SMU_IR_Alarm_1 for KIT_AURIX_TC397_TFT

Interrupt triggered by an SMU alarm

AURIX™ TC3xx Microcontroller Training
V1.0.3

Scope of work

The SMU triggers an alarm, which has, as preconfigured reaction, an interrupt. The interrupt turns on an LED.

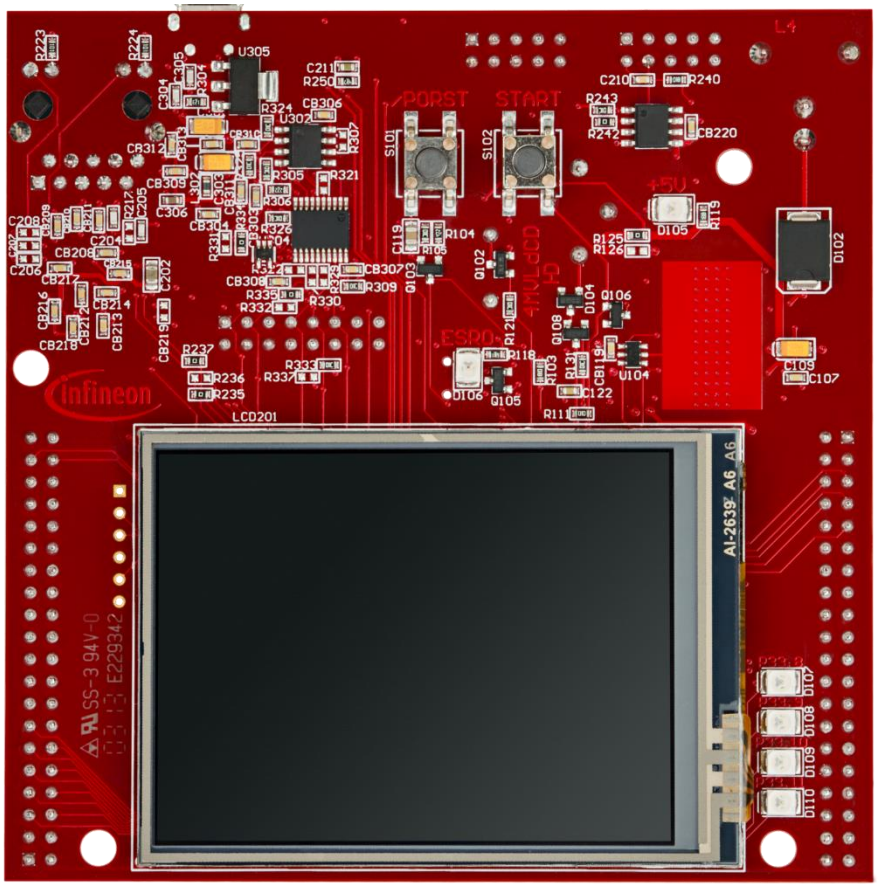
The Safety Management Unit (SMU) is configured to trigger an interrupt if an internal software alarm occurs. In case of an alarm, an LED will be turned on inside the Interrupt Service Routine.

Introduction

- The Safety Management Unit (SMU) is a central and modular component of the safety architecture, providing a generic interface to manage the behavior of the microcontroller under the presence of faults
- The SMU centralizes all the alarm signals related to the different hardware- and software-based safety mechanisms
- Each alarm can be individually configured to trigger internal or external actions
- The SMU, in combination with the embedded safety mechanisms, is able to detect and report more than 99% of the critical failure modes
- In this example, Software Alarm 0 is used to trigger an interrupt

Hardware setup

This code example has been developed for the board KIT_A2G_TC397_5V_TFT.



Implementation

Configure the SMU module

- To trigger an interrupt with an SMU alarm, a few steps are required:
  - To modify the SMU registers, the SMU module has to be unlocked with the function ***IfxSmu_unlockConfigRegisters()***. After the modification is finished, the SMU registers should be locked again using the function ***IfxSmu_lockConfigRegisters()***
  - Additionally, it is required to clear and set the Safety ENDINIT protection before and after the modification of the SMU configuration registers. This is done with the functions ***IfxScuWdt_clearSafetyEndinit()*** and ***IfxScuWdt_setSafetyEndinit()***
  - The Alarm Global Configuration register (**SMU_AGC**) provides the software interface to control how the SMU triggers interrupt requests to the interrupt router. By setting the **IGCS0** bitfield to **1**, SMU Interrupt Request 0 is triggered
  - The function ***IfxSmu_setAlarmAction()*** configures the alarm's behavior by writing a 3-bit code to the three Alarm Configuration Registers associated to the specific alarm and its group. In this example, the software alarm 0 (***IfxSmu_Alarm_Software_Alarm0)*** and the Interrupt Generation Configuration Set 0 (***IfxSmu_InternalAlarmAction_igcs0***) are selected. The iLLD function itself selects the group based on the above mentioned parameters
  - Configure and enable the SMU Service Request 0 with the functions ***IfxSrc_init()*** and ***IfxSrc_enable()***
  - Start the SMU state machine (SSM) with the function ***IfxSmu_activateRunState()***

The functions above are provided by the iLLD headers ***IfxSmu.h*** and ***IfxSrc.h***.

Implementation

LED configuration

- The port pin with the connected LED is configured to push-pull output mode by calling the function ***IfxPort_setPinMode()*** with the parameter ***IfxPort_Mode_outputPushPullGeneral*** (enumerated type value)

- With the function ***IfxPort_setPinState()***, using the enumerated type value ***IfxPort_State_high***, the LED is turned off as default state

All functions above are provided by the iLLD header ***IfxPort.h***.

Triggering of the alarm

- The Software Alarm 0 can be triggered with the function ***IfxSmu_setAlarmStatus()*** provided by the iLLD header ***IfxSmu.h***

Implementation

The Interrupt Service Routine (ISR)

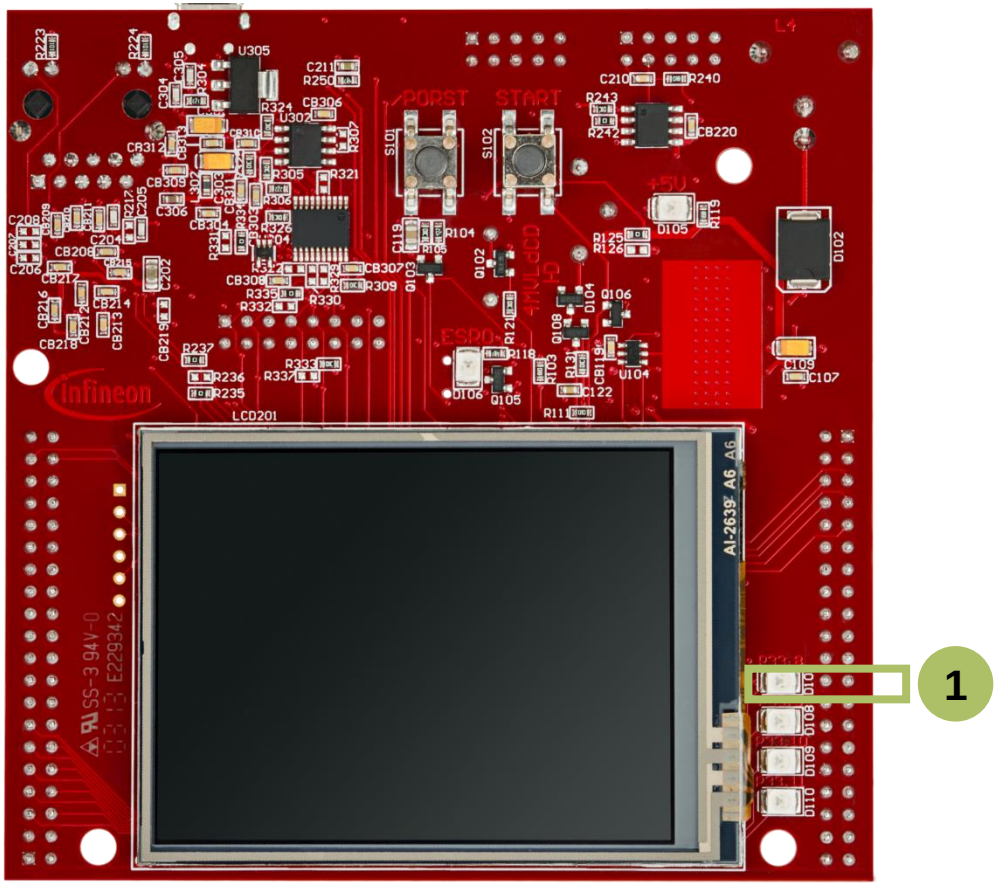
- The alarm status flag reset is implemented inside the ISR triggered by the SMU (***IfxSmu_clearAlarmStatus()***). Furthermore, the bit relative to the executed alarm mechanism in the Alarm Executed Status (AEX) register needs to be cleared, which is done by the function ***IfxSmu_clearAlarmExecutedStatus()***. These functions are provided by the iLLD header ***IfxSmu.h***

- The LED is turned on inside the ISR to indicate the successful configuration of the SMU and the triggering of the interrupt. This is done by setting the port pin of the connected LED by using the function ***IfxPort_setPinState()*** from the iLLD header ***IfxPort.h***

- The method implementing the ISR needs to be assigned a **CPU core** responsible for its execution done by the function ***IfxSrc_init()***. The method implementing the ISR needs to be assigned a **priority** via the macro ***IFX_INTERRUPT(isr, vectabNum, priority)***

Run and Test

After code compilation and flashing the device, check if LED D107 (1) is turned on.



References

› AURIX™ Development Studio is available online:
› https://www.infineon.com/aurixdevelopmentstudio
› Use the *„Import...“* function to get access to more code examples.

› More code examples can be found on the GIT repository:
› https://github.com/Infineon/AURIX_code_examples

› For additional trainings, visit our webpage:
› https://www.infineon.com/aurix-expert-training

› For questions and support, use the AURIX™ Forum:
› https://www.infineonforums.com/forums/13-Aurix-Forum

Revision history

Revision	Description of change
V1.0.3	Added function to clear AEX bit in the SMU ISR according to the code changes
V1.0.2	Corrected interrupt service routine initialization description
V1.0.1	Update of version to be in line with the code example’s version
V1.0.0	Initial version

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2021-09
Published by
Infineon Technologies AG
81726 Munich, Germany

Do you have a question about this document?
Email: erratum@infineon.com

Document reference
SMU_IR_Alarm_1_KIT_TC397_TFT

IMPORTANT NOTICE
The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie") .

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, 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.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

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.