definition.
Rule 8.7	Octal constants shall not be used.
Rule 8.9	A object should be defined at block scope if its identified only appears in a single function.
Rule 10.1	Operands shall not be of an inappropriate essential type.
Rule 10.3	The value of an expression shall not be assigned to an object with a narrower essential type of a different essential type category.
Rule 10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category.
Rule 10.5	The value of an expression should not be cast to an inappropriate essential type.
Rule 10.6	The value of a composite expression shall not be assigned to an object with wider essential type.
Rule 10.7	If a composite expression is used as one operand of an operator in which the usual arithmetic conversions are performed then the other operand shall not have wider essential type.
Rule 10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type.
Rule 11.1	Conversions shall not be performed between a pointer to a function and any other type.
Rule 11.3	A case shall not be performed between a pointer to object type and a pointer to a different object type.
Rule 11.4	A conversion should not be performed between a pointer to object and an integer type.
Rule 11.5	A conversion should not be performed from pointer to void into pointer to object.
Rule 11.6	A cast shall not be performed between pointer to void and an arithmetic type.
Rule 12.1	The precedence of operators within expressions should be made explicit.

Table continues on the next page...

Table 3. MISRA exceptions (continued)

Rule 12.2	The right hand operator of a shift operator shall lie in the range zero to one less than the width in bits of the essential type of the left hand operand.
Rule 13.3	A full expression containing an increment(++) or decrement(--) operator should have no other potential side effects other than that caused by the increment or decrement operator.
Rule 13.5	The right hand operand of a logical && or || operator shall not contain persistent side effects.
Rule 14.2	A for loop shall be well formed.
Rule 14.4	The controlling expressions of an statement and the controlling expression of an iteration-statement shall have essentially Boolean type.
Rule 15.5	A function should have a single point of exit at the end.
Rule 16.1	All switch statements shall be well-formed.
Rule 17.1	The feature of <stdarg.h> shall not be used.
Rule 18.4	The +,-,+=and -=operators should not be applied to an expression of pointer type.
Rule 19.2	The union keyword should not be used.
Rule 20.1	#include directives should only be preceded by preprocessor directives or comments.
Rule 20.10	The #and ## preprocessor operators should not be used.
Rule 21.1	#define and #undef shall not be used on a reserved identifier or reserved macro name.

8 Known issues

8.1 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.

9 Revision history

Table 4. Revision history

Revision number	Date	Substantive changes
0	12/2018	Initial release
1	03/2020	Update Table 1

MCUXpresso SDK Release Notes Supporting FRDM-KE16Z

Change Logs

Contents

Driver Change Log 1

ACMP . . . 1
ADC12 . . . 1
CLOCK . . . 1
CRC . . . 2
EWM . . . 2
FLASH . . . 2
FTM . . . 3
GPIO . . . 4
LPI2C . . . 4
LPIT . . . 5
LPSPI . . . 6
LPTMR . . . 6
LPUART . . . 6
MMDVSQ . . . 7
MSCAN . . . 8
PDB . . . 8
PMC . . . 8
PORT . . . 8
PWT . . . 9
RCM . . . 9

Contents

Title	Page Number
RTC	9
SIM	9
SMC	10
TRGMUX	10
WDOG32	10

1 Driver Change Log

ACMP

The current ACMP driver version is 2.0.4.

- 2.0.4
  - Bug fix:
    - Avoid change w1c bit in ACMP_SetRoundRobinPreState();
- 2.0.3
  - Added feature functions for different power domain's usage (1.8 V and 3 V). These functions are first enabled in ULP1. They are about:
    - ACMP_EnableLinkToDAC()
    - ACMP_SetDiscreteModeConfig()
    - ACMP_GetDefaultDiscreteModeConfig()
- 2.0.2
  - Coding style changes:
    - Changed coding style of peripheral base address from "s_acmpBases" to "s_acmpBase";
- 2.0.1
  - Bug fix:
    - Fixed bug regarding the function "ACMP_SetRoundRobinConfig". It will not continue execution but returns directly after disabling round robin mode;

ADC12

The current ADC12 driver version is 2.0.2.

- 2.0.2
  - Bug fix:
    - Set ADC clock frequency as half of the maximum value for calibration.
- 2.0.1
  - Bug fix:
    - Add feature to control the enablement of DMA.
- 2.0.0
  - Initial version.

CLOCK

Current CLOCK driver version is 2.0.0

- 2.0.0
  - Initial version.

CRC

The current CRC driver version is 2.0.1.

- 2.0.1
  - Bug fix:
    - DATA and DATALL macro definition moved from header file to source file.
- 2.0.0
  - Initial version.

EWM

The current EWM driver version is 2.0.1.

- 2.0.1
  - Fixed EWM_Deinit hardfault issue.
- 2.0.0
  - Initial version.

FLASH

The current FLASH driver version is 3.0.0.

- 3.0.0
  - Improvements:
    - Reorganized FTFx Flash driver source file.
    - Extracted Flash cache driver from FTFx driver.
    - Extracted FLEXNVM flash driver from FTFx driver.
- 2.3.1
  - Bug fixes:
    - Unified Flash IFR design from K3.
    - New encoding rule for K3 flash size.
- 2.3.0
  - New features:
    - Added support for device with LP flash (K3S/G).
    - Added Flash prefetch speculation APIs.
  - Improvements:
    - Refined flash_cache_clear function.
    - Reorganized the member of flash_config_t struct.
- 2.2.0
  - New features:
    - Supported FTFL device in FLASH_Swap API.
    - Supported various pflash start addresses.
    - Added support for KV58 in cache clear function.
    - Added support for devices with secondary flash (KW40).

  - Bug fixes:
    - Compiled execute-in-ram functions as PIC binary code for driver use.
    - Added missed FLEXRAM properties.
    - Fixed unaligned variable issue for execute-in-ram function code array.
- 2.1.0
  - Improvements:
    - Updated coding style to align with KSDK 2.0.
    - Different alignment size support for PFLASH and FLEXNVM.
    - Improved the implementation of execute-in-ram functions.
- 2.0.0
  - Initial version.

FTM

The current FTM driver version is 2.1.0.

- 2.1.0
  - New feature:
    - Add a new API FTM_SetupPwmMode() to allow user set the channel match value in unit of timer ticks. New configure structure called ftm_chnl_pwm_config_param_t was offerred to configure the channel's PWM parameters. This API is similar with FTM_-SetupPwm() API, but the new API will not set the timer period(MOD value), it will be useful for users to set the PWM parameters without changing the timer period.
  - Bug fixes:
    - Add feature macro to enable/disable the external trigger source configuration.
- 2.0.4
  - Features:
    - Added to enable DMA transfer with new API:
      - FTM_EnableDmaTransfer()
- 2.0.3
  - Bug fixes:
    - Updated the FTM driver to enable fault input after configuring polarity.
- 2.0.2
  - Features:
    - Added support to Quad Decoder feature with new APIs:
      - FTM_GetQuadDecoderFlags()
      - FTM_SetQuadDecoderModuloValue()
      - FTM_GetQuadDecoderCounterValue()
      - FTM_ClearQuadDecoderCounterValue()
- 2.0.1
  - Bug fixes:
    - Updated the FTM driver to fix write to ELSA and ELSB bits.
    - FTM combine mode: set the COMBINE bit before writing to CnV register.
- 2.0.0

  - Initial version.

GPIO

The current driver version is 2.3.2.

- 2.3.2
  - Fix the issue for MISRA-2012 check.
    - Fixed rule 3.1.
- 2.3.1:
  - Remove deprecated APIs.
- 2.3.0:
  - New feature:
    - Update the driver code to adapt the case of interrupt configurations in GPIO module. New APIs were added to configure the GPIO interrupt settings if the module has this feature on it.
- 2.2.1:
  - API interface changes:
    - Refined naming of API while keep all original APIs by marking them as deprecated. Original API will be removed in next release. The main change is update API with prefix of _PinXXX() and _PortXXX.
- 2.1.1:
  - API interface changes:
    - Added API for the check attribute bytes.
- 2.1.0:
  - API interface changes:
    - Added "pins" or "pin" to some APIs' names.
    - Renamed "_PinConfigure" to "GPIO_PinInit".

LPI2C

The current LPI2C driver version is 2.1.5.

- 2.1.5
  - Bug fix:
    - Extended the Driver IRQ handler to support LPI2C4 and change to use ARRAY_SIZE(kLpi2cBases) instead of FEATURE COUNT to decide the array size for handle pointer array.
    - 2.1.4
  - Bug fix:
    - Fixed the LPI2C_MasterTransferEDMA receive issue when LPI2C share same request source for TX/RX DMA request. In the previous way the API uses scatter gather method, handle command transfer first, then handles the linked TCD which preset with the receive data transfer. The issue is that TX DMA request and RX DMA request are both enabled,

when DMA finished the first command TCD transfer and handled the receive data TCD, the TX DMA request still happens due to TX FIFO empty. This results the RX DMA transfer starts, without waiting on the expected RX DMA request. Fix the issue by enabling IntMajor interrupt for the command TCD and checking if there is a linked TCD to disable the TX DMA request in LPI2C_MasterEDMACallback API.

- 2.1.3
  - Improvement:
    - Added LPI2C_WATI_TIMEOUT macro to allow the user to specify the timeout times for waiting flags in functional API and blocking transfer API.
    - Added LPI2C_MasterTransferBlocking API.
- 2.1.2
  - Bug fix:
    - In LPI2C_SlaveTransferHandleIRQ, reset the slave status to idle when stop flag is detected.
- 2.1.1
  - Bug fix:
    - Disabled auto stop feature in eDMA driver. Previously, the autostop feature was enabled at transfer when transferring with stop flag. If the previous transfer was without stop flag, because the auto stop feature is enabled, then when starting a new transfer with stop flag, the stop flag sends before starting the new transfer, and the start flag cannot successfully send, so the transfer can not start.
    - Changed default slave configuration with address stall false.
- 2.1.0
  - API name change:
    - LPI2C_MasterTransferCreateHandle -> LPI2C_MasterCreateHandle.
    - LPI2C_MasterTransferGetCount -> LPI2C_MasterGetTransferCount.
    - LPI2C_MasterTransferAbort -> LPI2C_MasterAbortTransfer.
    - LPI2C_MasterTransferHandleIRQ -> LPI2C_MasterHandleInterrupt.
    - LPI2C_SlaveTransferCreateHandle -> LPI2C_SlaveCreateHandle.
    - LPI2C_SlaveTransferGetCount -> LPI2C_SlaveGetTransferCount.
    - LPI2C_SlaveTransferAbort -> LPI2C_SlaveAbortTransfer.
    - LPI2C_SlaveTransferHandleIRQ -> LPI2C_SlaveHandleInterrupt.
- 2.0.0
  - Initial version.

LPIT

The current LPIT driver version is 2.0.0.

- 2.0.0
  - Initial version.

LPSPI

The current LPSPI driver version is 2.0.3.

- 2.0.3
  - Bug Fix:
    - Remove the LPSPI_Reset() from LPSPI_MasterInit() and LPSPI_SlaveInit(), because this API may glitch the slave select line, if needed, please call this function manually.
- 2.0.2
  - New feature:
    - Added dummy data setup API to allow users to configure the dummy data to be transferred.
    - Enabled the 3-wire mode, SIN and SOUT pins can be configured as input/output pin.
- 2.0.1
  - Bug fix:
    - The clock source should divided by PRESCALE setting in LPSPI_MasterSetDelayTimes function.
    - Fixed the bug that LPSPI_MasterTransferBlocking function would hang in some corner cases.
  - Optimization:
    - Added #ifndef/#endif to allow user to change the default TX value at compile time.
- 2.0.0
  - Initial version.

LPTMR

The current LPTMR driver version is 2.0.1.

- 2.0.1
  - Driver update:
    - Updated the LPTMR driver to support 32-bit CNR and CMR registers in some devices.
- 2.0.0
  - Initial version.

LPUART

The current LPUART driver version is 2.2.7.

- 2.2.7
  - Fix the issue for MISRA-2012 check.
    - Fixed rule-12.1, rule-17.7.
- 2.2.6
  - Fix the repeatly reading status register issue while dealing with the IRQ routine.
- 2.2.5
  - Do not set or clear the TIE/RIE bits when using LPUART_EnableTxDMA() and LPUART_-

EnableRxDMA().

- 2.2.4
  - Added hardware flow control function support.
  - Added idle line detected feature in LPUART_TransferNonBlocking function. If an idle line was detected, a callback is triggered with status kStatus_LPUART_IdleLineDetected returned. This feature may be useful when the received Bytes is less than the expected receive data size. Before triggering the callback, data in the FIFO (if has FIFO) is read out, and all interrupts will not be disabled, except if the receive data size reaches 0.
  - Enabled the RX FIFO watermark function. With the idle line detected feature enabled, you can set the watermark value to whatever you want (should be less than the RX FIFO size). Data is received and a callback is triggered when data receive is end.
- 2.2.3
  - Changed parameter type in LPUART_RTOS_Init() struct rtos_lpuart_config –> lpuart_rtos_config_t.
  - Bug fix:
    - Disabled LPUART receive interrupt instead of disabling all NVIC when read data from ring buffer. Because the ring buffer is used, receive nonblocking disables all NVIC interrupts to protect the ring buffer. This has a negative effect to other IPS which are using the interrupt.
- 2.2.2
  - Added software reset feature support.
  - Added software reset API to LPUART_Init().
- 2.2.1
  - Added separate RX,TX IRQ number support.
- 2.2.0
  - Added 7 data bits and MSB support.
- 2.1.1
  - Removed needless check of event flags and assert in LPUART_RTOS_Receive.
  - Always wait for RX event flag in LPUART_RTOS_Receive.
- 2.1.0
  - Update transactional APIs.

MMDVSQ

The current MMDVSQ driver version is 2.0.2.

- 2.0.2
  - Bug fix:
    - Fixed MMDVSQ_GetExecutionStatus function get execution status wrong.

2.0.1

- Other changes:
  - Changed name of MMDVSQ_GetDivideRemainder and MMDVSQ_GetDivideQuotient functions.

2.0.0

- Initial version.

MSCAN

The current MSCAN driver version is 2.0.1.

- 2.0.1
  - Fix timestamp can't be enabled issue.
  - Fix standard mode frame buffer configuration.
  - Add RX Message Buffer Mask helper macro, MSCAN_RX_MB_STD_MASK, MSCAN_RX_MB_EXT_MASK.
  - Fix dataByte[0-7] order in struct type mscan_frame_t.
  - Update function MSCAN_WriteTxMb. The MSCAN registers are 8bits, using 8bits write for registers.
- 2.0.0
  - Initial version.

PDB

The current PDB driver version is 2.0.1.

- 2.0.1
  - Changed PDB register base array to const.
- 2.0.0
  - Initial version.

PMC

The current PMC driver version is 2.0.0.

- 2.0.0
  - Initial version.

PORT

The current PORT driver version is 2.1.0.

- 2.1.0
  - New feature:
    - Update the driver code to adapt the case of the interrupt configurations in GPIO module. will move the pin configuration APIs to GPIO module.
- 2.0.2

  - Miscellaneous changes:
    * Added feature guard macros in the driver.
- 2.0.1
  - Miscellaneous changes:
    * Added "const" in function parameter.
    * Updated some enumeration variables' names.

PWT

The current PWT driver version is 2.0.0.

- 2.0.0
  - Initial version.

RCM

The current RCM driver version is 2.0.1.

- 2.0.1
  - [KPSDK-10249] Fixed kRCM_SourceSw bit shift issue.
- 2.0.0
  - Initial version.

RTC

The current RTC driver version is 2.1.1.

- 2.1.1
  - Fix MISRA C-2012 issue. -Fixed rule contain: rule-17.7, rule-14.4.
- 2.1.0
  - Add feature macro check for many features.
- 2.0.0
  - Initial version.

SIM

The current SIM driver version is 2.1.0.

- 2.1.0
  - Added new APIs of SIM_GetRfAddr() and SIM_EnableSystickClock().
- 2.0.0
  - Initial version.

SMC

The current SMC driver version is 2.0.5.

- 2.0.5
  - Fix the issue for MISRA-2012 check.
    - Fixed rule 15.7.
- 2.0.4
  - When entring stop modes, use ram function for the flash synchronize issue. Application should make sure that, the rw data of fsl_smc.c is located in memory region which is not powered off in stop modes.
- 2.0.3
  - Added APIs SMC_PreEnterStopModes, SMC_PreEnterWaitModes, SMC_PostExitWaitModes, and SMC_PostExitStopModes.
- 2.0.2
  - Bug fix:
    - Added DSB before WFI, add ISB after WFI.
  - Miscellaneous changes:
    - Updated SMC_SetPowerModeVlpw implementation.
- 2.0.1
  - Miscellaneous changes:
    - Updated for KL8x.
- 2.0.0
  - Initial version.

TRGMUX

The current TRGMUX driver version is 2.0.0.

- 2.0.0
  - Initial version.

WDOG32

The current WDOG32 driver version is 2.0.1.

- 2.0.1
  - Bug fixes:
    - WDOG must be configured within it's configuartion time period
      - Added WDOG32_Init API to quick access section.
      - Defined register variable in WDOG32_Init API.
- 2.0.0
  - Initial version.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP 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 NXP 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. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

Document Number MCUXSDKKE16RN
Revision 1, 03/2020