Document Title	Requirements on ADC Driver
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	111

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R25-11

Document Change History

Date	Release	Changed by	Description
2025-11-27	R25-11	AUTOSAR Release Management	• No content changes
2024-11-27	R24-11	AUTOSAR Release Management	• No content changes
2023-11-23	R23-11	AUTOSAR Release Management	• No content changes
2022-11-24	R22-11	AUTOSAR Release Management	• No content changes
2021-11-25	R21-11	AUTOSAR Release Management	• No content changes
2020-11-30	R20-11	AUTOSAR Release Management	• No content changes
2019-11-28	R19-11	AUTOSAR Release Management	• No content changes • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• Editorial changes
2017-12-08	4.3.1	AUTOSAR Release Management	• Scope of ADC driver clarified • References to not applicable documents removed • Editorial changes

△

2016-11-30	4.3.0	AUTOSAR Release Management	• Chapter 5 added with requirements tracing table
2015-07-31	4.2.2	AUTOSAR Release Management	• Editorial changes
2014-10-31	4.2.1	AUTOSAR Release Management	• Editorial changes
2013-10-31	4.1.2	AUTOSAR Release Management	• Editorial changes
2013-03-15	4.1.1	AUTOSAR Administration	• ECU degradation concept added • Formal updates to TPS standardization template • Traceability of BSWAndRTE_Features
2010-02-02	3.1.4	AUTOSAR Administration	• SRS_Adc_12824 : Result alignment changed from right-alignment to configurable alignment • Legal disclaimer revised
2008-08-13	3.1.1	AUTOSAR Administration	• Legal disclaimer revised
2007-12-21	3.0.1	AUTOSAR Administration	• SRS_Adc_12821 : SRS_Adc_12821 now valid for all channel groups independent of the mode selected. • SRS_Adc_12820 : Queuing for sequential requests for the same group removed. • SRS_Adc_12280 : Single value results access mode removed; ADC result buffer access possible for all ADC channel groups; access to last group conversion results added • SRS_Adc_12822 : Tailoring on largest channel in any group. • SRS_Adc_12291 : add state 'Stream Completed' • Document meta information extended • Small layout adaptations made

▽

△

2007-01-24	2.1.15	AUTOSAR Administration	• "Advice for users" revised • "Revision Information" added
2006-11-28	2.1	AUTOSAR Administration	• Removed requirement for "On Demand" functionality. • Removed requirement for "Gated Continuous" conversion mode. • Removed the distinction between internal and external hardware trigger. • Introduced a requirement for a priority mechanism for channel groups for allowing channel groups with higher priority to interrupt ongoing conversions. • Legal disclaimer revised
2006-05-16	2.0	AUTOSAR Administration	• Release as a separate document. The SRS SPAL V1.0.0 has been split into 12 independent documents for Release 2.0: • Gated continuous conversion mode rqmts added. • On Demand conversion mode rqmts added. • Other minor changes.
2005-05-31	1.0	AUTOSAR Administration	• Initial release as a part of the SRS SPAL V1.0.0.

Disclaimer

Table of Contents

1 Scope of Document

This document specifies requirements on the module ADC Driver. The ADC driver is targeting Successive Approximation ADC Hardware. Delta Sigma ADC conversion requirements are out of scope of this document.

Constraints

First scope for specification of requirements on basic software modules are systems which are not safety relevant. For this reason safety requirements are assigned to medium priority.

2 How to Read this Document

Each requirement has its unique identifier starting with the prefix "BSW" (for "Basic Software"). For any review annotations, remarks or questions, please refer to this unique ID rather than chapter or page numbers!

2.1 Document Conventions

The representation of requirements in AUTOSAR documents follows the table specified in [TPS_STDT_00078], see [1, Standardization Template].

The verbal forms for the expression of obligation specified in [TPS_STDT_00053] shall be used to indicate requirements, see [1, Standardization Template].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as follows.

Note that the requirement level of the document in which they are used modifies the force of these words.

- MUST: This word, or the adjective "LEGALLY REQUIRED", means that the definition is an absolute requirement of the specification due to legal issues.
- MUST NOT: This phrase, or the phrase "MUST NOT", means that the definition is an absolute prohibition of the specification due to legal issues.
- SHALL: This phrase, or the adjective "REQUIRED", means that the definition is an absolute requirement of the specification.
- SHALL NOT: This phrase means that the definition is an absolute prohibition of the specification.
- SHOULD: This word, or the adjective "RECOMMENDED", means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
- SHOULD NOT: This phrase, or the phrase "NOT RECOMMENDED", means that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
- MAY: This word, or the adjective "OPTIONAL", means that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item.

An implementation, which does not include a particular option, SHALL be prepared to interoperate with another implementation, which does include the option, though perhaps with reduced functionality. In the same vein an implementation, which does include a particular option, SHALL be prepared to interoperate with another implementation, which does not include the option (except, of course, for the feature the option provides).

2.2 Requirements Structure

Each module specific chapter contains a short functional description of the Basic Software Module. Requirements of the same kind within each chapter are grouped under the following headlines (where applicable):

Functional Requirements:

- Configuration (which elements of the module need to be configurable).
- Initialization.
- Normal Operation.
- Shutdown Operation.
- Fault Operation.
- ...

Non-Functional Requirements:

- Timing Requirements.
- Resource Usage.
- Usability.
- Output for other WPs (e.g. Description Templates, Tooling,...).
- ...

3 Acronyms and Abbreviations

Acronyms and abbreviations that have a local scope are not contained in the AUTOSAR glossary. These must appear in a local glossary.

Acronym:	Description:
CS	Chip select
DIO	Digital Input Output
ECU	Electric Control Unit
EOL	End Of Line Often used in the term 'EOL Programming' or 'EOL Configuration'
ICU	Input Capture Unit
MAL	Old name of Microcontroller Abstraction Layer (replaced by MCAL because 'MAL' is a French term meaning 'bad')
MCAL	Microcontroller Abstraction Layer
MCU	Microcontroller Unit
MMU	Memory Management Unit
Master	A device controlling other devices (slaves, see below)
Slave	A device being completely controlled by a master device
NMI	Non maskable interrupt
OS	Operating System
PLL	Phase Locked Loop
PWM	Pulse Width Modulation
RX	Reception (in the context of bus communication)
SPAL	The name of this working group
SFR	Special Function Register
RTE	Runtime environment
WP	Work Package
ADC	Analogue Digital Converter
FFT	Fast Fourier Transformer

Abbreviation:	Description:
STD	Standard
REQ	Requirement
UNINIT	Uninitialized (= not initialized)

The following expressions are used within the ADC driver:

Expression	Explanation
HW Unit	Represents a microcontroller input electronic device that includes all parts necessary to perform an "analogue to digital conversion".
ADC channel	Represents a logical ADC entity bound to one port pin. Multiple ADC entities can be mapped to the same port pin.
ADC channel group	A group of ADC channels linked to the same ADC hardware unit (e.g. one Sample&Hold and one A/D converter). The conversion of the whole group is triggered by one trigger source.

▽

△

Expression	Explanation
ADC result buffer	The user of the ADC Driver has to provide a buffer for every group. This buffer can hold multiple samples of the same channel group if streaming access mode is selected. If single access mode is selected one sample of each group channel is held in the buffer.
Trigger Source	Source event that starts a single conversion or a continuous series of conversions.
Conversion Mode	One-Shot: The conversion of an ADC channel group is performed once after a trigger and the result is written to the assigned buffer. A trigger can be a software API call or a hardware event. Continuous: The conversions of an ADC channel group are performed continuously after a software API call (start). The conversions themselves are running automatically (hardware/interrupt controlled). The Continuous conversions can be stopped by a software API call (stop).
Sampling Time	Time during which the analogue value is sampled (e.g. loading the capacitor, ...)
Conversion Time	Time during which the sampled analogue value is converted into digital representation.
Acquisition Time	Sample Time + Conversion Time.

As this is a document from professionals for professionals, all other terms are expected to be known.

4 Requirements Specification

4.1 Functional Overview

The ADC driver initializes and controls the internal Analogue Digital Converter unit(s) of the microcontroller. It provides services to start and stop a conversion respectively to enable and disable the trigger source of a conversion. Furthermore, it provides services to enable and disable a notification mechanism and routines to query the status and result of a conversion.

The ADC Driver shall work on so called ADC Channels. An ADC channel combines an analogue input pin, the needed ADC circuitry itself and a conversion result register into an entity that can be individually controlled and accessed via the ADC Driver.

All used terms are specified in the following chapter.

4.2 Functional Requirements

4.2.1 Configuration

[SRS_Adc_12307] The ADC Driver shall support a specific basic static configurations per channel ⌈

Description:	The ADC Driver shall support the following basic static configurations per channel (if supported by HW): • Symbolic name for channel. • Sampling time. • Conversion time. • Resolution in bits. • Reference voltage source. • Clock source with optional prescaler settings. • Other MCU dependent parameters. Note: If one or more of these configuration parameters can only be assigned to the whole module, it's on the configuration tool to optimize data representation.
Rationale:	To allow different usage for each channel.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Adc_12447] The ADC Driver shall allow to group ADC channels that belong to the same ADC HW unit ⌈

Description:	The ADC Driver shall allow to group ADC channels that belong to the same ADC HW unit into so called ADC channel groups (or groups for short). An ADC Channel Group shall consist of at least one ADC Channel. All channels of a group share the same group configuration. The ADC Driver shall support the following basic static configurations per channel group (if supported by HW): • Symbolic name for group. • Group notification on/off. • Callback notification function. • List of channels configured for use. • Group trigger source. • Group conversion mode.
Rationale:	Allow sampling of multiple channels using only one control thread.
Use Case:	Grouping of several channels where the conversion results shall be consistent.
Dependencies:	[SRS_Adc_12307] ADC Channel Configuration.
Supporting Material:	–

⌋

[SRS_Adc_12817] The ADC Driver shall allow for each ADC channel group the static configuration of exactly one trigger source ⌈

Description:	The ADC Driver shall allow for each ADC channel group the static configuration of exactly one trigger source. Possible trigger sources are: • Hardware event (e.g. on an IO pin or a timer interrupt generated MCU internal), if supported by HW. • SW API call.
Rationale:	To make sure that one group is only controlled from one source (avoid inconsistencies, concurrency issues and so forth).
Use Case:	A group of ADC channels configured to be converted as reaction to an external interrupt on an IO pin.
Dependencies:	[SRS_Adc_12447] ADC channel group configuration.
Supporting Material:	–

⌋

[SRS_Adc_12818] The ADC Driver shall allow assigning one ADC channel to more than one ADC Channel Group ⌈

Description:	The ADC Driver shall allow assigning one ADC channel to more than one ADC Channel Group.
Rationale:	This allows for different trigger sources for an ADC Channel if it is configured for different groups.
Use Case:	Same ADC channel usually converted cyclically (per HW event, in the background) can be converted once per SW API call.
Dependencies:	–
Supporting Material:	Note: The ADC Driver shall not handle any problems arising from concurrent operation of the affected groups. The user of the ADC Driver has to ensure that there are no such concurrent operations of the affected groups.

⌋

[SRS_Adc_12821] For all channel groups, the ADC driver shall provide at configuration time the possibility to define a buffer for storing samples ⌈

Description:	For all channel groups, the ADC driver shall provide at configuration time the possibility to define a buffer for storing samples. As the result, the following parameters shall be added to the ADC channel group parameters: • Pointer to data buffer (destination for conversion results). • Number of ADC values.
Rationale:	Conversion results for all ADC channel groups are stored in external buffers.
Use Case:	Cyclic AD conversions as input for electric motor controller.
Supporting Material:	–

⌋

[SRS_Adc_12820] The ADC driver shall allow the configuration of a priority level for each channel group ⌈

Description:	The ADC driver shall allow the configuration of a priority level for each channel group. This implies a prioritization mechanism, implemented in SW (or where available supported by the HW), which offers following features: • Queuing of requests for different groups. • Priority handling (aborting/suspending and restarting/resuming conversions). • Higher priority groups can abort/suspend lower priority groups. • Priorities 0..255, lowest priority 0.
Rationale:	Give the opportunity to a group to be immediately converted (on demand) by interrupting an already ongoing conversion. Due to the possibility of nested request for conversion it is necessary to define a priority to solve the sequence in which the driver will manage them.

▽

△

Use Case:	Triggered conversions are more and more used in Powertrain controllers (one example is the measurement of injector bank currents/voltages which need to be done synchronously with the fuel pulses - while there are auto-scan conversions ongoing in background).
Dependencies:	[SRS_Adc_12447], [SRS_Adc_12818]
Supporting Material:	–

⌋

[SRS_Adc_12280] The ADC Driver shall allow a specific result access modes for each ADC Channel Group ⌈

Description:	The ADC Driver shall allow the following result access modes for each ADC Channel Group: • Result access to the ADC result buffer. The ADC result buffer contains in streaming access mode the streaming results and in single access mode the results of the last group conversion. • If statically configured, a result access to the results of the last group conversion shall be supported, also in streaming access mode. The group channel results are stored in the buffer in ascending order, in contrast to the buffer layout of the ADC result buffer in streaming access mode.
Rationale:	Every channel value shall be accessible until the next scan of the channel group is completed.
Use Case:	–
Dependencies:	[SRS_Adc_12447] ADC channel group configuration.
Supporting Material:	–

⌋

4.2.2 Normal Operation

[SRS_Adc_12283] The ADC driver shall mask out information bits from the conversion result not belonging to the ADC value ⌈

Description:	The ADC driver shall mask out information bits from the conversion result not belonging to the ADC value.
Rationale:	Information bits are microcontroller specific.
Use Case:	Some μCs are saving information bits in the result register (e.g. the channel number). These bits shall be masked out.
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Adc_12824] The result alignment shall be configurable between right-alignment and left-alignment ⌈

Description:	The result alignment shall be configurable between right-alignment and left-alignment.
Rationale:	Information bits are microcontroller specific. Code and runtime optimization in driver and upper layer.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Adc_12819] The ADC Driver shall provide a synchronous service for reading the last valid conversion results of the selected channel group ⌈

Description:	The ADC Driver shall provide a synchronous service for reading the last valid conversion results of the selected channel group (passed as parameter).
Rationale:	Provide access to the most recent converted values of a channel group.
Use Case:	This is the standard functionality for reading ADC values.
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Adc_12822] The structure containing the results of a channel group conversion shall be generated with a uniform dimension ⌈

Description:	The structure containing the results of a channel group conversion shall be generated by the configuration tool with a uniform dimension, tailored on the largest (in number of bits) channel belonging to any group.
Rationale:	Channels within one group may have different sized result values (e.g. 8 bit, 10 bit, 12 bit, ...).
Use Case:	–
Dependencies:	[SRS_Adc_12447] ADC channel group configuration.
Supporting Material:	–

⌋

[SRS_Adc_12317] The ADC Driver shall provide notification functions to inform the caller about the end of a conversion for a Channel Group ⌈

Description:	The ADC Driver shall provide notification functions to inform the caller about the end of a conversion for a Channel Group.
Rationale:	Allow for non blocking access to the ADC.

▽

△

Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Adc_12291] The ADC Driver shall provide a service for querying the status of an ADC Channel Group ⌈

Description:	The ADC Driver shall provide a service for querying the status of an ADC Channel Group. The ADC channel group status shall be one out of following four: • Idle. • Busy (conversion ongoing). • Completed. • Stream Completed.
Rationale:	Before reading a value it might be necessary to query the status of the ADC.
Use Case:	The status can be used for access synchronization.
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Adc_12318] The ADC driver shall provide a service to enable and disable each notification function separately ⌈

Description:	The ADC driver shall provide a service to enable and disable each notification function separately.
Rationale:	Prevents from invoking unwanted notifications (interrupts) and allows to select the exact point in the execution flow of the software when the first or next notification may arise.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Adc_12364] The ADC driver shall provide services to start and stop the conversion of an ADC Channel Group for all conversion modes ⌈

Description:	The ADC driver shall provide services to start and stop the conversion of an ADC Channel Group for all conversion modes.
Rationale:	Allow for software control of the start time of conversion of a group of channels.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Adc_12292] If the ADC provides signed values, the ADC driver shall put the sign bit into the MSB of the return value ⌈

Description:	If the ADC provides signed values, the ADC driver shall put the sign bit into the MSB of the return value.
Rationale:	Allow mapping of ADC return values to standard AUTOSAR integer data types.
Use Case:	Some microcontrollers provide signed ADC values. Example: ADC register with 12 bit, sign bit in bit 11 (in a microcontroller with a "Big Endian" byte ordering).
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Adc_12288] Based on the channel group configuration the ADC driver shall be able to handle the buffers of stream jobs ⌈

Description:	Based on the channel group configuration the ADC driver shall be able to handle the buffers of stream jobs in two different way: • As "linear buffers", i.e. the ADC driver stops the conversion as soon as the stream buffer is full (number of samples reached). • As "circular buffers", i.e. the ADC driver continue the conversion even if the stream buffer is full (number of samples reached) by wrapping around the buffer itself.
Rationale:	Allow cyclic streams of ADC conversion that run automatically.
Use Case:	Electric motor control.
Dependencies:	[SRS_Adc_12280] ADC channel group results access mode.
Supporting Material:	–

⌋

[SRS_Adc_12802] The ADC driver shall provide (for streaming access mode) a service to identify most recent sample and number of available samples of a channel group ⌈

Description:	The ADC driver shall provide (for streaming access mode) a service to identify most recent sample and number of available samples of a channel group. Passed parameters shall be: • ADC channel group. Returned objects shall be: • Number of samples. • Pointer to last sample.
Rationale:	To identify which samples can be used.
Use Case:	Digital knock sensor acquisition.
Dependencies:	–
Supporting Material:	–

⌋

[SRS_Adc_12823] The ADC driver shall provide services to enable and disable HW triggers for each channel group ⌈

Description:	The ADC driver shall provide services to enable and disable HW triggers for each channel group.
Rationale:	Give the possibility to ignore upcoming hardware triggers at run-time.
Use Case:	–
Dependencies:	[SRS_Adc_12447], [SRS_Adc_12817]
Supporting Material:	–

⌋

[SRS_Adc_12825] The results of the conversion of a channel group configured in streaming access mode shall be returned into a buffer with a fixed number of elements ⌈

Description:	The results of the conversion of a channel group configured in streaming access mode shall be returned into a buffer of n*m elements (integers), where n is the number of channels belonging to the group, m the number of samples acquired per channel. In this way the first m elements belong to the first channel in the group, the second m elements to the second channel and so on.
Rationale:	Support channel group streaming conversion mode.
Use Case:	–
Dependencies:	[SRS_Adc_12280] ADC channel group results access mode.

▽

△

Supporting Material:	–

⌋

[SRS_Adc_12826] The ADC driver shall implement an API which allows to read the current power state of the ADC HW module ⌈

Description:	The ADC driver shall implement an API which allows to read the current power state of the HW perifpheral from outside of the MCAL module. This API can be used by IoHwAbs and CDD SW Components.
Rationale:	It must be possible to gather information on the current power state of a peripheral; this is useful to assert the actual state of the peripheral e.g. in order to know what operations are supported by the peripheral in a given moment or to decide in which power state it should be set.
Use Case:	–
Dependencies:	–
Supporting Material:	Concept "ECU Degradation", V1.8

⌋

[SRS_Adc_12827] The ADC driver shall implement an API which allows to read the target power state of the ADC HW module ⌈

Description:	The ADC driver shall implement an API which allows to read the target power state of the HW perifpheral from outside of the MCAL module. This API can be used by IoHwAbs and CDD SW Components.
Rationale:	It is necessary to get information on the target power state in order to understand if a power state transition is being executed and, in positive case, which one.
Use Case:	–
Dependencies:	–
Supporting Material:	Concept "ECU Degradation", V1.8

⌋

[SRS_Adc_12828] The ADC Driver shall separate the power state transition sequence in two parts ⌈

Description:	The ADC Driver shall separate the power state transition sequence in two parts, a preparation phase (preliminary configuration changes to allow the peripheral to enter the target power state) and a setting phase (effective enabling of the validpower state).

▽

△

Rationale:	Some peripherals may take more time than others to execute all preliminary transitions needed to get into a given power state. Moreover there could be some HW-related dependency between module, requiring to set a different peripheral in a specific power state or HW configuration before being able to transition the target peripheral. By splitting the process in a preparation and a setting phase, it is possible to synchronize different peripherals which than enter their validpower state all at the same time and it is also possible to coordinate HW state intermediate transitions.
Use Case:	–
Dependencies:	–
Supporting Material:	Concept "ECU Degradation", V1.8

⌋

[SRS_Adc_12829] It shall be possible to configure a synchnous or an asynchronous power state transition behavior ⌈

Description:	It shall be possible to configure a synchnous or an asynchronous power state transition behavior. In case a synchronous behavior is configured, the power transition preparation shall happen atomically and the SWC executing it must wait on the result. In case an asynchronous behavior is configured, the power transition preparation shall progress in background once requested and the MCAL module shall notify the registered SWC upon completion.
Rationale:	Some peripheral can be prepared to the validpower state in a negligible or acceptable time, which allows to spare the infrastructure needed to handle notifications, while blocking the caller for the time needed by the preparation. Some other peripheral could require some lengthier time to be prepared for the target power state, in this case the SWCs which expect a notification can instruct different modules to prepare for power state and go on with its execution, until all notifications are issued by the involved MCAL module.
Use Case:	–
Dependencies:	–
Supporting Material:	Concept "ECU Degradation", V1.8

⌋

5 References

[1] Standardization Template
AUTOSAR_FO_TPS_StandardizationTemplate

A Change history of AUTOSAR traceable items

Please note that the lists in this chapter also include traceable items that have been removed from the specification in a later version. These items do not appear as hyperlinks in the document.

A.1 Traceable item history of this document according to AUTOSAR Release R25-11

A.1.1 Added Requirements in R25-11

none

A.1.2 Changed Requirements in R25-11

none

A.1.3 Deleted Requirements in R25-11

none

A.2 Traceable item history of this document according to AUTOSAR Release R24-11

A.2.1 Added Requirements in R24-11

none

A.2.2 Changed Requirements in R24-11

none

A.2.3 Deleted Requirements in R24-11

none

A.3 Traceable item history of this document according to AUTOSAR Release R23-11

A.3.1 Added Requirements in R23-11

none

A.3.2 Changed Requirements in R23-11

none

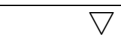
A.3.3 Deleted Requirements in R23-11

none

A.4 Traceable item history of this document according to AUTOSAR Release R22-11

A.4.1 Added Requirements in R22-11

Number	Heading
[SRS_Adc_12280]	The ADC Driver shall allow a specific result access modes for each ADC Channel Group
[SRS_Adc_12283]	The ADC driver shall mask out information bits from the conversion result not belonging to the ADC value
[SRS_Adc_12288]	Based on the channel group configuration the ADC driver shall be able to handle the buffers of stream jobs
[SRS_Adc_12291]	The ADC Driver shall provide a service for querying the status of an ADC Channel Group
[SRS_Adc_12292]	If the ADC provides signed values, the ADC driver shall put the sign bit into the MSB of the return value
[SRS_Adc_12307]	The ADC Driver shall support a specific basic static configurations per channel
[SRS_Adc_12317]	The ADC Driver shall provide notification functions to inform the caller about the end of a conversion for a Channel Group
[SRS_Adc_12318]	The ADC driver shall provide a service to enable and disable each notification function separately
[SRS_Adc_12364]	The ADC driver shall provide services to start and stop the conversion of an ADC Channel Group for all conversion modes
[SRS_Adc_12447]	The ADC Driver shall allow to group ADC channels that belong to the same ADC HW unit



△

Number	Heading
[SRS_Adc_12802]	The ADC driver shall provide (for streaming access mode) a service to identify most recent sample and number of available samples of a channel group
[SRS_Adc_12817]	The ADC Driver shall allow for each ADC channel group the static configuration of exactly one trigger source
[SRS_Adc_12818]	The ADC Driver shall allow assigning one ADC channel to more than one ADC Channel Group
[SRS_Adc_12819]	The ADC Driver shall provide a synchronous service for reading the last valid conversion results of the selected channel group
[SRS_Adc_12820]	The ADC driver shall allow the configuration of a priority level for each channel group
[SRS_Adc_12821]	For all channel groups, the ADC driver shall provide at configuration time the possibility to define a buffer for storing samples
[SRS_Adc_12822]	The structure containing the results of a channel group conversion shall be generated with a uniform dimension
[SRS_Adc_12823]	The ADC driver shall provide services to enable and disable HW triggers for each channel group
[SRS_Adc_12824]	The result alignment shall be configurable between right-alignment and left-alignment
[SRS_Adc_12825]	The results of the conversion of a channel group configured in streaming access mode shall be returned into a buffer with a fixed number of elements
[SRS_Adc_12826]	The ADC driver shall implement an API which allows to read the current power state of the ADC HW module
[SRS_Adc_12827]	The ADC driver shall implement an API which allows to read the target power state of the ADC HW module
[SRS_Adc_12828]	The ADC Driver shall separate the power state transition sequence in two parts
[SRS_Adc_12829]	It shall be possible to configure a synchnous or an asynchronous power state transition behavior

Table A.1: Added Requirements in R22-11

A.4.2 Changed Requirements in R22-11

none

A.4.3 Deleted Requirements in R22-11

none

A.4.4 Added Constraints in R22-11

none

A.4.5 Changed Constraints in R22-11

none

A.4.6 Deleted Constraints in R22-11

none

A.4.7 Added Advisories in R22-11

none

A.4.8 Changed Advisories in R22-11

none

A.4.9 Deleted Advisories in R22-11

none