XMC1000

32-bit Microcontroller Series for Industrial Applications

Driving LED Strips with the RGB LED Lighting Shield

AP32313

Application Note

About this document

Scope and purpose

This document provides hints for using the RGB LED Lighting Shield to drive and control LED strip lights.

Intended audience

This document is intended for engineers interested in evaluating or making prototypes with the RGB LED Lighting Shield.

Applicable Products

- XMC1202

References

Infineon: DAVE™, http://www.infineon.com/dave

Infineon: XMC Family, http://www.infineon.com/XMC

The application code can be downloaded from http://www.infineon.com/arduino

Table of Contents

1 Introduction

The RGB LED Lighting Shield is designed to drive up to 3 parallel LED strings with constant current. This constant current can be modulation-dimmed to control brightness and color.

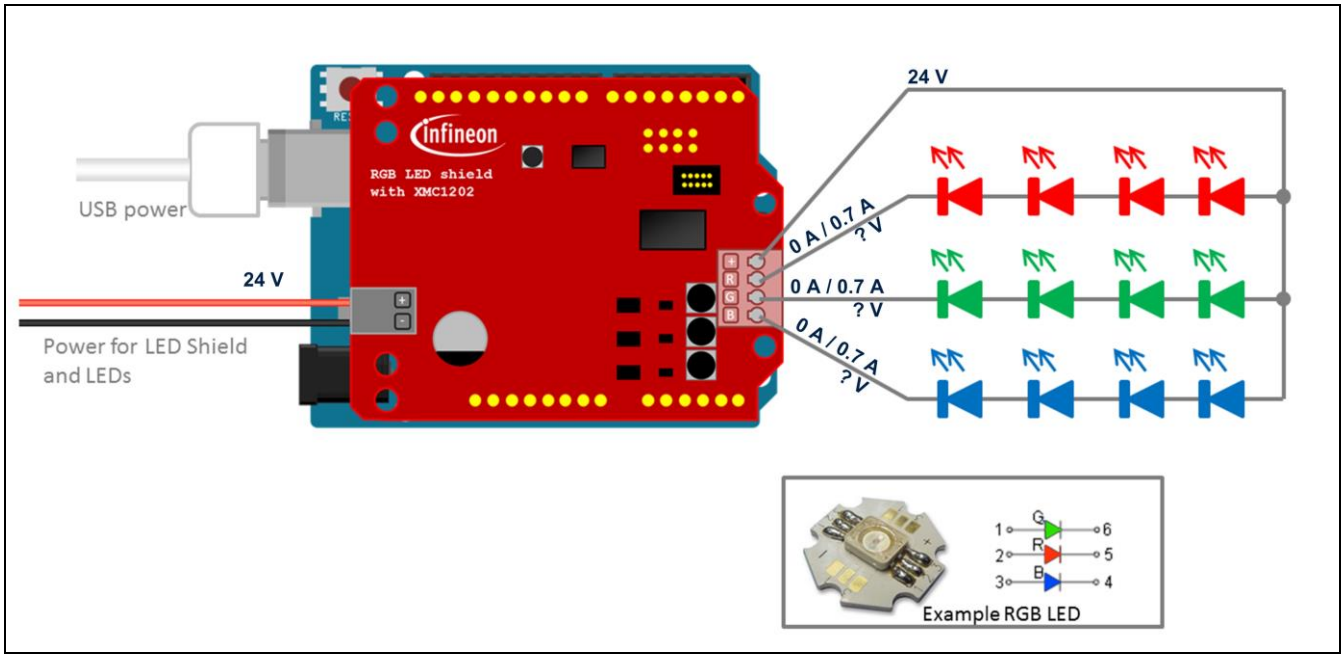

Figure 1 RGB LED Lighting Shield with LED strings connected (original by shabaz from here)

The RGB LED Lighting Shield contains 3 inverted buck converters with current feedback. Each buck converter can maintain constant current at its output. However, the output voltage depends on the connected LED string and cannot be controlled by the buck converter.

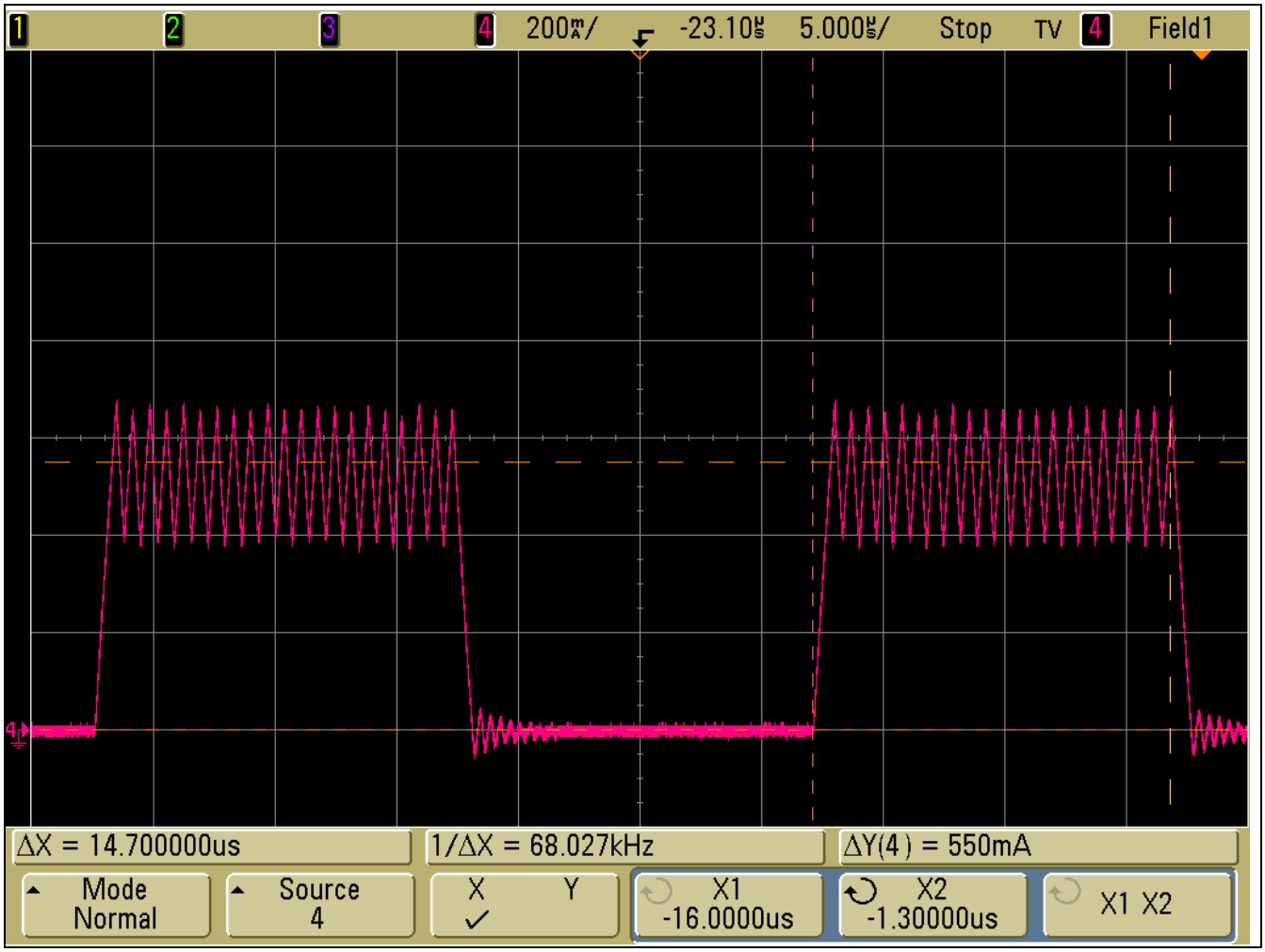

Figure 2 Typical modulation dimmed output current

LED strips are more than LED strings. LED strings behave like diodes and require constant current. LED strips contain current control circuitry and require constant voltage, typically 12V or 24V. They can be modulation-dimmed by turning this voltage on-off. The current control circuitry is typically either a series resistor or a linear LED driver IC, such as BCR421U.

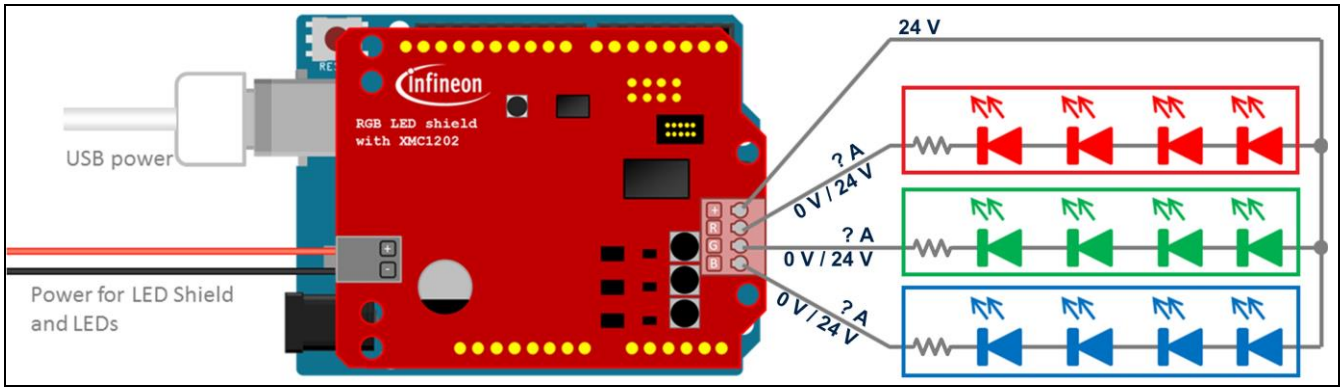

Figure 3 RGB LED Lighting Shield with LED strips connected

2 Driving LED Strip Lights

If the RGB LED Lighting Shield is used to drive LED strips that require constant voltage, its current control feature must be turned off. The input voltage to the shield has to be the same as what the strip requires, for example, 24V. The inductors and schottky diodes located on the shield are not used. The LED current is controlled by built-in logic in each LED strip.

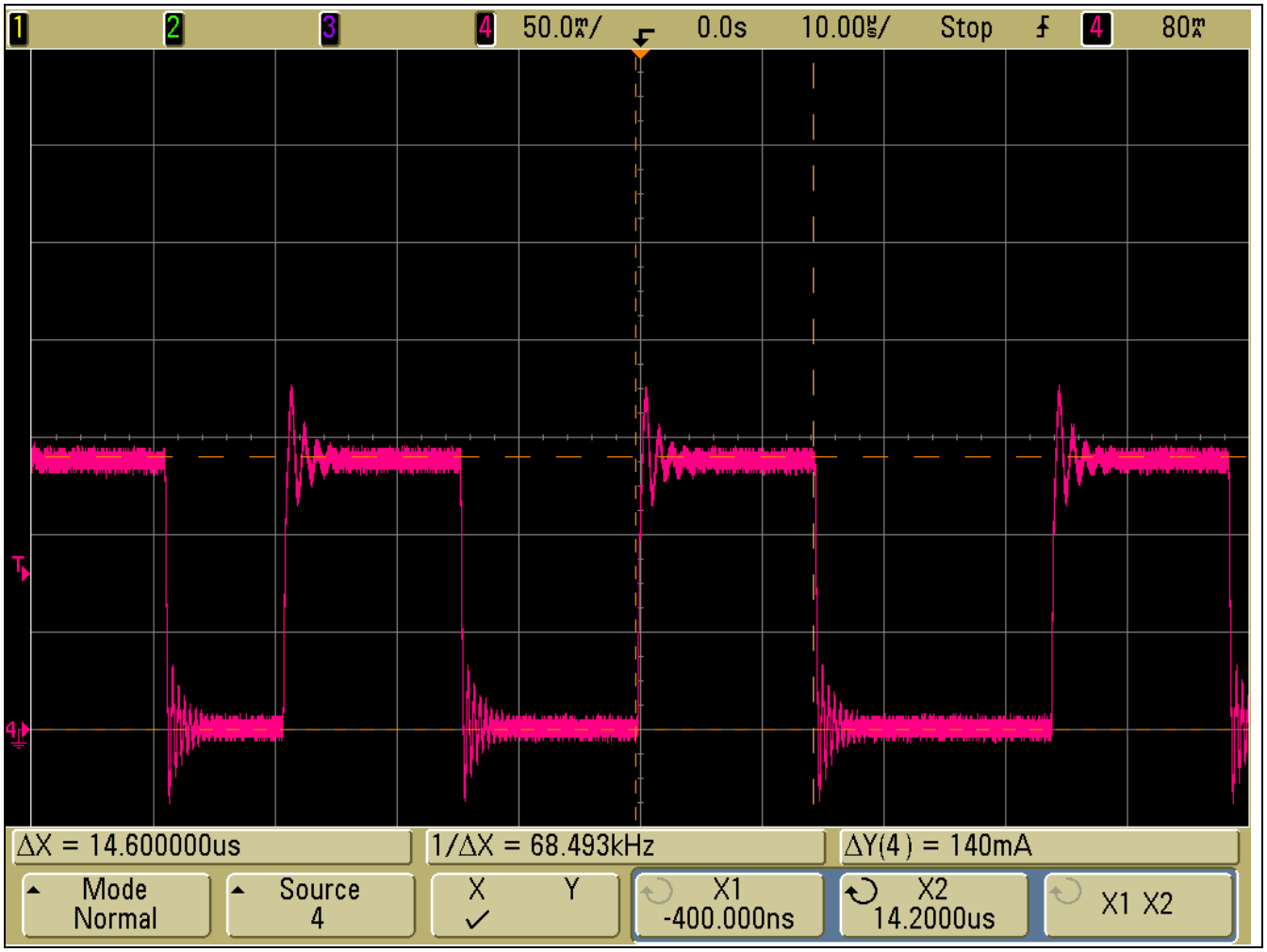

Figure 4 Typical modulation dimmed output current when LED strips are connected

The shield does not have to perform current control which means the onboard XMC1202 microcontroller has resources available for fast overcurrent protection. For example, it can shut down an LED strip within 200ns if the LED current exceeds 900mA (or any other user-defined value). Color and dimming level control are unchanged. They are performed by pulse density modulated signals from the BCCU module in the microcontroller and can be controlled by the same Colour Intensity and Dimming commands over I^2C.

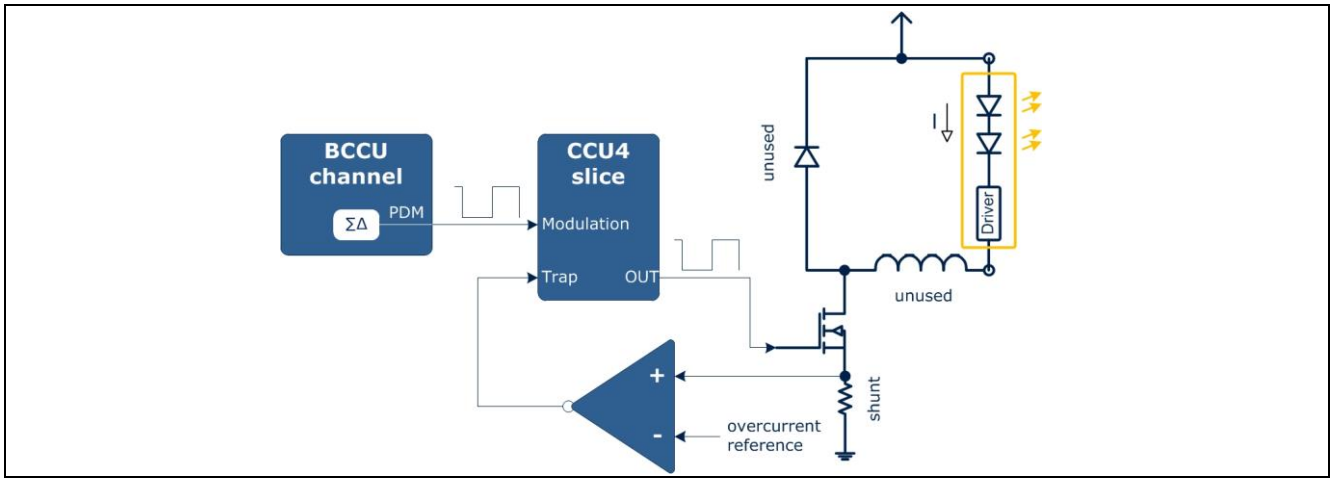

Figure 5 Modulation dimming and overcurrent protection

3 How to Reconfigure the Shield for LED Strips

Registers in the XMC1202 microcontroller on board the RGB LED Lighting Shield can be accessed directly without the use of the pre-defined I^2C commands. This is described in detail in the "RGB LED Lighting Shield with XMC1202 for Arduino" board manual. Information on the XMC1202 registers can be found in the XMC1200 Reference Manual.

Reconfiguring the registers requires 4 steps:

1. **Set current limit to the desired overcurrent level**
    a. The Peak Current Reference commands, such as CURRENT_RED, are limited to about 781mA.
    b. I2CWRITE_DIRECTACCESS command with DIRECTACCESS_MOVE parameter is recommended.
2. **Reprogram CCU4 inputs**
    a. When the LED current reaches the reference level it causes a TRAP event rather than a CLEAR event and the output channel is shut down to protect the circuitry.
    b. I2CWRITE_DIRECTACCESS command with DIRECTACCESS_MOVE parameter is recommended.
3. **Increase the free running duty cycle to 100%**
    a. The MOSFET should be always on when modulation enables it. There is no need for switched-mode current control.
    b. I2CWRITE_DIRECTACCESS command with DIRECTACCESS_MOVE parameter is recommended.
4. **Clear TRAP in all channels**
    a. This enables all channels.
    b. I2CWRITE_DIRECTACCESS command with DIRECTACCESS_MOVE parameter is recommended.

The following Arduino code snippet performs these 4 steps.

```
//-----------------------------------------------------------------------------------------------------------------------
    // Reconfigure RGB LED Lighting Shield for constant voltage LED load (e.g. LED strip)

    // set current limit to 900mA --> if the current goes above, the channel will be shut down immediately (TRAP)
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x5003003C, 148);          // BCCU0_INTS0 = 148; --> ~900mA
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x50030064, 148);          // BCCU0_INTS2 = 148; --> ~900mA
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x500300B4, 148);          // BCCU0_INTS6 = 148;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x50030018, 0x00000045);   // BCCU0_CHSTRCON = 0x00000045;

    // reprogram CCU4 inputs (no more peak-current control, overcurrent protection instead: TRAP)
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x48040200, 0x00100100);   // CCU40_CC41INS = 0x00100100;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x48040300, 0x00100100);   // CCU40_CC42INS = 0x00100100;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x48040400, 0x00100100);   // CCU40_CC43INS = 0x00100100;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x48040204, 0x00020000);   // CCU40_CC41CMC = 0x00020000;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x48040304, 0x00020000);   // CCU40_CC42CMC = 0x00020000;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x48040404, 0x00020000);   // CCU40_CC43CMC = 0x00020000;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x48040214, 0x03420100);   // CCU40_CC41TC  = 0x03420100;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x48040314, 0x03420100);   // CCU40_CC42TC  = 0x03420100;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x48040414, 0x03420100);   // CCU40_CC43TC  = 0x03420100;

    // 100% duty cycle (no off-time needs to be generated anymore)
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x4804023C, 0x00000000);   // CCU40_CC41CRS = 0;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x4804033C, 0x00000000);   // CCU40_CC42CRS = 0;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x4804043C, 0x00000000);   // CCU40_CC43CRS = 0;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x48040010, 0x00001110);   // CCU40_GCSS    = 0x00001110;

    // clear TRAP in all channels (in case it accidentally occurred while rewiring)
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x480402B0, 0x00000C00);   // CCU40_CC41SWR = 0x00000C00;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x480403B0, 0x00000C00);   // CCU40_CC42SWR = 0x00000C00;
    I2CWRITE_DIRECTACCESS (ADDRESS, DIRECTACCESS_MOVE, 0x480404B0, 0x00000C00);   // CCU40_CC43SWR = 0x00000C00;
//-----------------------------------------------------------------------------------------------------------------------
```

An example project is available at the "RGB LED Lighting Shield with XMC1202 for Arduino" website.

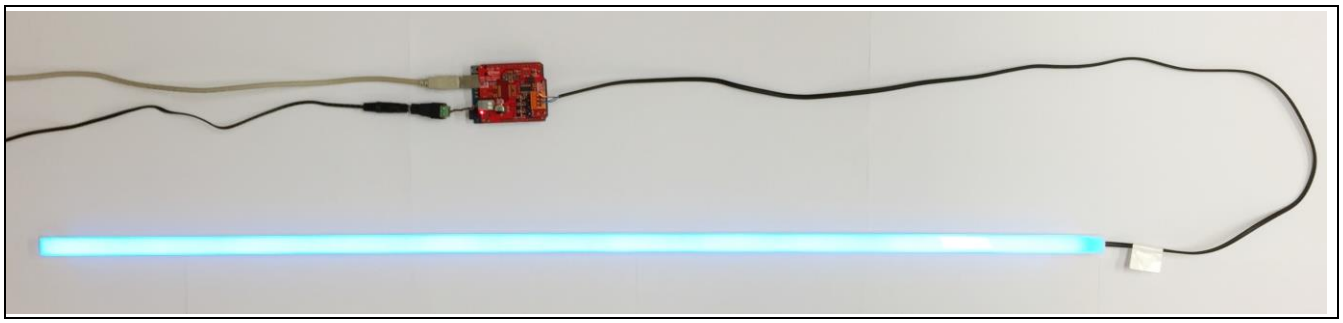

Figure 6 RGB LED Lighting Shield with LED Linear VarioLED Flex VENUS RGB

4 Revision History

Current Version is V1.0, 2015-08

Page or Reference	Description of change
V1.0, 2015-08	
	Initial Version

www.infineon.com

Edition 2015-08

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference

AP32313

Legal Disclaimer

THE INFORMATION GIVEN IN THIS APPLICATION NOTE (INCLUDING BUT NOT LIMITED TO CONTENTS OF REFERENCED WEBSITES) IS GIVEN AS A HINT FOR THE IMPLEMENTATION OF THE INFINEON TECHNOLOGIES COMPONENT ONLY AND SHALL NOT BE REGARDED AS ANY DESCRIPTION OR WARRANTY OF A CERTAIN FUNCTIONALITY, CONDITION OR QUALITY OF THE INFINEON TECHNOLOGIES COMPONENT. THE RECIPIENT OF THIS APPLICATION NOTE MUST VERIFY ANY FUNCTION DESCRIBED 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.

Information

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

Warnings

Due to technical requirements, components may contain dangerous substances. For information on the types in question, please contact the nearest Infineon Technologies Office. Infineon Technologies components may be used in life-support devices or systems only with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.