# ADC_Queued_Scan_1 for KIT_AURIX_TC334_LK

## ADC queued source

AURIX™ TC3xx Microcontroller Training
V1.0.0

# Scope of work

**The Enhanced Versatile Analog-to-Digital Converter (EVADC) is configured to measure multiple analog signals in a sequence using queued request.**

The Queued Request of the Enhanced Versatile Analog-to-Digital Converter (EVADC) module is used to continuously scan the analog inputs channels 7, 6 and 5 of group 8.

# Introduction

- The Enhanced Versatile Analog-to-Digital Converter module (EVADC) of the AURIX™ TC33x comprises 4 independent analog to digital converters (EVADC groups) with up to 16 analog input channels each
- Each channel can convert analog inputs with a resolution of up to 12-bit
- Analog/Digital conversions can be requested by several request sources:
  - **Queued request source**, specific to a single group
  - **Synchronization source**, synchronized conversion request from another ADC master kernel
- A queued source can issue conversion requests for an arbitrary sequence of input channels. The channel numbers for this sequence can be freely programmed
- The trigger for the conversion via the queued source can be sent:
  - Once (by another external module)
  - On a regular time base (by an external timer)
  - Permanently (by using the refill option)

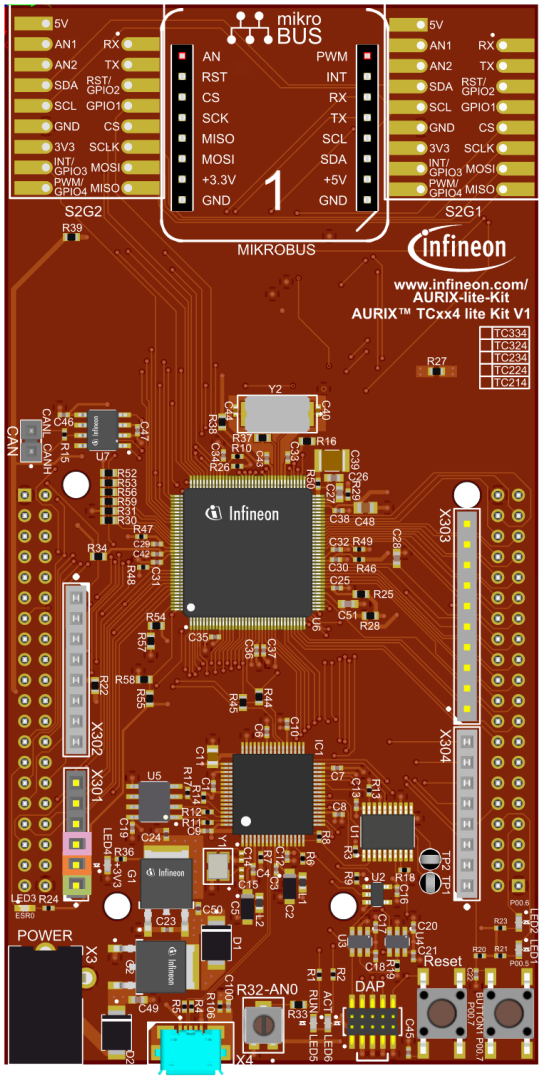
# Hardware setup

This code example has been developed for the board KIT_A2G_TC334_LITE.

The signals to be measured have to be connected to channels 7, 6 and 5 of the group 8 of the EVADC (pins AN39, AN38, AN37).

| | X301 |
|---|---|
| 5 | 6 |
| 4 | 5 |
| 6 | 4 |
| 7 | 3 |
| 8 | 2 |
| 9 | 1 |

**Note:** The reference voltage (VAREF) of the EVADC on the board KIT_A2G_TC334_LITE is 3.3 V.
**Note**: The channels can be HW filtered by the board, depending on which capacitor/resistors couples are soldered. Consult the AURIX™ TC334 lite Kit's User Manual to check which channels are filtered by HW.



# Implementation

**Configuration of the EVADC**

The configuration of the EVADC is done in the ***initEVADC()*** function in four different steps:

- Configuration of the **EVADC module**
- Configuration of the **EVADC group**
- Configuration of the **EVADC channels**
- Filling the queue

**Configuration of the EVADC module with the function *initEVADCModule()***

The default configuration of the EVADC module, given by the iLLDs, can be used for this example.
This is done by initializing an instance of the ***IfxEvadc_Adc_Config*** structure and applying default values to its fields through the function ***IfxEvadc_Adc_initModuleConfig()***.
Then, the configuration can be applied to the EVADC module with the function ***IfxEvadc_Adc_initModule()***.

# Implementation

**Configuration of the EVADC group with the function *initEVADCGroup()***

The configuration of the EVADC group is done by initializing an instance of the ***IfxEvadc_Adc_GroupConfig*** structure with default values through the function ***IfxEvadc_Adc_initGroupConfig()*** and modifying the following fields:

- ***groupId*** – to select which converters to configure
- ***master*** – to indicate which converter is the master. In this example, only one converter is used, therefore it is also the master
- ***arbiter*** – a structure that represents the enabled request sources. In this example, it is set to ***arbiter.requestSlotQueue0Enabled***
- ***triggerConfig*** – a parameter that specify the trigger configuration

Then, the user configuration is applied through the function ***IfxEvadc_Adc_initGroup()***.

# Implementation

**Configuration of the EVADC channels with the function *initEVADCChannels()***

The configuration of each channel is done by initializing a separate instance of the ***IfxEvadc_Adc_ChannelConfig*** structure with default values through the function ***IfxEvadc_Adc_initChannelConfig()*** and modifying the following fields:

- ***channelId*** – to select the channel to configure
- ***resultRegister*** – to indicate the register where the A/D conversion value is stored

Then, the configuration is applied to the channel with the function ***IfxEvadc_Adc_initChannel()***.

**Filling the queue**

Each channel is added to the queue through the function ***IfxEvadc_Adc_addToQueue()***.

When the EVADC configuration is done and the queue is filled, the conversion is started with the function ***IfxEvadc_Adc_startQueue()***.

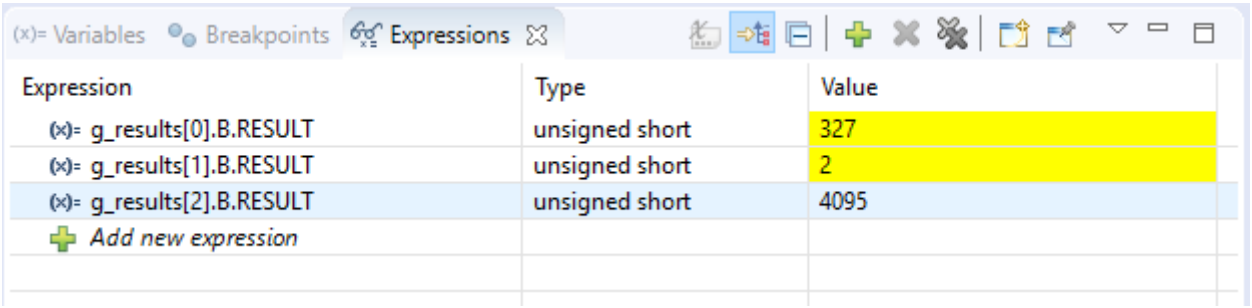
To read a conversion, the iLLD API ***IfxEvadc_Adc_getResult()*** is used inside the function ***readEVADC()***.

All the functions used for configuring the EVADC module, its groups and channels together with reading the conversion results can be found in the iLLD header ***IfxEvadc_Adc.h***.

# Run and Test

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

- Run the code and then pause it
- Repeat step number one to see that the result is changing accordingly to the signal you measure, AN39 is ***g_results[0]***, AN38 is ***g_results[1]*** and AN37 is ***g_results[2]***.




**Note:** The maximum and minimum values are expressed as a 12-bits integer value, in decimal format (0 - 4095 range), 3.3V being the maximum measurable value.

# 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 2021-12**
**Published by**
**Infineon Technologies AG**
**81726 Munich, Germany**



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

**Document reference**
**ADC_Queued_Scan_1_KIT_TC334_LK**

**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.