TC3xx_SW_MCAL_DemoApp

User Manual

About this document

1.1 Scope and purpose

This application note provides details for building and running the Demo Application. The Demo Application enables users to perform preliminary test on delivered MCAL driver.

1.2 Intended audience

This document is intended for anyone who needs to perform a quick test of the product delivery.

Table of contents

1.1 Scope and purpose 1
1.2 Intended audience 1
Table of contents 1
2 Getting started 3
3 Building the DemoApp 4
3.1 Command to merge multiple configured xdm to single epc file 4
3.2 Command to generate code from epc file 5
3.3 Utilities used for Building the DemoApp 5
3.4 DemoApp Resource Usage 5
3.5 Integrating DEM module 5
4 Running the DemoApp 7
4.1 Option a: ADC Demo 8
4.2 Option b: Can_17_McmCan Demo 8
4.3 Option c: Dio Demo 9
4.4 Option d: Dma Demo 9
4.5 Option e: Eth_17_GEthMac Demo 10
4.6 Option f: Fee Demo 10
4.7 Option g: Fls_17_Dmu Demo 11
4.8 Option h: Fr_17_Eray Demo 11
4.9 Option i: Gpt Demo 11
4.10 Option j: Icu_17_TimerIp Demo 12
4.11 Option k: Mcu Demo 12
4.12 Option l: Pwm_17_GtmCcu6 Demo 12
4.13 Option m: Spi Demo 13
4.14 Option n: Wdg_17_Scu Demo 13
4.15 Option o: Crc Demo 14
4.16 Option p: FlsLoader Demo 14
4.17 Option q: Smu Demo 15
Revision History 15

2 Getting started

The following steps must be performed, before building/running the DemoApp.

1. Install the Mcal package and ensure the tool versions required are also installed as per the Release Notes.
2. If the EB tresos Studio GUI is operational, close the GUI before proceeding to next step.
3. Copy all the plugins provided in to EB tresos plugin folder. The installed package plugin folder can be located at this path (`<Installed-Package>\McIsar\PluginsTresos\eclipse\plugins`).

Note: The configured XDMs are applicable to TC389 & TC387 which are in TC38A folder and TC397,TC397adas, TC399 which are in TC39B folder. The XDMs must be configured according to the configuration parameters provided in the `TC3xx_SW_MCAL_DemoConfigs.xls` document, which is common for both TC38x and TC39x devices. In addition, this document also provides additional information regarding the resource requirements, for example interrupt priority, Port pin usage, DEM usage and GTM usage for the DemoApp.

3 Building the DemoApp

The DemoApp build is launched by calling the `DemoAppBuild.bat` present in `<Installed Package>\DemoWorkspace\McalDemo` folder. The application (DemoApp files in this case), source and configuration files are compiled and linked together. The output files, `*.elf/*.hex`, are generated after completion of the build process in the below folder location,
`<InstalledPackage>\DemoWorkspace\McalDemo\<TC38A/TC39B>\2_Out\<Tricore_Tasking/Tricore_Gnuc/Tricore_Dcc>`

The output `DemoApp_Node0.elf/DemoApp_Node0.hex` and `DemoApp_Node1.elf/DemoApp_Node1.hex` file are generated if the FR module is selected for installation.

a. DemoApp execution of all modules except for FR use `DemoApp_Node0.elf/DemoApp_Node0.hex`
b. DemoApp execution of FR module use `DemoApp_Node1.elf/DemoApp_Node1.hex` along with `DemoApp_Node0.elf/DemoApp_Node0.hex`

The output file `DemoApp.elf/DemoApp.hex` is generated if the FR module is not selected for installation.
The output files are generated in the following folder location,
`<Installed-Package>\DemoWorkspace\McalDemo\<TC38A/TC39B>\2_Out\<Tricore_Tasking/Tricore_Gnuc/Tricore_Dcc>`

Note: All packages (BASIC, COM-E, CD and DEMO) with all provided drivers should be installed in the same directory in order to build the DemoApp provided for MCAL drivers. However, it is possible to install individual packages in a separate directory for use but the DemoApp cannot be built individually/selectively.

Note: In case of selective installation (for example, only ETH needs to be is installed from COM-E package), remove the modules and their respective `.xdm` files from the EB tresos workspace. Also remove the driver source code, demo code, Irq and integration files.

Follow the steps below for building the DemoApp executable:

1. Open the command prompt in the `<Installed Package>\DemoWorkspace\McalDemo\<TC38A/TC39B>` folder.
2. DemoAppBuild.bat parameters:
   Option 1: Add command ‘GNU’ to build with GnuC compiler
   Option 2: Add command ‘DCC’ to build the Windriver compiler
   Option 3: It takes Tasking compiler as default compiler

Example: Code listing for running the DemoApp

```
DemoAppBuild.bat
```

```
DemoAppBuild.bat GNU
```

3. The batch file follows two steps, generation of files and compilation/linking of files and generates the executable in the folder `<Installed-Package>\DemoWorkspace\McalDemo\<TC38A/TC39B>\2_Out\<Tasking/Gnuc/DCC>`.

3.1 Command to merge multiple configured xdm to single epc file

An example command to merge multiple configured `XDM` files to generate an `EPC` file is as follows:

Example: Merging the xdm to generate EPC

```
C:\<tresos-path>\bin\tresos_cmd.bat  -DMapOptionalAsList=false
-Dtarget=AURIX2G -Dderivate=TC389/87 legacy convert Port.xdm Pwm.xdm
Spi.xdm Adc.xdm Dem.xdm Dio.xdm EcuM.xdm Gpt.xdm Icu.xdm Irq.xdm Mcu.xdm
ResourceM.xdm Combined.epc@asc:4.2.2
```

Note: "-Dderivate" should be changed as per the desired derivate to run the DemoApp

3.2 Command to generate code from epc file

An example command to generate code from the `EPC` file is as follows:

Example: Command to generate code from EPC file

```
C:\<tresos-path>\tresos\bin\tresos_cmd.bat -Dtarget=AURIX2G
-Dderivate=TC389/87 legacy generate Combined.epc@asc:4.2.2 -o outputdir -g
Irq_Aurix2GAS422 -g Mcu_Aurix2GAS422 -g Port_Aurix2GAS422 -g
Dem_Aurix2GAS422 -g EcuM_Aurix2GAS422 -g Adc_Aurix2GAS422 -g
Dio_Aurix2GAS422 -g Gpt_Aurix2GAS422 -g Icu_17_TimerIp_Aurix2G422 -g
Pwm_17_GtmCcu6Ccu6_Aurix2G422 -g Spi_Aurix2GAS422 -g
ResourceM_Aurix2GAS422
```

3.3 Utilities used for Building the DemoApp

The following utilities are used for the build process:

Table 1 Utilities Used for the Build Process

Description	Path / Where to find it
Make file used for building the application, the file contains rules for generating the executable	`<Installed-Package>\DemoWorkspace\McalDemo\<TC38A/TC39B>\1_ToolEnv\0_Build`

Note: Any errors in compilation and linking will be displayed in the DOS prompt itself.

3.4 DemoApp Resource Usage

Refer to the `TC3xx_SW_MCAL_DemoConfigs.xls` for details.

3.5 Integrating DEM module

Non-productive DEM is delivered along with the MCAL package; user is expected to replace Non-Productive Dem with Productive Dem module. If Non-Productive Dem module is used, any new event has to be added manually in the generated Dem_cfg.h available in following path of installed files,

`DemoWorkspace\McalDemo\<TC38A/TC39B>\0_Src\BaseSw\Infra\Autosar_Srv\Dem_cfg.h`

The code listing below provides an example for adding a new event.

Example: Adding DEM Event in Dem_cfg.h file

```
#define DemConf_DemEventParameter_FLS_E_ERASE_FAILED          (1U)
```

4 Running the DemoApp

Before running the DemoApp, flash the DemoApp executable to the board. Start the terminal application (for example, HyperTerminal) in PC with the following COM properties.

Table 2 Terminal properties

Property	Value
Baud rate	115200
No of data bits	8
Parity	None
Number of stop bits	1
Flow control	None

Run the program on target and the menu as below appears on terminal window.

DemoApp Main Menu

```
< >: ------------- MAIN Menu -------------
<0>: Go to MCAL DemoApp List
```

On pressing 0 in main menu, the following demo menu displays. If a particular driver demo is not available, the option for that driver will not be visible.

DemoApp Sub Menu

```
/*************************************************************************/
 < >: ............ AUTOSAR DRIVERS..............
 <a>:   Adc: Demo
 <b>:   Can_17_McmCan: Demo
 <c>:   Dio: Demo
 <d>:   Dma: Demo
 <e>    Eth_17_GEthmac: Demo
 <f>:   Fee: Demo
 <g>:   Fls_17_Dmu: Demo
 <h>    Fr_17_Eray: Demo
 <i>:   Gpt: Demo
 <j>:   Icu_17_TimerIp: Demo
 <k>:   Mcu: Demo
 <l>:   Pwm_17_GtmCcu6Ccu6: Demo
 <m>:   Spi: Demo
 <n>:   Wdg_17_Scu: Demo
 <o>:   Crc: Demo
 <p>:   FlsLoader: Demo
 <q>:   Smu: Demo
```

```
 <.>:   Go to Main Menu
/*****************************************************************/
```

4.1 Option a: ADC Demo

The Adc demo converts Adc channels AN0 and AN1 with 12-bit resolution. G0CH0 and G0CH1 channels are used for the TC38x.

Table 3 External connections for Adc Demo

Signal name	Connections to be made
AN0	Connect the potentiometer/variable voltage source
AN1	Connect the potentiometer/variable voltage source

Table 4 Adc DemoApp menu

Option	Description of the Demo
<1>	For ADC SW Group Demo The conversion results should be the Voltage supplied to the Pin's AN0 and AN1
<2>	For ADC HW Group Demo The conversion results should be the Voltage supplied to the Pin's AN0 and AN1
<x>	Back to main menu

4.2 Option b: Can_17_McmCan Demo

The DemoApp uses the CAN controllers 0 and 1 of MCMCAN. These two CAN nodes communicate through CAN00(X301) and CAN10(X302) of the TriBoard. The following connections have to be made externally to run the demo successfully.

Table 5 External connections for Can_17_McmCan Demo

Signal name	Connections to be made
CANL	(pin 3 of X301) <-> (pin 3 of X302)
CANH	(pin 4 of X301) <-> (pin 4 of X302)

Multiple CAN demo options are supported. The following table provides details of the CAN driver DemoApp menu.

Table 6 Can_17_McmCan DemoApp menu

Option	Description of the Demo
<1>	Transfer of Standard Frame between CAN controllers: **Operational Specific:** Sets the mode of CAN controller 0 & 1 to STARTED, Transmits two STANDARD ID frames each from controller 0 to 1 and 1 to 0, Sets the mode of CAN controller 0 & 1 to STOPPED.
<2>	Transfer of Extended Frame between CAN controllers: **Operational Specific:** Sets the mode of CAN controller 0 & 1 to STARTED, Transmits one EXTENDED ID frame each from controller 0 to 1 and 1 to 0, Sets the mode of CAN controller 0 & 1 to STOPPED.

Option	Description of the Demo
<3>	Mixed Mode Support: **Operational Specific:** Sets the mode of CAN controller 0 & 1 to STARTED, Transmits one STANDARD ID frame and 1 EXTENDED ID frame each from controller 0 to 1 and 1 to 0 (both Txn and Rxn are using MIXED Mode Hardware Object), Sets the mode of CAN controller 0 & 1 to STOPPED.
<4>	Test Enabling and Disabling of Tx and Rx Interrupts: **Operational Specific:** Demonstrates the behavior of Can_17_McmCan_DisableControllerInterrupts and Can_17_McmCan_EnableControllerInterrupts APIs.
<5>	Changing Baudrate using Can_SetBaudRate: **Operational Specific:** Demonstrates the usage of Can_17_McmCan_SetBaudrate API
<6>	Activation and Deactivation of Pretended networking: **Operational Specific:** Demonstrates the usage of Can_17_McmCan_SetIcomConfiguration API for Activation and Deactivation of pretended networking.
<7>	Trigger transmit functionality: **Operational Specific:** Demonstrates the usage of Can_17_McmCan_Write when Trigger transmits functionality is enabled.
<8>	FD frames transmission and reception: **Operational Specific:** Sets the mode of CAN controller 0 & 1 to STARTED, Transmits two FD frames each from controller 0 to 1 and 1 to 0, Sets the mode of CAN controller 0 & 1 to STOPPED.
<x>	Back to main menu

4.3 Option c: Dio Demo

The following table provides details of the Dio driver DemoApp menu.

Table 7 Dio DemoApp menu

Option	Description of the Demo
<1>	Set one LED port pin ON -Glows the first LED on Triboard
<2>	Set one LED port pin OFF -Switches Off the first LED on Triboard
<3>	Set the LED Series group ON -Glows the first three LEDs on Triboard
<4>	Set the LED Series group OFF -Switches Off the first three LEDs on Triboard
<x>	Back to main menu

4.4 Option d: Dma Demo

Dma demo transfers data form the source address to the destination address. On selecting option 4 software transaction from source memory to destination memory for given number of transfer count is triggered. The nature of transaction is dependent on various configuration parameters (available in EB tresos). The DMA channel 16 is used for data transfer.

"Source value" is value present at source address & "Original Destination value" is value present at destination address before transaction. After successful execution of demo, "Final Destination value" will display value present at destination address. "DMA DEMO PASS" in Final Destination value indicates success of demo.

4.5 Option e: Eth_17_GEthMac Demo

The Eth_17_GEthMac demo requires the following to be done before the demo is run:

- Connect the Triboard to a PC/Laptop using the Ethernet cable.
- Assign a static address to the PC Ethernet interface where the Triboard is connected.

Multiple Ethernet demo options are supported. The following table provides details of the Ethernet driver DemoApp menu.

Table 8 Eth_17_EthMac DemoApp menu

Option	Description of the Demo
<1>	Promiscuous Mode: Receive All Frames - Receives All Frames Transmitted on Ethernet Bus irrespective of the Destination Address in ETH Frame Note: Press the ESC key to STOP receiving frames
<2>	PING Application - Asks User to Enter the IP Address to Ping and then transmits an ARP Request to the IP Address Entered and Waits for an ARP Reply from the PC and prints MAC Address of the PC having that IP Address Note : Press ESC Key to Exit If WRONG IP is Entered and hence NO ARP Reply
<3>	Transmit ETH Frame of User Defined Payload Length - Asks User to Enter the PayLoad Length(42...1500) of ETH Frame and then transmits an Ethernet Frame of Frame Type(0xABCD) of User defined Payload length(Total Frame Length = Payload Length +18)
<4>	Read PHY Registers - Read Ethernet Transceiver(PHY) registers Values(CTRL,STAT,PHYCTL1,PHYCTL2,MIICTRL)
<x>	Back to main menu

4.6 Option f: Fee Demo

For the Fee demo, two NVM blocks and one QS block (Blk 3) have been configured. The Fee demo will erase the entire DFlash (i.e., DF_EEPROM) area and then the `Fee_Init` API will be called. This is done only for the first time when FEE demo is called after a power-on reset. Each time the FEE demo is called, the following steps will be run sequentially and automatically.

Sequence 1: Block 1 is written
Sequence 2: Block 2 is written
Sequence 3: Block 1 is written again with different data
Sequence 4: Block 2 is written again with different data
Sequence 5: Block 1 is read; Latest data written to Block 1 (i.e., the data written in Step 3) is read & compared
Sequence 6: Block 2 is read; Latest data written to Block 2 (i.e., the data written in Step 4) is read & compared
Sequence 7: Block 3 is written
Sequence 8: Block 3 is read; data written to Block 3 is read and compared.

Sequence 9: Block 3 is erased and block state is checked if it is set to ‘erased’.

After successful execution of above, sequences Fee demo results are displayed

4.7 Option g: Fls_17_Dmu Demo

The Fls_17_Dmu demo writes the data into the DFlash and then reads back from the Dflash. The read data is compared with the data written for verification.

Sequence 1: All sectors of data flash are erased and verified for erase success.
Sequence 2: Data is written into the DFlash.
Sequence 3: Written data is compared with 'margin 0' directly from Dflash.
Sequence 4: Written data is read into local buffer and compared.
Sequence 5: Cancel operation is verified by cancelling ongoing Read, Write and Erase operations.

After successful execution of above sequences Fls_17_Dmu demo results are displayed

4.8 Option h: Fr_17_Eray Demo

Table 9 External connections for Fr_17_Eray Demo

Signal name	Connections to be made
ERAY_A	Pin 3 of ERAY_A of Board 1 - Pin 3 of ERAY_A of Board 2
ERAY_A	Pin 4 of ERAY_A of Board 1 - Pin 4 of ERAY_A of Board 2
ERAY_A	Pin 5 of ERAY_A of Board 1 - Pin 5 of ERAY_A of Board 2
ERAY_B	Pin 3 of ERAY_B of Board 1 - Pin 3 of ERAY_B of Board 2
ERAY_B	Pin 4 of ERAY_B of Board 1 - Pin 4 of ERAY_B of Board 2
ERAY_B	Pin 5 of ERAY_B of Board 1 - Pin 5 of ERAY_B of Board 2

Fr_17_Eray demo establishes FR Cluster consisting of two nodes Node0 and Node1.

Step 1: Download the `DemoApp_Node0.hex` to Node0 Triboard and `DemoApp_Node1.hex` to Node1 triboard.
Step 2: Select the Option 1 for choosing FR Node 1 Slave node in Slave Triboard.
Step 3: Both nodes synchronize to cluster after Module initialization.
Step 4: All the cluster related information are read and verified in Node0.
Step 5: Node0 transmits the data to Node1 on to network.
Step 6: Node1 transmits the received data to Node0 on to network.
Step 7: Transmitted and received data on Node0 is verified.

4.9 Option i: Gpt Demo

The following table provides the Gpt demo options supported.

Table 10 Gpt DemoApp menu

Option	Description of the Demo
<1>	Start continuous timer: LED must start blinking for every 2 seconds; The demo starts GPT Channel in continuous mode with timeout value as 2 seconds. The notification function will be called after timeout; inside notification function, a LED is toggled.

Option	Description of the Demo
<2>	Stop timer: LED must stop blinking; the demo stops the GPT Channel.
<.>	Back to main menu

4.10 Option j: Icu_17_TimerIp Demo

Icu_17_TimerIp demo will measure the High Time, Duty Cycle and Period of the PWM generated by PWM demo. Hence, Output of PWM needs to be connected to the input of ICU. First, generate duty cycle using PWM demo then run Icu_17_TimerIp demo.

Table 11 External connections for Icu_17_TimerIp Demo

Signal name	Connections to be made
PWM <-> ICU	Pin 33.7 <->Pin 00.6 (Plug Connector X703: T2LA Board X10 Connector Pin 2 and X1 Connector Pin 3 needs to interconnected using a connecting wire)

The output will be displayed in the following format:

- High Time = xx
- Period = xx
- Duty Cycle = xx

Note: High Time and Period are displayed in hexadecimal; Duty Cycle is displayed in decimal.

Note: In case the connections are made (PWM->ICU), and the Duty Cycle measured is displayed as 0, then additional hint message is printed indicating that a capture operation is in progress.

4.11 Option k: Mcu Demo

Table 12 Mcu DemoApp menu

Option	Description of the Demo
<1>	Perform Software Reset : The Board will get a Soft Reset
<2>	Get Reset Reason
<x>	Back to main menu

Note: In order to display the menu on the HyperTerminal after the Option <1>, press the ENTER key on the keyboard.

Note: The Mcu demo should be run before the Wdg_17_Scu demo otherwise the MCU software reset demo might not work.

4.12 Option l: Pwm_17_GtmCcu6 Demo

Table 13 Pwm_17_GtmCcu6 DemoApp menu

Option	Description of the Demo
<1>	Start PWM, Default DutyCycle :50%
<2>	Enter New DutyCycle: For Example: 0 75, Where 0 is Id (Pwm Channel Id) and 75 is Data (duty in

Option	Description of the Demo
	%)
<3>	Stop PWM Demo
<4>	Back to main menu

4.13 Option m: Spi Demo

Spi DemoApp uses QSPI0. It transmits (Sync Transmit, Spi Level Delivered 2 and SpiChannelBuffersAllowed EB) and reads the data on pins of triboard.

Table 14 External connections for Spi Demo

Signal name	Connections to be made
MTSR <-> MRTS	P20.14 <-> P20.12 (Plug Connector X702: T2LA Board X3 Connector Pin 6 and Pin 5 needs to interconnected using a connecting wire)

SPI demo transmits {0x1A, 0x01, 0x18, and 0x00} on all four logical channels using internal buffers. Sequence and Job result are observed to ensure all data transmitted and nothing pending.

Table 15 Spi DemoApp menu

Option	Description of the Demo
<1>	Transfer 8-bit data from QSPI0 using sequence 0
<x>	Back to main menu

4.14 Option n: Wdg_17_Scu Demo

The following table provides the Wdg_17_Scu demo options supported.

Table 16 Wdg_17_Scu DemoApp menu

Option	Description of the Demo
<s>	Set WDG0 in slow mode. Each press of key‘s’ will set trigger timeout period of 7 Seconds. 1st LED on the Tri-Board toggles until watchdog timeout occurs (at the rate of WdgSlowRefreshTime).
<f>	Set WDG0 in Fast mode. Each press of key ‘f’ will set trigger timeout period of 4 Seconds. 1st LED on the Tri-Board toggles until watchdog timeout occurs (at the rate of WdgFastRefreshTime).
<t>	Trigger WDG0 timeout for 4 seconds in current mode. 1st LED on the Tri-Board toggles until watchdog timeout occurs.
<x>	Stop WDG0 Timer.
<.>	Stop All WDG Timers and exit WDG demo.

Note: The Wdg_17_Scu demo uses the STM timer internally. If this demo is run after 42.94 seconds (the timeout value for STM), DET is observed in the Wdg_17_Scu demo.

Note: Disconnect the Tri-Board from Debugger and reset once before running WDG demo.

4.15 Option o: Crc Demo

Table 17 Crc DemoApp menu

Option	Description of the Demo
<1>	CRC Driver Demo 8bitCRC with 0x1D polynomial and “MODE” based method calculation
<2>	CRC Driver Demo 8bitCRC with 0x2F polynomial and “MODE” based method calculation
<3>	CRC Driver Demo 16bitCRC with 0x1021 polynomial and “MODE” based method calculation
<4>	CRC Driver Demo 32bitCRC with 0x04C11DB7 polynomial and “MODE” based method calculation
<5>	CRC Driver Demo 32bitCRC with 0xF4ACFB13 polynomial and “MODE” based method calculation
<6>	CRC Version Information
<x>	Back to main menu

Note: MODE is the implementation mode of that Crc and is selected from the configuration. It is Runtime, Table or Hardware.

4.16 Option p: FlsLoader Demo

In this demo FlsLoader, basic APIs usage is demonstrated. First `FlsLoader_Init` API’s execution status is printed and then following menu options will be given.

Table 18 FlsLoader DemoApp Menu

Option	Description of FlsLoader Demo
<1>	DFlash Programming: DFlash erase and programming demo. Need to provide the valid start address of a Dflash sector, which needs to be erased and programmed. Example: <Parameter ID> <Space><Dflash sector start address in Hex> 0 AF000000<Enter Key> <Enter Key> (total 2 times) Erases single Dflash sector as per the input address, then programs with 512 bytes of dummy data. Also prints results of each operation.
<2>	DFlash Read: DFlash read demo. Reads first 8 words from the previously programmed location and prints on the screen.
<3>	PFlash Programming: PFlash erase and programming demo. Need to provide the valid start address of a Pflash sector, which needs to be erased and programmed. Example: <Parameter ID> <Space><Dflash sector start address in Hex> 0 A0500000<Enter Key> <Enter Key> (total 2 times) Erases single Pflash sector as per the input address, then programs with 512 bytes of dummy data. Also prints results of each operation.

Option	Description of FlsLoader Demo
<4>	PFlash Read: PFlash read demo. Reads first 8 words from the previously programmed location and prints on the screen.
<x>	Go back to main menu: Calls `FlsLoader_DeInit` API prints the returned value and exits the FlsLoader demo.

Note: The lock and unlock features demo is implemented in the DemoApp, but it is disabled by a compile switch to avoid unexpected memory lock situations.

4.17 Option q: Smu Demo

The following table provides the Smu demo options supported. This demo will trigger the SW alarm 0, to generate & verify the SMU Interrupt Request 0

Table 19 Smu DemoApp menu

Option	Description of the Demo
<1>	Trigger SWAlarm10[0] to send the SMU Interrupt Req0
<x>	Back to main menu

Revision History

Date	Version	Description of change
2017-12-14	Release v1.0	Initial Version of User Manual.
2018-05-30	Release v2.0	Modified Version of User Manual for Beta package.
2018-10-05	Release v3.0	Modified for ADC HW group demo.

Edition 2018-10-05

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference
AP32383

IMPORTANT NOTICE

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (**www.infineon.com**).

WARNINGS

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.