SCU_Die_Temp_Sensor_1

Die Temperature Sensor

AURIX™ TC2xx Microcontroller Training
V1.0.0

Scope of work

The die temperature is measured by an internal sensor and printed on a terminal program.

The internal Die Temperature Sensor (DTS) is continuously read in an infinite loop. When a new temperature result is available, an interrupt service routine (ISR) is triggered. The ISR notifies the availability of a new die temperature value by setting a flag, which is used to start printing the temperature via UART communication using the ASCLIN module. The temperature value can be read by using a terminal program connected to the virtual COM port of the board/kit.

Introduction

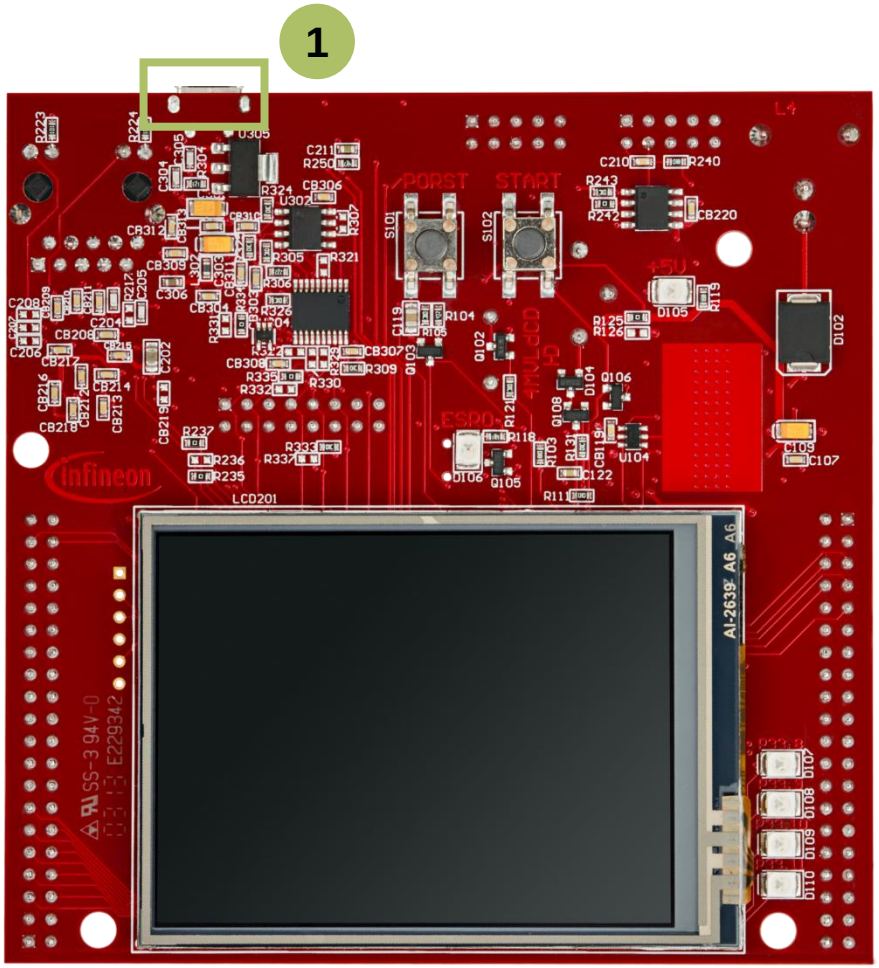
- The Die Temperature Sensor (DTS) is an internal sensor to measure the microcontroller's chip temperature.
  It generates a measurement value that indicates the current temperature of the die.

- Among other features, the DTS has the capability to **trigger an interrupt** when a new measurement is available.

Hardware setup

This code example has been developed for the board KIT_AURIX_TC297_TFT_BC-Step.

The board should be connected to the PC through the USB port (1).



Implementation

Configure the ASCLIN

Configuration of the ASCLIN module for UART communication is done in the setup phase by initializing an instance of the ***IfxAsclin_Asc_Config*** structure with the following parameters:

- ***baudrate*** – structure to set the actual communication speed in bit/s
- ***interrupt*** – structure to set:
  - interrupt priorities for transmit, receive and error events (***txPriority***, ***rxPriority*** and ***erPriority***)
  - ***typeOfService*** – defines which service provider is responsible for handling the interrupt, which can be any of the available CPUs, or the DMA
- ***pins*** – structure to set which GPIOs port pins are used for the communication
- ***rxBuffer***, ***rxBufferSize***, ***txBuffer***, ***txBufferSize*** – to configure the buffers that will hold the incoming/outgoing data

The function ***IfxAsclin_Asc_initModuleConfig()*** fills the configuration structure with default values and ***IfxAsclin_Asc_initModule()*** initializes the module with the user configuration. The standard interface is configured with the function ***IfxAsclin_Asc_stdIfDPipeInit()***.

All the above functions can be found in the iLLD header ***IfxAsclin_Asc.h***.

Implementation

Configure the DTS

Configuration of the DTS is done by initializing an instance of the ***IfxDts_Dts_Config*** structure, which contains the following fields:

- ***isrPriority*** – priority of the interrupt triggered by DTS when a new measurement is available (it can be a value from 0 to 255, with 0 meaning interrupt is disabled, and 255 is the highest priority)
- ***isrTypeOfService*** – defines which service provider is responsible for handling the interrupt, which can be any of the available CPUs, or the DMA
- ***lowerTemperatureLimit*** – to set the lower temperature limit for DTS measurements in Celsius
- ***upperTemperatureLimit*** – to set the upper temperature limit for DTS measurements in Celsius

An SMU alarm will be triggered if the measurement result is outside these limits.

The function ***IfxDts_Dts_initModuleConfig()*** fills the configuration structure with default values and ***IfxDts_Dts_initModule()*** function initializes the module with the user configuration.

Both functions are used in the DTS code section, and can be found in the iLLD header ***IfxDts_Dts.h***.

Implementation

Read measurements from the DTS

After the DTS configuration, a new measurement can be started through the function ***IfxDts_Dts_startSensor()***.

When a new measurement is available, an interrupt service request will be generated and the temperature can then be read with the function ***IfxDts_Dts_getTemperatureCelsius()***.

All functions used for the DTS measurements can be found in the iLLD header ***IfxDts_Dts.h***.

Run & Test

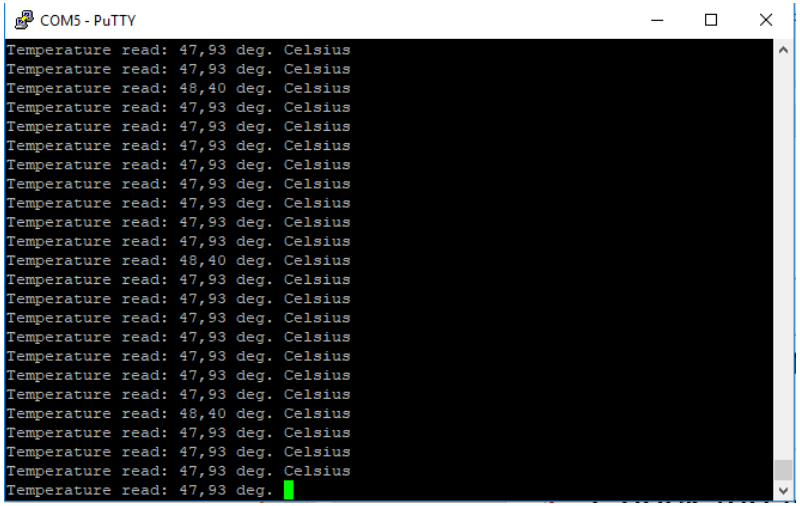
In order to see the DTS results, a terminal program (e.g. PuTTY, HTerm) is required. The terminal program must be configured with the following parameters to enable the communication between the board and the PC:

- **Port:** COMx (e.g. COM8, COM port number depends on the setup)
- **Baud rate**: 115200
- **Data bits**: 8
- **Parity**: None
- **Stop bits**: 1
- **Handshaking**: None

Additionally, the option AURIX C compiler >Floating-Point>Treat ‘double’ as ‘float’ must/should be set.

After code compilation and flashing the device, perform the following steps:

› Open a terminal program with the above configuration and connect.
› Check the temperature measurements.



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

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

Edition 2019-10
Published by
Infineon Technologies AG
81726 Munich, Germany

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

Document reference
SCU_Die_Temp_Sensor_1

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.