MCAL User Manual for Dsadc

32-bit TriCoreTM AURIXTM TC3xx microcontroller

About this document

Scope and purpose

This User Manual is intended to enable users to integrate the Microcontroller Abstraction Layer (MCAL) software for the TriCoreTM AURIXTM family of 32-bit microcontrollers.

This document describes responsibilities of integrator in-charge of integrating MCAL software with the basic software (BSW) stack. This document also provides detailed information on safety, configuration and functions along with examples of usage of significant features.

Note: Detailed information about package installation, safety and other generic information that are common across all modules are provided in MCAL User Manual General.

Intended audience

This document is intended for anyone using the Dsadc module of the TC3xx MCAL software.

Document conventions

Table 1 Conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, table and figure captions, screen names, windows, dialog boxes, menus, sub-menus
Italics	Denotes variable(s) and reference(s)
`Courier` `New`	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets
>	Indicates that a cascading sub-menu opens when you select a menu item
[cover parentID=<alpha numeric value>]	Used for traceability completeness. Reader should ignore these.

Reference documents

This User Manual should be read in conjunction with the following documents:

- AURIXTM TC3xx MCAL User Manual General

Table of contents

Table of contents

1 Dsadc driver

1.1 User information

1.1.1 Description

The DSADC driver provides analog-to-digital conversion based on the Delta Sigma (DS) conversion principle. The DSADC driver provides configurations for various parameters of the functional blocks of the EDSADC IP. The driver is responsible for the initialization and configuration of the channels (internal modulators, demodulators, filter chain) in the EDSADC IP, thus providing interfaces to convert analog input signals to digital data streams at a selectable output rate. The DSADC driver does not support multicore processing. The driver is delivered as a post-build variant.

1.1.2 Hardware-software mapping

This section describes the system view of the DSADC driver and peripherals administered by it.

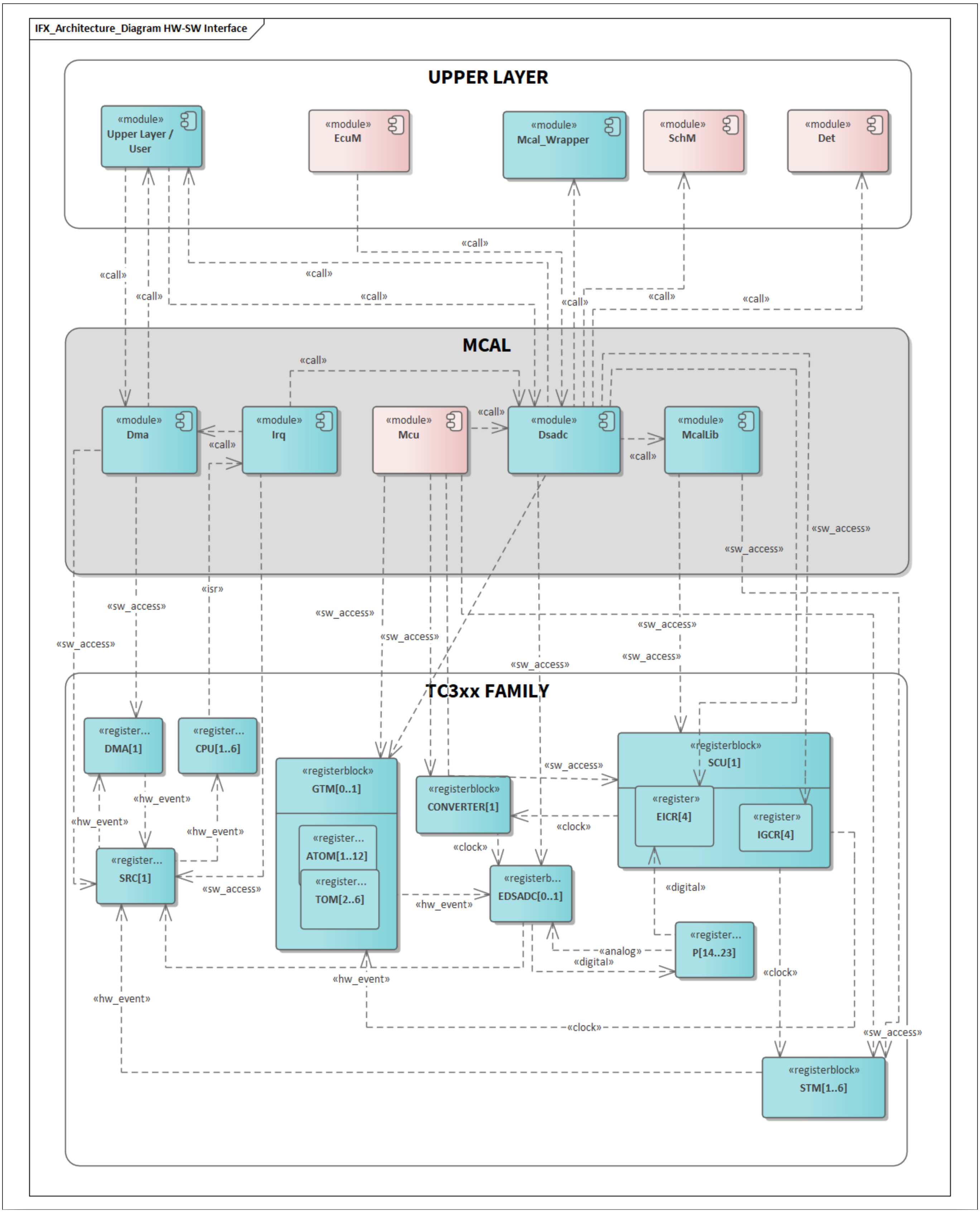


Figure 1 Mapping of hardware-software interfaces

1.1.2.1 CONVERTER: dependent hardware peripheral

Hardware functional features

The DSADC driver depends on the converter control block for the clock synchronization signal. The clock synchronization signal synchronizes the analog clocks of all EDSADC hardware channels.

Users of the hardware

The converter control block is configured by the MCU driver.

Hardware diagnostic features

The SMU alarms configured for the converter control block are not monitored by the DSADC driver.

Hardware events

Not applicable.

1.1.2.2 SCU: dependent hardware peripheral

Hardware functional features

The DSADC driver depends on the SCU IP for the clock, ENDINIT and reset functionalities. The driver requires the fSPB and fPER clock signals for functioning.

Users of the hardware

The SCU IP supplies clock for all the peripherals and the MCU driver, and is responsible for configuring the clock tree. To avoid conflicts due to simultaneous writes, update to all the ENDINIT protected registers is performed using the MCALLIB APIs.

Hardware diagnostic features

The SMU alarms configured for the SCU IP are not monitored by the DSADC driver.

Hardware events

Hardware events from the SCU are not used by the DSADC driver.

1.1.2.3 EDSADC: primary hardware peripheral.

Hardware functional features

The DSADC driver uses the EDSADC IP for converting the analog signals to digital values. The key hardware functional features used are:

- Service request generation using external trigger
- Filter chain configuration
- Calibration support
- Timestamp capture
- Carrier signal generation
- Integrator support
- Configuration support for return signal synchronization for resolver support
- Configuration support for limit checking feature

The unsupported features of EDSADC are:

- External modulator support
- Automatic power control

- Trigger signal from port pin
- Trigger signal from GTM ADC trigger lines
- Event handling for Limit checking
- Event handling for return signal synchronization

Users of the hardware

The DSADC driver exclusively utilizes the EDSADC IP.

Hardware diagnostic features

The SMU alarms configured for the EDSADC IP are not monitored by the DSADC driver.

Hardware events

The DSADC driver uses the following hardware events from the EDSADC IP:

- Result event: to trigger the conversion result transfer through DMA in DMA mode or CPU in interrupt mode
- Timestamp trigger event: to read the timestamp information for external trigger event (rising/falling)

1.1.2.4 GTM: dependent hardware peripheral

Hardware functional features

The DSADC driver depends on the GTM IP for realizing the gating features. The DSADC driver uses the compare-match event and the channel output signal for starting and stopping the conversion result acquisition of a DSADC channel. The selection of GTM trigger line for the DSADC channel is done by the DSADC driver and the corresponding TOM/ATOM selection for the GTM trigger line is done by the MCU driver.

Users of the hardware

The GTM IP is used by the PWM, OCU, ICU, WDG, GPT and ADC drivers. The GTM resources used by each driver are reserved through the configuration interface of the MCU driver to avoid resource conflict. The GTM TOM/ATOM configuration is done by the PWM driver to generate the gate signal for the DSADC driver.

Hardware diagnostic features

The SMU alarms configured for the GTM IP are not monitored by the DSADC driver.

Hardware events

- Compare-match event: to prepare the channel for the result acquisition or raise a window close notification
- Channel output level: to start/stop the conversion result acquisition

1.1.2.5 SRC: dependent hardware peripheral

Hardware functional features

The DSADC driver depends on interrupt router for raising an interrupt to the CPU based on the result event which indicates the end of conversion of a channel.

Users of the hardware

The interrupt router is configured either by the IRQ driver or the user software.

Hardware diagnostic features

The SMU alarms configured for interrupt router are not monitored by the DSADC driver.

Hardware events

The interrupt events raised by the interrupt router are serviced by the CPU or the DMA. The DSADC driver provides interrupt handlers as software interfaces, which must be invoked from the ISR.

1.1.2.6 EICR/IGCR: primary hardware peripheral

Hardware functional features

The DSADC driver uses the ERU IP for realizing the gating feature. The following features of the ERU are used by the driver:

- Pattern Detection
- Generation of interrupt based on the pattern detection output
- External resource selection for input channel
- Input channel trigger logic selection
- Selection of input channels for pattern detection logic per output unit

The unsupported features of the ERU IP are:

- Generation of interrupt based on the trigger output

Users of the hardware

The ERU IP is used by the ADC, DSADC and ICU drivers. The EICR and IGCR channels used by each driver are reserved through the configuration interfaces of the MCU driver. The channel-specific SFRs are programmed by the driver. Since multiple channels share common SFRs and to avoid corruption of data for other channels, the driver programs these SFRs atomically with a channel specific mask. Glitch filter configuration for digital ports is done by the MCU driver.

Hardware diagnostic features

The SMU alarms configured for the ERU IP are not monitored by the DSADC driver.

Hardware events

- Pattern match/miss event: to prepare the channel for the result acquisition or raise a window close notification
- Pattern detection output level: to start/stop the conversion result acquisition

1.1.2.7 DMA: dependent hardware peripheral

Hardware functional features

The DSADC driver depends on the DMA IP for transferring the conversion results to the application buffer using the DMA channel in the DMA mode of result handling.

Users of the hardware

The DMA channels are configured by the DMA driver.

Hardware diagnostic features

The SMU alarms configured for the DMA IP are not monitored by the DSADC driver.

Hardware events

Hardware events from DMA channels are not used by the DSADC driver.

1.1.2.8 PORT: dependent hardware peripheral

Hardware functional features

The analog signals are routed to the EDSADC through the analog port pads. The external trigger events for the channel are routed through the digital port pad. The generated carrier signal from the EDSADC are routed through the digital port pad. The port pads are configured and enabled through the PORT driver.

Users of the Hardware

The port pads are configured by the PORT driver.

Hardware diagnostic features

Not applicable.

Hardware events

Hardware events from port pads are not used by the DSADC driver.

1.1.2.9 STM: dependent hardware peripheral

Hardware functional features

The DSADC driver uses the STM IP for realizing the delay. STM TIM0 values are read by the driver using functions in MCALLIB to implement the delay function.

Users of the Hardware

The MCALLIB driver handles the requests to read the TIM0 register.

Hardware diagnostic features

Not applicable.

Hardware events

Not used.

1.1.3 File structure

1.1.3.1 C file structure

This section provides details of the C files of the DSADC driver.

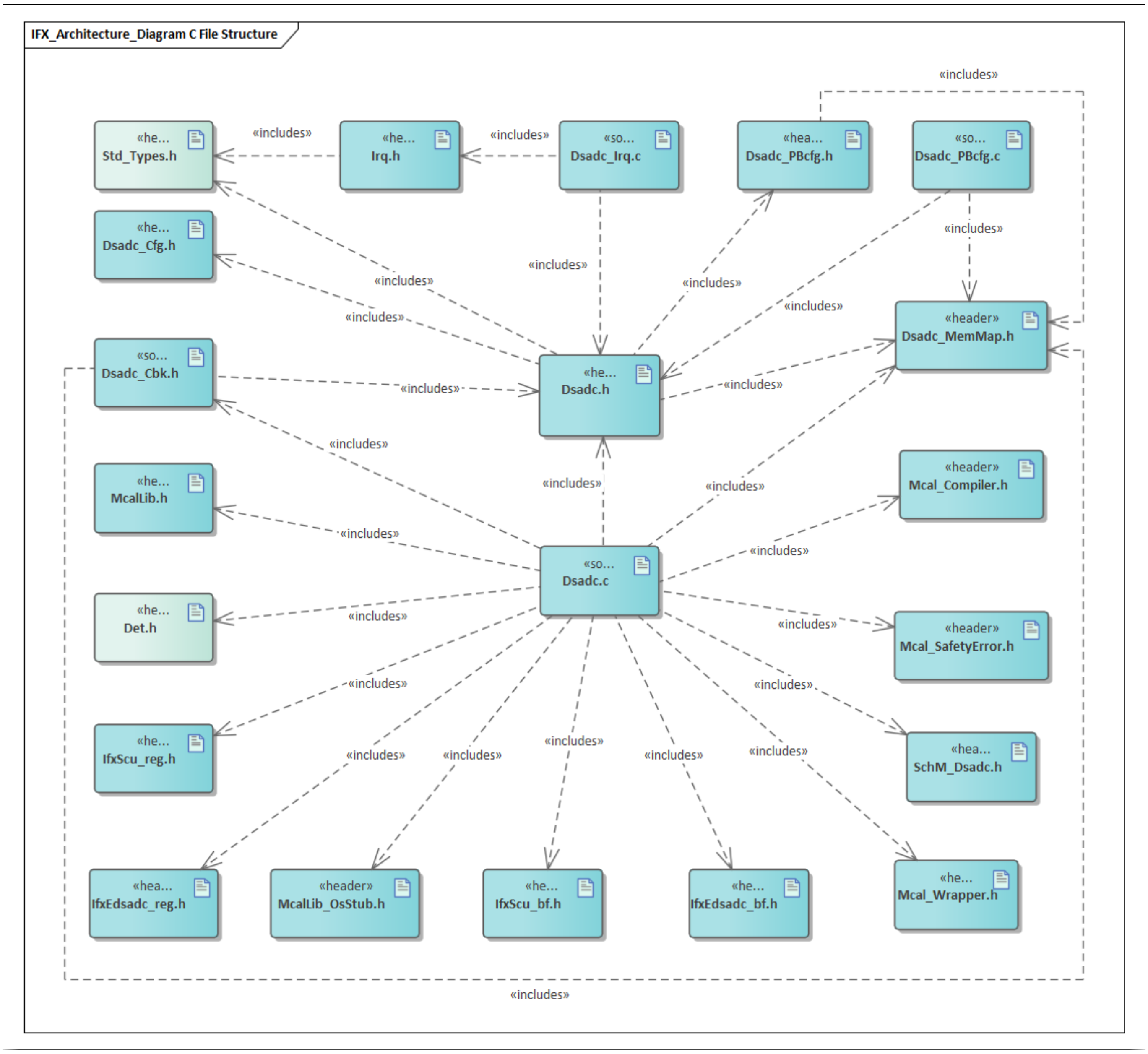


Figure 2 Dsadc_C_File_Structure-1.png

Table 2 C file structure

File name	Description
`Det.h`	Provides the exported interfaces of Development Error Tracer
`Dsadc.c`	File (Static) containing implementation of APIs
`Dsadc.h`	Header file (Static) defining prototypes of data structures, APIs and interrupt handlers
`Dsadc_Cbk.h`	Header file to declare the DSADC callback APIs
`Dsadc_Cfg.h`	Header file (Generated) containing constants and pre-processor macros as #defines
`Dsadc_Irq.c`	Interrupt handler file for DSADC

(table continues...)

Table 2 (continued) C file structure

File name	Description
`Dsadc_MemMap.h`	File (Static) containing the memory section definitions used by the DSADC driver
`Dsadc_PBcfg.c`	File (Generated) containing declaration of the post-build configuration data structures
`Dsadc_PBcfg.h`	File (Generated) containing declaration of the post-build configuration data structures
`IfxEdsadc_bf.h`	SFR header file for EDSADC
`IfxEdsadc_reg.h`	SFR header file for EDSADC
`IfxScu_bf.h`	SFR header file for SCU
`IfxScu_reg.h`	SFR header file for SCU
`Irq.h`	The file exports Mcal compiler specific functions and macros
`McalLib.h`	Static header file defining prototypes of data structure and APIs exported by the MCALLIB.
`McalLib_OsStub.h`	McalLib_OsStub.h provides macros to support user mode of Tricore. This shall be included by other drivers to call OS APIs.
`Mcal_Compiler.h`	Header file providing abstraction for TriCoreTM-intrinsic instruction.
`Mcal_SafetyError.h`	Header file containing the prototype of the API for reporting safety-related errors
`Mcal_Wrapper.h`	Provides the exported interfaces for Production Error and Runtime Development Errors. Implemented by default to include functions of Dem.h and Det.h files. This file can be modified by the user but function prototype is not user modifiable.
`SchM_Dsadc.h`	Export header for SchM functions of DSADC
`Std_Types.h`	Standard type declaration file as defined by AUTOSAR. It is independent of compiler or platform.

1.1.3.2 Code generator plugin files

This section provides details of the code generator plugin files of the DSADC driver.

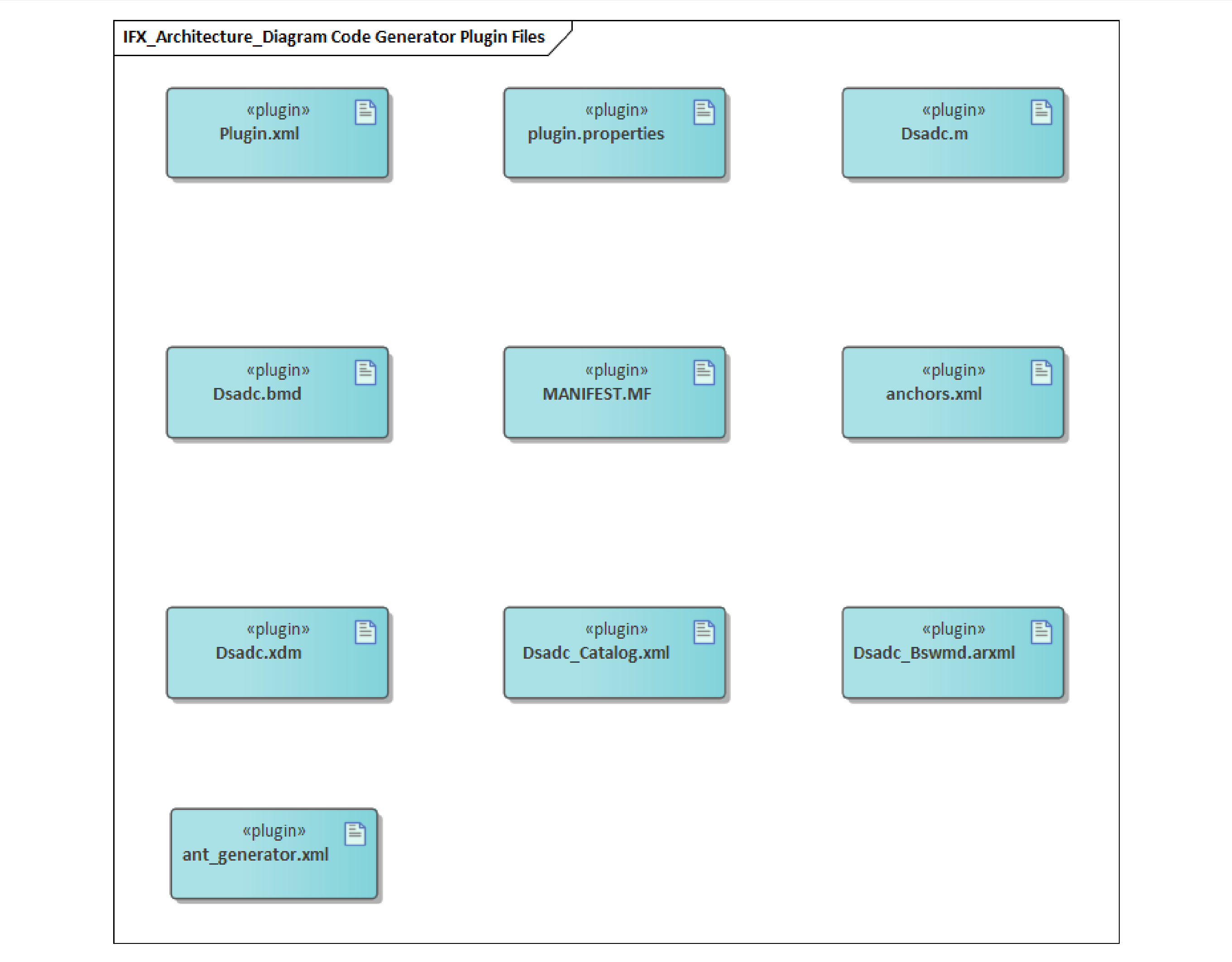

IFX_Architecture_Diagram Code Generator Plugin Files
«plugin» Plugin.xml
«plugin» plugin.properties
«plugin» Dsadc.m
«plugin» Dsadc.bmd
«plugin» MANIFEST.MF
«plugin» anchors.xml
«plugin» Dsadc.xdm
«plugin» Dsadc_Catalog.xml
«plugin» Dsadc_Bswmd.arxml
«plugin» ant_generator.xml

Figure 3 Dsadc_Code_Generator_Plugin_Files-1.png

Table 3 Code generator plugin files

File name	Description
`Dsadc.bmd`	AUTOSAR format XML data model schema file
`Dsadc.m`	Code template macro file for the DSADC driver
`Dsadc.xdm`	Tresos format XML data model schema file
`Dsadc_Bswmd.arxml`	AUTOSAR format module description file
`Dsadc_Catalog.xml`	AUTOSAR format catalog file
`MANIFEST.MF`	Tresos plugin support file containing the meta data for the DSADC driver
`Plugin.xml`	Tresos plugin support file for the DSADC driver
`anchors.xml`	Tresos anchors support file for the DSADC driver
`ant_generator.xml`	Tresos support file to generate and rename multiple post-build configurations when using variation point
`plugin.properties`	Tresos plugin support file for the DSADC driver

1.1.4 Integration hints

This section lists the key points that an integrator or user of the DSDAC driver must consider.

1.1.4.1 Integration with AUTOSAR stack

This section lists the modules, which are not part of the MCAL, but are required to integrate the DSADC driver.

- **EcuM**

  The ECU Manager module is a part of the AUTOSAR stack that manages common aspects of ECU. Specifically, in the context of the MCAL, the EcuM is used for initialization and de-initialization of the software drivers. The EcuM module provided in the MCAL package is a stub code and needs to be replaced with a complete EcuM module during the integration phase.

- **Memory mapping**

  Memory mapping is a concept from AUTOSAR that allows relocation of text, variables, constants and configuration data to user-specific memory regions. To achieve this, all the relocatable elements of the driver are encapsulated in different memory-section macros. These macros are defined in the `Dsadc_MemMap.h` file.

  The `Dsadc_MemMap.h` file is provided in the MCAL package as a stub code. The integrator must place appropriate compiler pragmas within the memory-section macros. The pragmas ensure that the elements are re-located to the correct memory region. A sample implementation listing the memory-section macros is shown as follows:

```
/*To be used for all global or static variables.*/
#if defined DSADC_START_SEC_VAR_CLEARED_ASIL_B_LOCAL_32
 /* User Pragma here */
 #undef DSADC_START_SEC_VAR_CLEARED_ASIL_B_LOCAL_32
 #undef MEMMAP_ERROR
#elif defined DSADC_STOP_SEC_VAR_CLEARED_ASIL_B_LOCAL_32
 /* User Pragma here */
 #undef DSADC_STOP_SEC_VAR_CLEARED_ASIL_B_LOCAL_32
 #undef MEMMAP_ERROR
#elif defined DSADC_START_SEC_VAR_CLEARED_ASIL_B_LOCAL_8
 /* User Pragma here */
 #undef DSADC_START_SEC_VAR_CLEARED_ASIL_B_LOCAL_8
 #undef MEMMAP_ERROR
#elif defined DSADC_STOP_SEC_VAR_CLEARED_ASIL_B_LOCAL_8
 /* User Pragma here */
 #undef DSADC_STOP_SEC_VAR_CLEARED_ASIL_B_LOCAL_8
 #undef MEMMAP_ERROR

/* DSADC module configuration data */
#elif defined DSADC_START_SEC_CONFIG_DATA_ASIL_B_LOCAL_UNSPECIFIED
 /* User Pragma here */
 #undef DSADC_START_SEC_CONFIG_DATA_ASIL_B_LOCAL_UNSPECIFIED
 #undef MEMMAP_ERROR
#elif defined DSADC_STOP_SEC_CONFIG_DATA_ASIL_B_LOCAL_UNSPECIFIED
 /* User Pragma here */
 #undef DSADC_STOP_SEC_CONFIG_DATA_ASIL_B_LOCAL_UNSPECIFIED
 #undef MEMMAP_ERROR

/* Code Sections */
#elif defined DSADC_START_SEC_CODE_ASIL_B_LOCAL
 /* your Pragma here */
 #undef DSADC_START_SEC_CODE_ASIL_B_LOCAL
 #undef MEMMAP_ERROR
#elif defined DSADC_STOP_SEC_CODE_ASIL_B_LOCAL
 /* your Pragma here */
 #undef DSADC_STOP_SEC_CODE_ASIL_B_LOCAL
 #undef MEMMAP_ERROR
#endif

#if defined MEMMAP_ERROR
#error "Dsadc_MemMap.h, wrong pragma command"
#endif
```

- **DET**

  The DET module is a part of the AUTOSAR stack that handles all the development errors reported by the BSW modules. The DSADC driver reports all the development errors to the DET module through the `Det_ReportError()` API. The user of the DMA driver must process all the errors reported to the DET module through the `Det_ReportError()` API.

  The `Det.h` and `Det.c` files are provided in the MCAL package as a stub code and needs to be replaced with a complete DET module during the integration phase.

- **Mcal_Wrapper**

  This Driver performs reporting of the Production and Runtime Errors. The Handling of the reported errors shall be done by the user. The Mcal_Wrapper_Det_ReportRuntimeError() API, Mcal_Wrapper_Dem_SetEventStatus() API and Mcal_Wrapper_Dem_ReportErrorStatus() API are provided in the Mcal_Wrapper.c and Mcal_Wrapper.h files as a stub code, and can be updated by the integrator to handle the reported errors. The files Mcal_Wrapper.c and Mcal_Wrapper.h are user modifiable but the function prototype is not user modifiable and by default the Mcal Wrapper function shall call AUTOSAR DEM and DET Modules.

  The user of the DSADC driver shall process all the Production Errors (fail/pass) reported to the Mcal_Wrapper module. Runtime Errors are not applicable for DSADC. The interface used for reporting Production Error In AUTOSAR version 4.2.2 is Mcal_Wrapper_Dem_ReportErrorStatus() and for AUTOSAR version 4.4.0 is Mcal_Wrapper_ Dem_SetEventStatus(). The Mcal_Wrapper.c and Mcal_Wrapper.h files are provided in the MCAL package as a stub code and can be replaced with a user specific Production handling module/s during the integration phase.
- **SchM**

  The SchM module is a part of the RTE that manages the BSW Scheduler. The DSADC driver uses the exclusive areas defined in the `SchM_Dsadc.h` file to protect the SFRs and variables from concurrent accesses from different threads. The SchM identified for the DSADC driver are:

  - ChannelData

  - IntegratorRestart

  The `SchM_Dsadc.h` and `SchM_Dsadc.c` files are provided in the MCAL package as an example code and needs to updated by the integrator. The user must implement the SchM functions defined by the DSADC driver as suspend / resume of interrupts for the CPU on which the API is invoked. A sample implementation of the SchM functions is shown as follows:

```
/**** Sample implementation of SchM_Dsadc.c ****/
#include "Os.h"
void SchM_Enter_Dsadc_ChannelData(void)
{
/* Start of Critical Section */
SuspendAllInterrupts(); /* Suspend CPU core interrupt */
}
void SchM_Exit_Dsadc_ChannelData(void)
{
/* End of Critical Section */
ResumeAllInterrupts(); /* Resume CPU core interrupt */
}
```

- **Safety error**

  The DSADC driver will report all the detected safety errors through the `Mcal_ReportSafetyError()` API.

  The driver performs only detection and reporting of the safety errors. The handling of the reported errors should be carried out by the user. The `Mcal_ReportSafetyError()` API is provided in the `Mcal_SafetyError.c` and `Mcal_SafetyError.h` files as a stub code, and must be updated by the integrator to handle the reported errors.

  Note: All DET errors are also reported as safety errors (error code used is same as DET).
- **Notifications and callbacks**

The DSADC driver does not implement any notifications. However, the DSADC driver reports the detection of new conversion result (for access mode not configured for DMA Access), window close event (for trigger mode window) and buffer full event (for access mode linear buffer) through notification functions. These notification functions can be configured by the user in Tresos for each DSADC Channel.

The driver does not expect any callbacks from the application, however the driver requires the callback ISR from the MCU.

- **Operating system(OS)**

  The OS or the application must ensure correct type of service and interrupt priority is configured in the SR register. Enabling and disabling of interrupts must also be managed by the OS or application.

  The OS files provided by the MCAL package are only an example code and must be updated by the integrator with the actual OS files for the desired function.

1.1.4.2 Multicore and Resource Manager

The DSADC driver does not support execution on multiple cores simultaneously.

1.1.4.3 MCU support

The DSADC driver is dependent on the MCU driver for the clock configuration and timer IP-related services. The initialization of the DSADC driver must be started only after completing the MCU initialization. The following must be considered while configuring the MCU driver in the EB tresos:

- The Phase synchronizer inside the CONVCTRL block must be programmed according to the required EDSADC modulator frequency, that is, fPHSYNC = fMOD when the synchronized mode is selected in DSADC driver. The configuration and programming of the CONVCTRL block is managed directly by the MCU driver.
- DSADC channel may require gate signal generated by a GTM timer for result data acquisition. DSADC driver shall not configure the GTM channel to generate the gate signal. PWM driver may reserve the GTM channel and generate gate signal for DSADC channel result data acquisition. The same GTM channel needs to be configured in MCU in corresponding GtmchannelForDsadc parameter
- DSADC channel may require gate signal generated by EICR-IGCR for result data acquisition. The EICR-IGCR channels used by the DSADC driver must be reserved in the MCU configuration for exclusive use by the DSADC.

1.1.4.4 Port support

The PORT driver configures the port pins of the entire microcontroller. The user must configure port pins used by the DSADC driver through the PORT driver configuration and initialize the port pins prior to invoking the DSADC initialization.

1.1.4.5 DMA support

The DSADC driver may be configured such that the conversion results are directly transferred from the result register to the application buffers through the DMA move engines. The APIs and configuration parameters of the DMA driver may be used to achieve this. Enabling the DMA mode is a channel-wise feature.

The result register event from EDSADC triggers a service request, which is serviced by the DMA. The DMA move engine transfers the conversion results from result registers to the application buffers. Result event will be triggered on completion of conversion.

The user must ensure the following points, when using this mode:

- Configuration to enable the DMA-based result transfer must be done through the EB tresos parameter: DsadcAccessMode.
- DMA channels intended to be used for the DSADC channel must be reserved and configured through the DMA driver in the EB tresos.
- DSADC driver does not configure the DMA channels. The user of the DSADC should invoke proper DMA APIs to start/stop the DMA channels before starting/stopping an DSADC channel.
- DSADC channel result register address must be configured as a source address in the corresponding DMA channel.
- Address space 0xD and 0xC should not be used for DMA-related usage. The MemMap sections allocating memory in the scratch pad RAM should always generate global addresses instead of local addresses.
- Since the Data CRC and Address CRC features of the DMA are not used for the DSADC driver, the user should ensure that while using the DMA mode a plausibility check of the conversion result is performed either by redundancy or by other means.

1.1.4.6 Interrupt connections

The interrupt connections of the DSADC driver are described in this section.

- **Result handling in interrupt mode**

  Conversion result transfer in interrupt is selected when DsadcAccessMode is not equal to DSADC_DMA_ACCESS. In this mode, the conversion results are transferred from

  result registers to the application buffers in the ISR. The following figure depicts the interrupt connections required by the DSADC driver.

  Note: User shall ensure that correct hardware channel id is passed while invoking `Dsadc_Isr()` from the interrupt frame.

  i.e : In DSADC0SRGM_ISR, `Dsadc_Isr(0)` shall be invoked, in DSADC13SRGM_ISR, `Dsadc_Isr(13)` shall be invoked

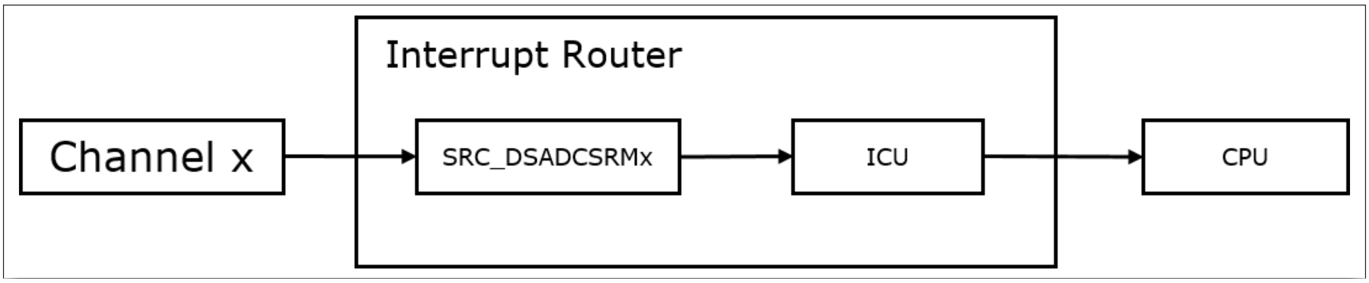


Figure 4 Result handling in the interrupt mode

Invoking the interrupt handlers provided by the driver must be done by the user. A sample invocation for Channel 0 and channel 6 is shown as follows:

```
#include “Dsadc.h”

/* EDSADC Channel 0 */
/******* DSADc Channel 0 Main service Request********/
ISR(DSADC0SRGM_ISR)
{
 /* Enable Global Interrupts */
 ENABLE();
 #if(DSADC_ALL_CH_RESULT_HANDLING_DMA != STD_ON)
 /* Call Dsadc Main Service request Interrupt function*/
 Dsadc_Isr(0); /* 0 indicates the HW channel number */
 #endif
}

/* EDSADC Channel 6 */
/******* DSADc Channel 6 Main service Request********/
ISR(DSADC6SRGM_ISR)
{
 /* Enable Global Interrupts */
 ENABLE();
 #if(DSADC_ALL_CH_RESULT_HANDLING_DMA != STD_ON)
 /* Call Dsadc Main Service request Interrupt function*/
 Dsadc_Isr(6); /* 6 indicates the HW channel number */
 #endif
}
```

- **Result handling in the DMA mode:**
  Conversion result transfer through DMA is selected when DsadcAccessMode is equal to DSADC_DMA_ACCESS. In this mode, the conversion results are transferred from result registers to the application buffers by a DMA move engine. The result register event triggers a service request which is serviced by the DMA. The following figure represents the interrupt connectivity.

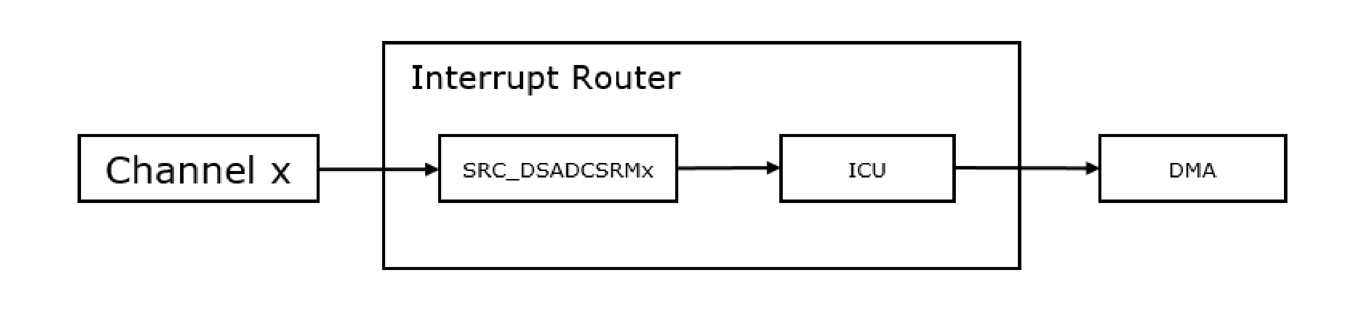


Figure 5 **Result handling in the DMA mode**

1.1.4.7 Example usage

The following are some of the key use cases of the DSADC driver.

Note: Refer to the comments in the code snippets for additional information.

Initialization of the driver

The code sequence for initializing the DSADC driver is as follows:

```
/*
Configuration values mandatory for below code snippet:-
DsadcAccessMode = DSADC_DMA_ACCESS: Then Dma_Init() is required prior to use of runtime DSADC
services.
*/

#include “Dsadc.h”
#include “Mcu.h”
#include “Port.h”
#include “Dma.h”
#include “Irq.h”

/* MCU Initialization */
Mcu_Init(&Mcu_Config);
Mcu_InitClock(0U);
while(Mcu_GetPllStatus() != MCU_PLL_LOCKED);
Mcu_DistributePllClock();

/* Port Initialization */
Port_Init(&Port_Config);

/* Dma Initialization, Only if DMA mode of Result handling is used */
Dma_Init(&Dma_Config);

/* DSADC Initialization */
Dsadc_Init(&Dsadc_Config);

/* Enable Interrupts for used Dsadc Hardware (x) */
SRC_DSADC_DSADCx_SRM.B.SRE = 1U;
/* Further APIs of DSADC driver can be called now */
```

Calibration for Configured channel

The code sequence for starting the calibration and checking for the calibration status is follows:

```
/* Each Dsadc channel has to be calibrated using the service Dsadc_StartCalibration provided by
the Dsadc Driver after the Reset.*/

#include ‘Dsadc.h’

Dsadc_ChannelType ChannelId;
Dsadc_CalibrationStatusType CalibStatus;
Std_ReturnType lRetVal;

/* DsadcChannel_DsadcChannelConfiguration_x is a valid SW channel ID macro Generated in
Dsadc_Cfg.h */

ChannelId = DsadcChannel_DsadcChannelConfiguration_x;

lRetVal = Dsadc_StartCalibration(ChannelId);

if(lRetVal!=E_NOT_OK)
{
 Do
 {
 CalibStatus = Dsadc_GetCalibrationStatus (ChannelId);
 /* Wait till the Start Calibration is over*/
 }while(CalibStatus== DSADC_CALIBRATION_RUNNING);

 If(CalibStatus == DSADC_CALIBRATION_DONE)
 {
 /* Calibration is successful */
 }
 else
 {
 /* Calibration is failed */
 }
}
else
{
/*Could not start the calibration*/
}
```

Software-triggered stream result read (linear buffer)

The code sequence for setting up the linear buffer and reading the conversion results from the buffer is follows:

```
/*
Configuration values mandatory for below code snippet:-
1. DsadcAccessMode = DSADC_STREAM_LINEAR_BUFFER: Dsadc Channel Access mode is Linear buffer*/
2. DsadcTriggerMode = DSADC_TRIGGER_MODE_NORMAL: Conversion Result Acquisition will start
Dsadc_StartModulation API calls
3. DsadcBufferFullNotification = FunctionRead: In this Example Buffer Full notification is
enabled. So once the Buffer is Full user shall get this function call
*/

#include ‘Dsadc.h’

Dsadc_ChannelType ChannelId;
Dsadc_ResultType DataBufferPtr[25]; // Assuming buffer size of 25 is sufficient for the Dsadc
Channel
Dsadc_ResultType UserBufferPtr[25]
Std_ReturnType lRetVal;


/* DsadcChannel_DsadcChannelConfiguration_x is a valid SW channel ID macro Generated in
Dsadc_Cfg.h */
ChannelId = DsadcChannel_DsadcChannelConfiguration_x;

lRetVal = Dsadc_SetupResultBuffer(ChannelId, &DataBufferPtr[0],25);
if(lRetVal!=E_NOT_OK)
{
 Dsadc_EnableNotifications(ChannelId);
 lRetVal = Dsadc_StartModulation(ChannelId);
 if(lRetVal!=E_NOT_OK)
 {
 /* Result Acquisition is started. Result data can be read once the buffer is Full. Assuming
that only buffer full notification is enabled for this channel */
 }
 Else
 {
 /* Result Acquisition is not started */
 }
}
else
{
 /* Could not setup the result buffer */
}
/* Buffer Full notification function */
In FunctionRead()
{
 Dsadc_ReadStreamResults(ChannelId, &UserBufferPtr);
}
```

Software-triggered single result read

The code sequence for setting up the channel for single access and reading the result is follows:

```
/*
Configuration values mandatory for below code snippet:-
1. DsadcAccessMode = DSADC_SINGLE_READ: Dsadc Channel Access mode is Single Read*/
2. DsadcTriggerMode = DSADC_TRIGGER_MODE_NORMAL: Conversion Result Acquisition will start after
Dsadc_StartModulation API calls
3. DsadcNewResultNotification = FunctionRead: In this Example New Result notification event is
enabled. So once the conversion is completed user shall get this function call
*/

#include ‘Dsadc.h’

Dsadc_ChannelType ChannelId;
Dsadc_ResultType ConversionResult;
Std_ReturnType lRetVal;


/* DsadcChannel_DsadcChannelConfiguration_x is a valid SW channel ID macro Generated in
Dsadc_Cfg.h */
ChannelId = DsadcChannel_DsadcChannelConfiguration_x;

Dsadc_EnableNotifications(ChannelId);
lRetVal = Dsadc_StartModulation(ChannelId);
if(lRetVal!=E_NOT_OK)
{
 /* Result Acquisition is started. Result data can be read once the conversion is
completed.Assuming that New Result notification is enabled for this channel */
}
Else
{
 /* Result Acquisition is not started */
```

Software-triggered circular buffer read

The code sequence for setting up the circular buffer and reading the conversion results from the buffer is follows:

```
/*
Configuration values mandatory for below code snippet:-
1. DsadcAccessMode = DSADC_CIRCULAR_BUFFER: Dsadc Channel Access mode is circular buffer*/
2. DsadcTriggerMode = DSADC_TRIGGER_MODE_NORMAL: Conversion Result Acquisition will start
Dsadc_StartModulation API calls
*/

#include ‘Dsadc.h’

Dsadc_ChannelType ChannelId;
Dsadc_ResultType DataBufferPtr[25]; // Assuming buffer size of 25 is sufficient for the Dsadc
Channel
Dsadc_ResultType ConversionResult;
Dsadc_ChannelstatusType ChannelStatus;
Std_ReturnType lRetVal;


/* DsadcChannel_DsadcChannelConfiguration_x is a valid SW channel ID macro Generated in
Dsadc_Cfg.h */
ChannelId = DsadcChannel_DsadcChannelConfiguration_x;

lRetVal = Dsadc_SetupResultBuffer(ChannelId, &DataBufferPtr[0],25);
if(lRetVal!=E_NOT_OK)
{
 lRetVal = Dsadc_StartModulation(ChannelId);
 if(lRetVal!=E_NOT_OK)
 {
 /* Result Acquisition is started. Result data can be read anytime after the first conversion
results are available*/
 }
 Else
 {
 /* Result Acquisition is not started */
 }
}
else
{
 /* Could not setup the result buffer */
}

/* Read Single conversion results from the circular buffer */
In FunctionRead()
{
 ChannelStatus = Dsadc_GetStatus(ChannelId);
 If(ChannelStatus == DSADC_RESULT_READY)
 {
 /* read the circular buffer data */
 Dsadc_ReadResult (ChannelId, & ConversionResult);
 }
 Else
```

```
    {
    /* Buffer is Empty */
    }
  }
```

Hardware-triggered stream result read (linear buffer)

The code sequence for setting up the linear buffer and reading the conversion results from the buffer is follows:

```
/*
Configuration values mandatory for below code snippet:-
1. DsadcAccessMode = DSADC_STREAM_LINEAR_BUFFER: Dsadc Channel Access mode is Linear buffer*/
2. DsadcTriggerMode = DSADC_TRIGGER_MODE_WINDOW: Conversion Result Acquisition will start after
the Window Open Event
3. DsadcWindowCloseNotification = FunctionRead: In this Example Window close notification is
enabled. So once the window close event, user shall get this function call
*/

#include ‘Dsadc.h’

Dsadc_ChannelType ChannelId;
Dsadc_ResultType DataBufferPtr[25]; // Assuming buffer size of 25 is sufficient for the Dsadc
Channel
Dsadc_ResultType UserBufferPtr[25]
Std_ReturnType lRetVal;


/* DsadcChannel_DsadcChannelConfiguration_x is a valid SW channel ID macro Generated in
Dsadc_Cfg.h */
ChannelId = DsadcChannel_DsadcChannelConfiguration_x;

lRetVal = Dsadc_SetupResultBuffer(ChannelId, &DataBufferPtr[0],25);
if(lRetVal!=E_NOT_OK)
{
 Dsadc_EnableNotifications(ChannelId);
 lRetVal = Dsadc_StartModulation(ChannelId);
 if(lRetVal!=E_NOT_OK)
 {
 /* Result Acquisition will start after the window Open event*/
 }
 Else
 {
 /* Result Acquisition is not started */
 }
}
else
{
 /* Could not setup the result buffer */
}
/* Window close notification function */
In FunctionRead()
{
 Dsadc_ReadStreamResults(ChannelId, &UserBufferPtr);
}
```

Stop the result data acquisition

The code sequence for stopping the Result data acquisition is follows:

```
#include ‘Dsadc.h’

Dsadc_ChannelType ChannelId;
Std_ReturnType lRetVal;


/* DsadcChannel_DsadcChannelConfiguration_x is a valid SW channel ID macro Generated in
Dsadc_Cfg.h */
ChannelId = DsadcChannel_DsadcChannelConfiguration_x;

/* Make sure Channelhas already start result data acquisition by calling Dsadc_StartModulation
API */

/* Disable the result data acquisition */
lRetVal = Dsadc_StopModulation(ChannelId);
if(lRetVal!=E_NOT_OK)
{
 /* Result data Acquisition is stopped */
}
Else
{
 /* Result Acquisition is not stoped */
}
```

Read the timestamp for the last read result event

The code sequence to read the timestamp value for read result event is as follows:

```
/*
Configuration values mandatory for below code snippet:-
1. DsadcAccessMode = DSADC_SINGLE_READ: Dsadc Channel Access mode is Single Read
2. DsadcTriggerMode = DSADC_TRIGGER_MODE_NORMAL: Conversion Result Acquisition will start after
invoking the Dsadc_StartModulation API.
3. DsadcTimestampFeature = DSADC_TIMESTAMP_ENABLED: Timestamp is enabled
*/

#include ‘Dsadc.h’

Dsadc_ChannelType ChannelId;
Dsadc_ResultType ConversionResult;
Dsadc_TimeStampType Timestamp;
Std_ReturnType lRetVal;


/* DsadcChannel_DsadcChannelConfiguration_x is a valid SW channel ID macro Generated in
Dsadc_Cfg.h */
ChannelId = DsadcChannel_DsadcChannelConfiguration_x;

lRetVal = Dsadc_StartModulation(ChannelId);
if(lRetVal!=E_NOT_OK)
{
 /* Result Acquisition is started. Result data can be read once the conversion is completed. */
}
Else
{
 /* Result Acquisition is not started */
}

/* Read Single conversion results from the circular buffer */
In FunctionRead()
{
 ChannelStatus = Dsadc_GetStatus(ChannelId);
 If(ChannelStatus == DSADC_RESULT_READY)
 {
 /* read the circular buffer data */
 lRetVal = Dsadc_ReadResult (ChannelId, & ConversionResult);
 if(lRetVal!=E_NOT_OK)
 {
 ConversionResult = Dsadc_GetTimestamp(ChannelId);
 }
 }
 Else
 {
 /* Result is not available */
 }
}
```

Start the carrier signal

The code sequence to start the carrier signal Generation is follows:

```
#include ‘Dsadc.h’

Std_ReturnType lRetVal;

lRetVal = Dsadc_StartCarrierSignal();

if(lRetVal!=E_NOT_OK)
{
 /* Carrier signal Generation is started. */
}
Else
{
 /* Carrier signal Generation is not started. */
}
```

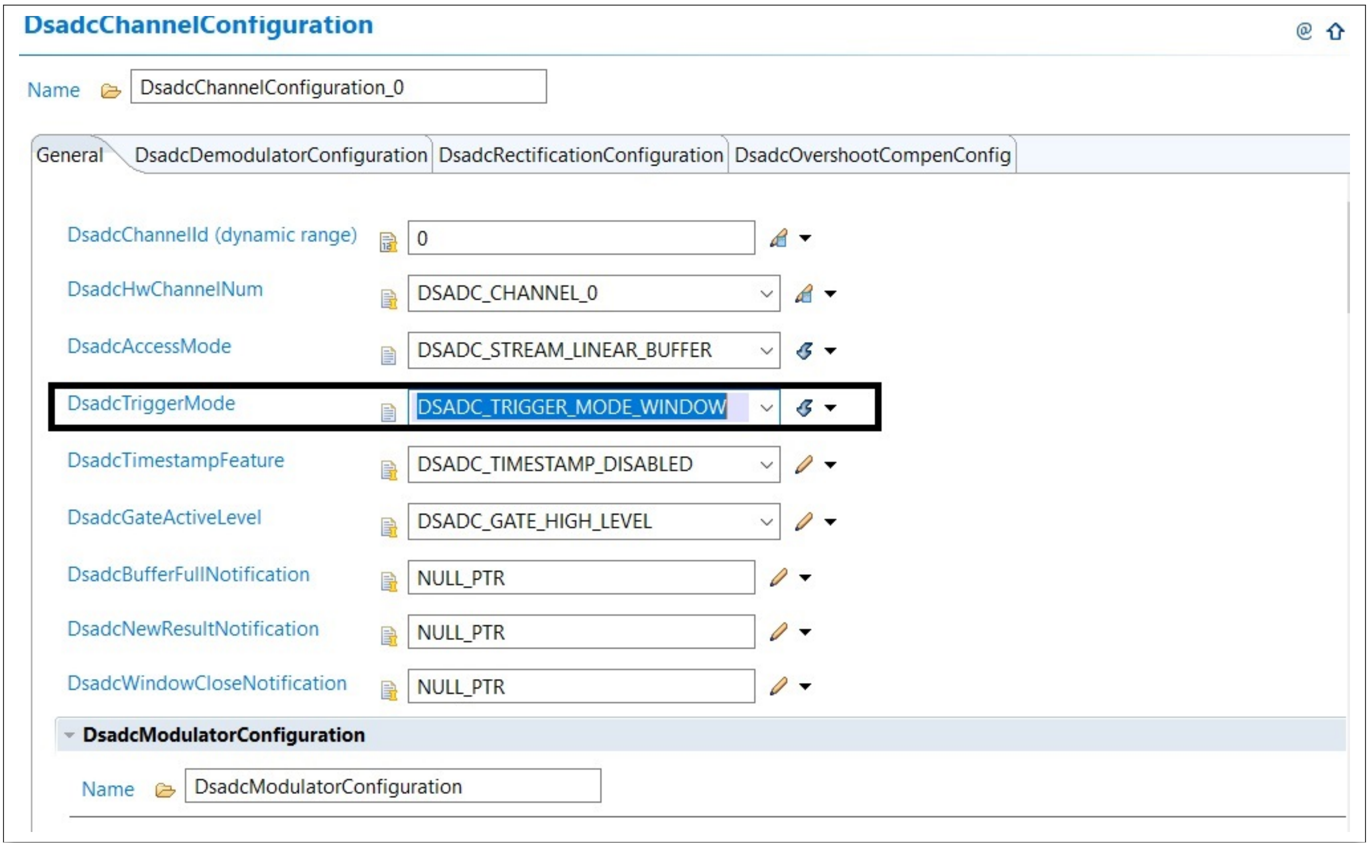
Stop the carrier signal

The code sequence to start the carrier signal generation is as follows:

```
#include ‘Dsadc.h’

Std_ReturnType lRetVal;

/* Ensure that the Carrier generation is already started using the service
Dsadc_StartCarrierSignal */

lRetVal = Dsadc_StopCarrierSignal();

if(lRetVal!=E_NOT_OK)
{
 /* Carrier signal Generation is started. */
}
Else
{
 /* Carrier signal Generation is not started. */
}
```

Configuration example for trigger mode window

Below Example shows how to configure the trigger mode window using ERU/GTM



DsadcChannelConfiguration

Name: DsadcChannelConfiguration_0

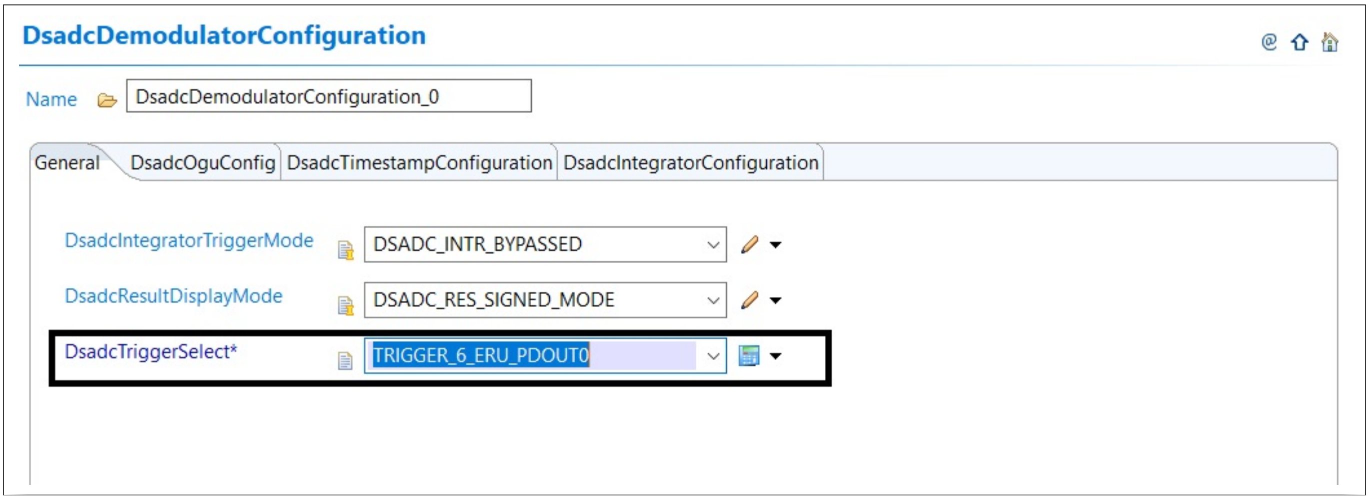
General
DsadcDemodulatorConfiguration
DsadcRectificationConfiguration
DsadcOvershootCompenConfig

DsadcChannelId (dynamic range): 0
DsadcHwChannelNum: DSADC_CHANNEL_0
DsadcAccessMode: DSADC_STREAM_LINEAR_BUFFER
DsadcTriggerMode: DSADC_TRIGGER_MODE_WINDOW
DsadcTimestampFeature: DSADC_TIMESTAMP_DISABLED
DsadcGateActiveLevel: DSADC_GATE_HIGH_LEVEL
DsadcBufferFullNotification: NULL_PTR
DsadcNewResultNotification: NULL_PTR
DsadcWindowCloseNotification: NULL_PTR

DsadcModulatorConfiguration

Name: DsadcModulatorConfiguration

Figure 6 **Configuration: Trigger mode window**

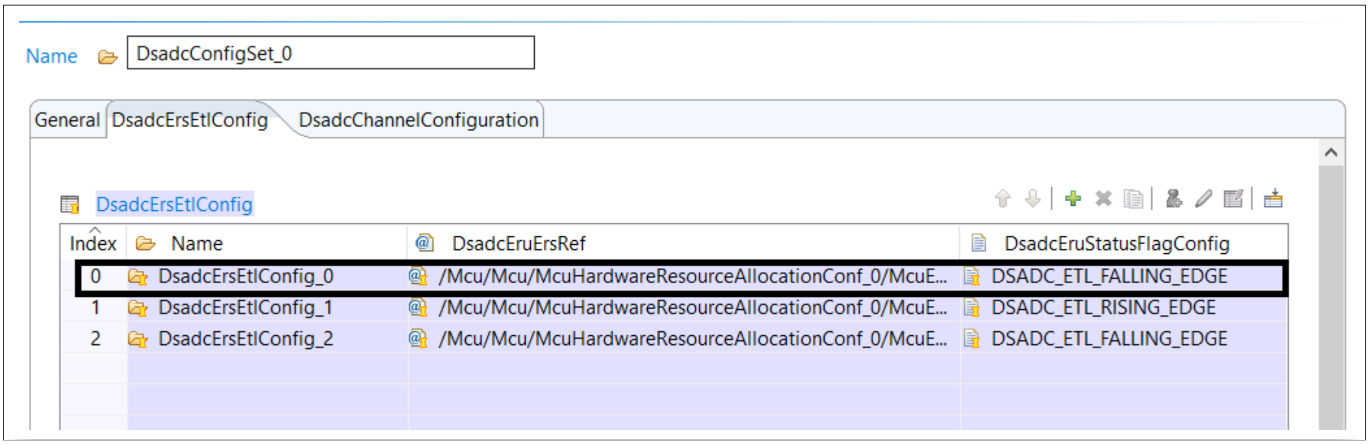


DsadcDemodulatorConfiguration

Name: DsadcDemodulatorConfiguration_0

DsadcIntegratorTriggerMode: DSADC_INTR_BYPASSED
DsadcResultDisplayMode: DSADC_RES_SIGNED_MODE
DsadcTriggerSelect*: TRIGGER_6_ERU_PDOUT0

Figure 7 **Configuration: Trigger source ERU pattern detection output**

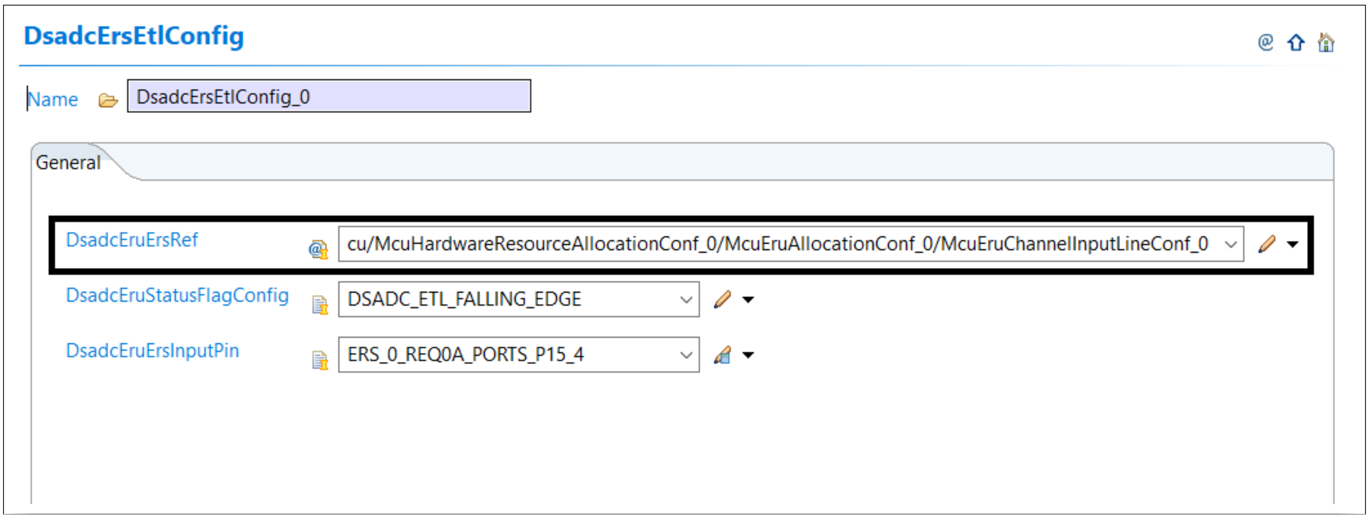


Name: DsadcConfigSet_0

DsadcErsEtlConfig

Index	Name	DsadcEruErsRef	DsadcEruStatusFlagConfig
0	DsadcErsEtlConfig_0	/Mcu/Mcu/McuHardwareResourceAllocationConf_0/McuE...	DSADC_ETL_FALLING_EDGE
1	DsadcErsEtlConfig_1	/Mcu/Mcu/McuHardwareResourceAllocationConf_0/McuE...	DSADC_ETL_RISING_EDGE
2	DsadcErsEtlConfig_2	/Mcu/Mcu/McuHardwareResourceAllocationConf_0/McuE...	DSADC_ETL_FALLING_EDGE

Figure 8 Configuration: Add ERS container



DsadcErsEtlConfig

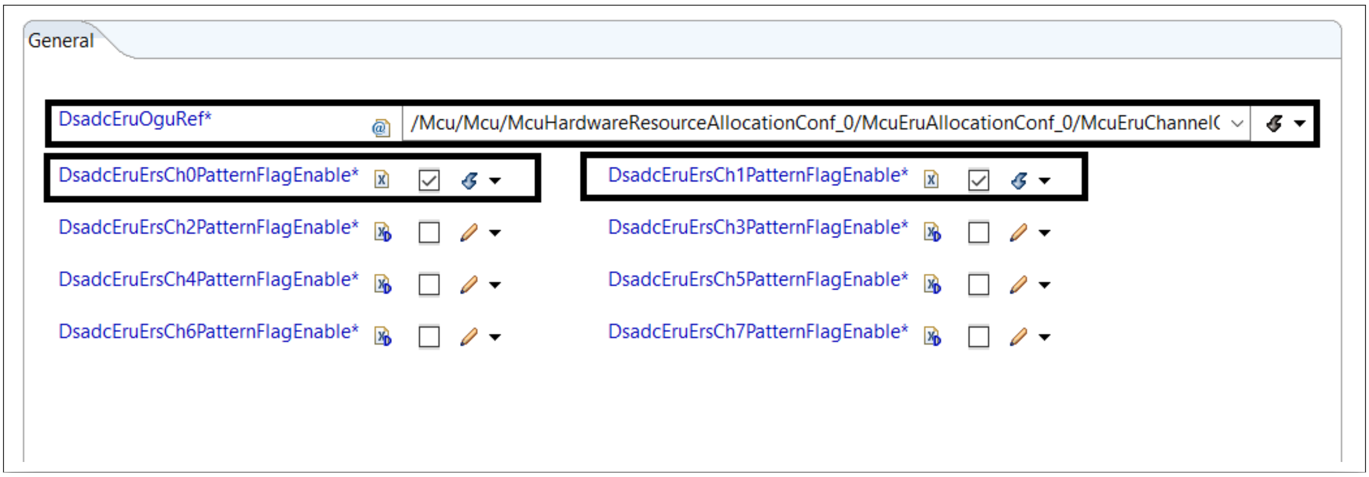
Name: DsadcErsEtlConfig_0

DsadcEruErsRef: cu/McuHardwareResourceAllocationConf_0/McuEruAllocationConf_0/McuEruChannelInputLineConf_0

DsadcEruStatusFlagConfig: DSADC_ETL_FALLING_EDGE

DsadcEruErsInputPin: ERS_0_REQ0A_PORTS_P15_4

Figure 9 Configuration: ERS channel selection



DsadcEruOguRef*: /Mcu/Mcu/McuHardwareResourceAllocationConf_0/McuEruAllocationConf_0/McuEruChannel(

DsadcEruErsCh0PatternFlagEnable* ☑

DsadcEruErsCh1PatternFlagEnable* ☑

DsadcEruErsCh2PatternFlagEnable* □

DsadcEruErsCh3PatternFlagEnable* □

DsadcEruErsCh4PatternFlagEnable* □

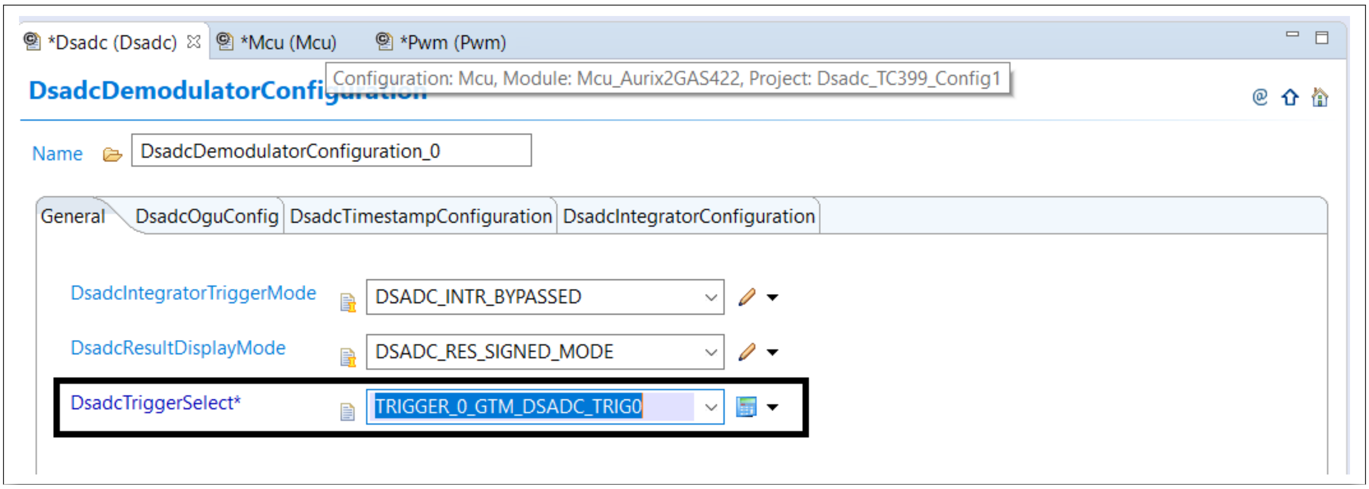
DsadcEruErsCh5PatternFlagEnable* □

DsadcEruErsCh6PatternFlagEnable* □

DsadcEruErsCh7PatternFlagEnable* □

Figure 10 Configuration: Select the OGU channel, select the ERS input channels for pattern detection

Trigger source as GTM



DsadcDemodulatorConfiguration

Configuration: Mcu, Module: Mcu_Aurix2GAS422, Project: Dsadc_TC399_Config1

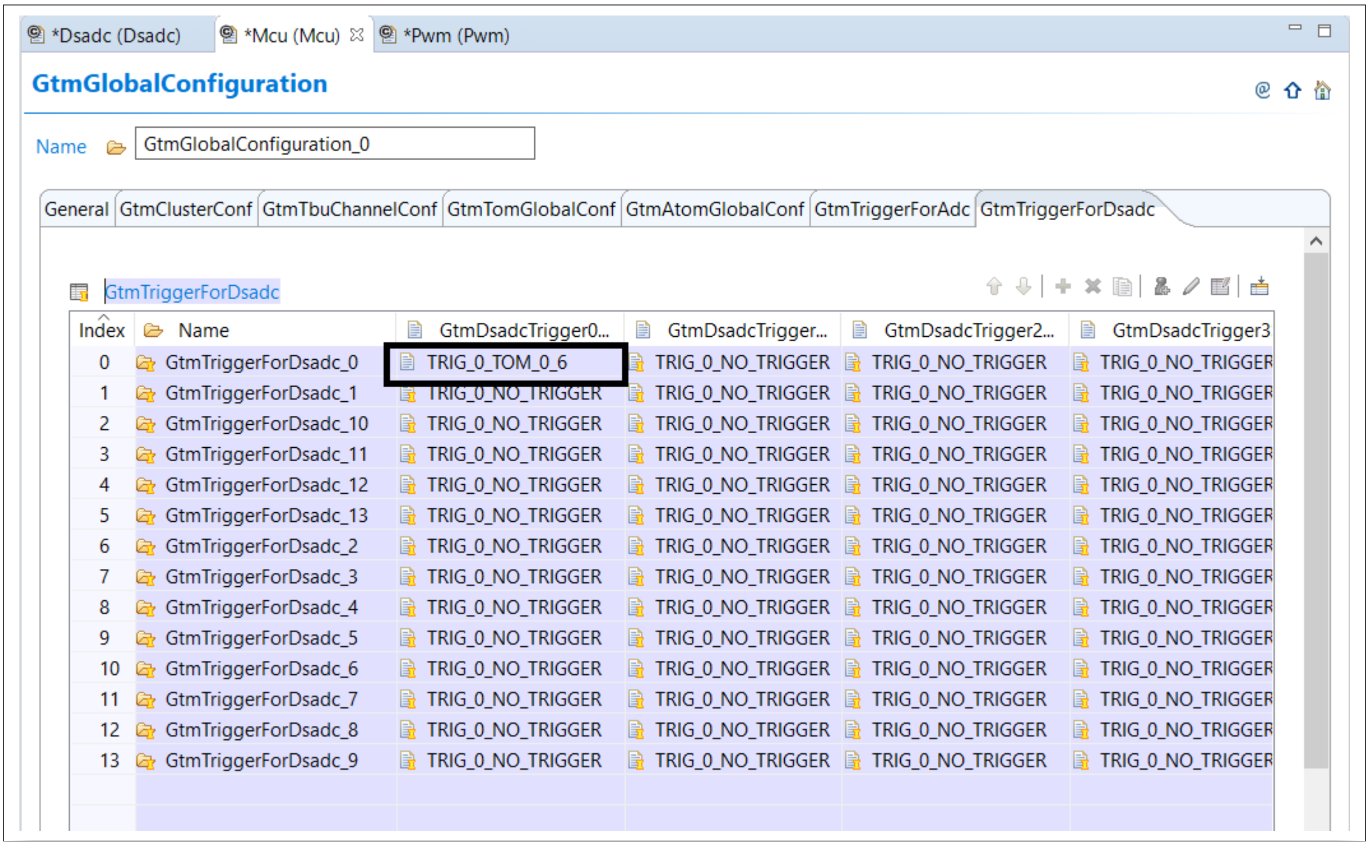
Name: DsadcDemodulatorConfiguration_0

DsadcIntegratorTriggerMode: DSADC_INTR_BYPASSED

DsadcResultDisplayMode: DSADC_RES_SIGNED_MODE

DsadcTriggerSelect*: TRIGGER_0_GTM_DSADC_TRIG0

Figure 11 **Configuration: Trigger source is GTM DSADC trigger line 0**



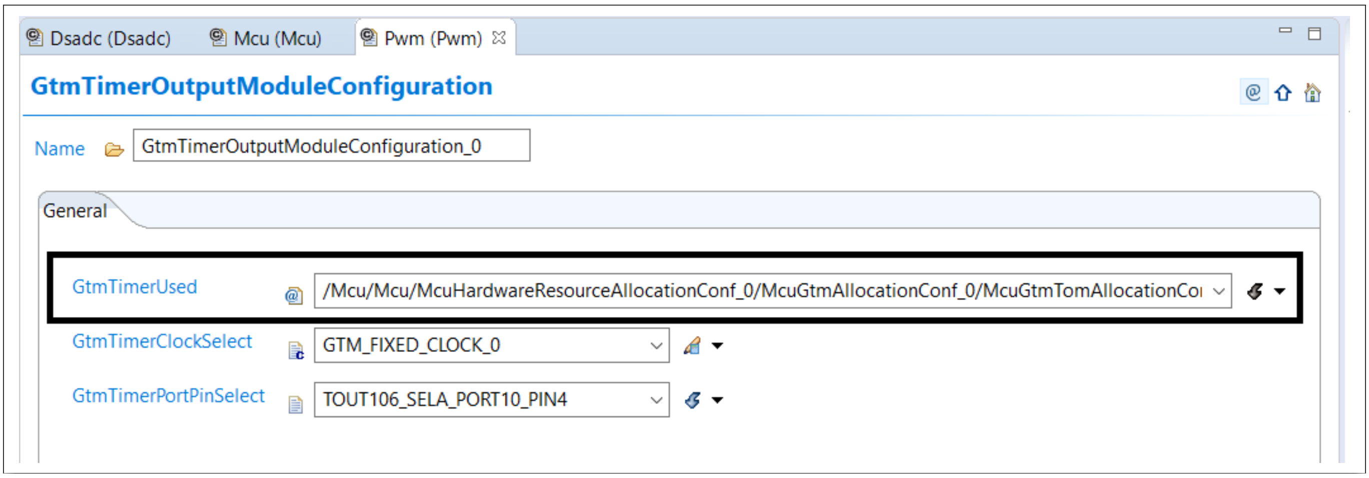
GtmGlobalConfiguration

Name: GtmGlobalConfiguration_0

GtmTriggerForDsadc

Index	Name	GtmDsadcTrigger0...	GtmDsadcTrigger...	GtmDsadcTrigger2...	GtmDsadcTrigger3
0	GtmTriggerForDsadc_0	TRIG_0_TOM_0_6	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
1	GtmTriggerForDsadc_1	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
2	GtmTriggerForDsadc_10	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
3	GtmTriggerForDsadc_11	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
4	GtmTriggerForDsadc_12	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
5	GtmTriggerForDsadc_13	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
6	GtmTriggerForDsadc_2	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
7	GtmTriggerForDsadc_3	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
8	GtmTriggerForDsadc_4	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
9	GtmTriggerForDsadc_5	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
10	GtmTriggerForDsadc_6	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
11	GtmTriggerForDsadc_7	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
12	GtmTriggerForDsadc_8	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER
13	GtmTriggerForDsadc_9	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER	TRIG_0_NO_TRIGGER

Figure 12 **Configuration: Resource selection for GTM DSADC trigger line 0 in MCU driver**



GtmTimerOutputModuleConfiguration

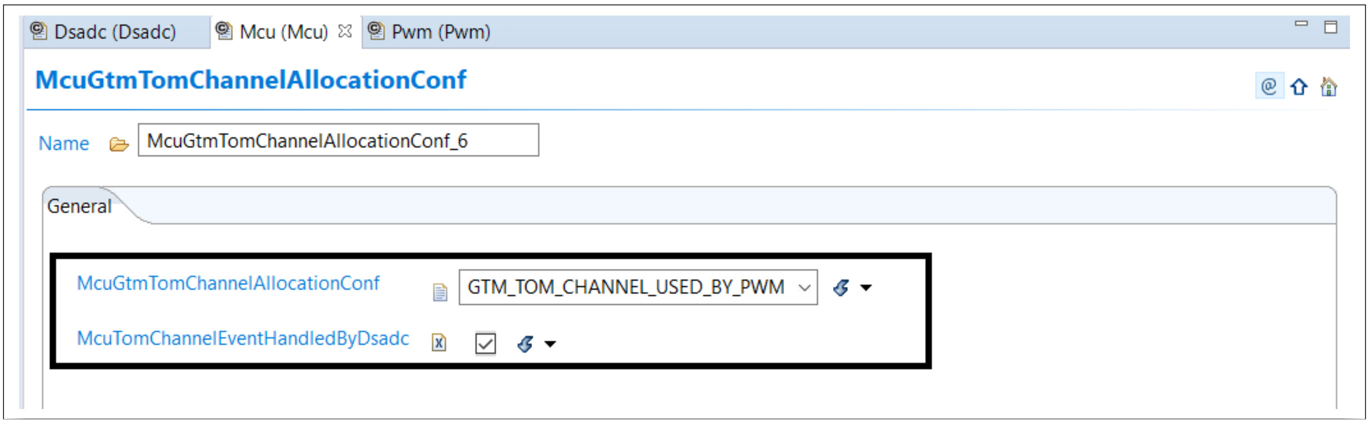
Name: GtmTimerOutputModuleConfiguration_0

GtmTimerUsed: /Mcu/Mcu/McuHardwareResourceAllocationConf_0/McuGtmAllocationConf_0/McuGtmTomAllocationCo

GtmTimerClockSelect: GTM_FIXED_CLOCK_0

GtmTimerPortPinSelect: TOUT106_SELA_PORT10_PIN4

Figure 13 Configuration: GTM resource(TOM/ATOM) configuration in PWM driver



McuGtmTomChannelAllocationConf

Name: McuGtmTomChannelAllocationConf_6

McuGtmTomChannelAllocationConf: GTM_TOM_CHANNEL_USED_BY_PWM

McuTomChannelEventHandledByDsadc: ☑

Figure 14 Configuration: GTM reservation in MCU driver.

Trigger mode window for GTM with PWM driver

The code sequence to initialize and run the trigger mode window functionality for GTM with PWM driver is follows:

```
/*
Configuration values mandatory for below code snippet:-
1. DsadcAccessMode = DSADC_STREAM_LINEAR_BUFFER/DSADC_SINGLE_READ: Dsadc Channel Access mode is
Linear buffer/single read*/
2. DsadcTriggerMode = DSADC_TRIGGER_MODE_WINDOW: Conversion Result Acquisition will start after
the Window Open Event
*/

#include ‘Dsadc.h’
#include ‘Pwm_17_GtmCcu6.h’
#include ‘Irq.h’

Dsadc_ChannelType ChannelId;
Dsadc_ResultType DataBufferPtr[25]; // Assuming buffer size of 25 is sufficient for the Dsadc
Channel
Std_ReturnType lRetVal;


/* DsadcChannel_DsadcChannelConfiguration_x is a valid SW channel ID macro Generated in
Dsadc_Cfg.h */
ChannelId = DsadcChannel_DsadcChannelConfiguration_x;

/* MCU Initialization */
Mcu_Init(&Mcu_Config);
Mcu_InitClock(0U);
while(Mcu_GetPllStatus() != MCU_PLL_LOCKED);
Mcu_DistributePllClock();

/*DSAD0C Initialization*/
Dsadc_Init(&Dsadc_Config);

/*PWM Initialization*/
Pwm_17_GtmCcu6_Init(&Pwm_17_GtmCcu6_Config)

/*Enable Interrupt*/
IrqDsadc_Init();
IrqGtm_Init();
MODULE_SRC.DSADC.DSADC[x].SRM.B.SRE = 1;
MODULE_SRC.GTM.GTMTOMxx.B.SRE = 1;

/* make sure that window is not started */
Pwm_17_GtmCcu6_SetOutputToIdle(ChannelNumber);

lRetVal = Dsadc_SetupResultBuffer(ChannelId, &DataBufferPtr[0],25);
if(lRetVal!=E_NOT_OK)
{
 lRetVal = Dsadc_StartModulation(ChannelId);
 if(lRetVal!=E_NOT_OK)
 {
 /* To enable the GTM interrupt since window open and close event is required in DSADC driver */
 Pwm_17_GtmCcu6_EnableNotification(ChannelNumber);
 /* To start generating the PWM signal from the GTM */
 Pwm_17_GtmCcu6_SetDutyCycle(ChannelNumber,xxxx)
 /* Result Acquisition will start after the window Open event*/
 }
 Else
 {
 /* Result Acquisition is not started */
 }
}
else
{
 /* Could not setup the result buffer */
}

/* Sequence when stop modulation is required*/
/* To stop the gate signal generation */
Pwm_17_GtmCcu6_SetOutputToIdle(ChannelNumber);
Dsadc_StopModulation(Dsadc_ChannelNumber)
```

Read IRMS value and Set gain correction

The code sequence to read the IRMS value from the UCB section and to set the gain correction in GAINCORR register is follows:

```
#include ‘Dsadc.h’

Dsadc_ChannelType ChannelId;
Dsadc_IrmsValueType lIrmsVal;
Std_ReturnType lRetVal;
Dsadc_GainCorrType lRegisterValue;
uint8 lCicshiftVal;
uint16 lGainfactor;

/* DsadcChannel_DsadcChannelConfiguration_x is a valid SW channel ID macro Generated in
Dsadc_Cfg.h */
ChannelId = DsadcChannel_DsadcChannelConfiguration_x;

lIrmsVal = Dsadc_GetIrmsValue(ChannelId);

/* The CIC shift value should not be greater than 0x1C and reserve bits are written with value
zero only.*/
lRegisterValue = (Dsadc_GainCorrType)((lCicshiftVal<<IFX_EDSADC_CH_GAINCORR_CICSHIFT_OFF)|
(lGainfactor));

/* Make sure Channel is not busy */
lRetVal = Dsadc_SetGainCorrRegValue (ChannelId,lRegisterValue);

if(lRetVal!=E_NOT_OK)
{
 /* GAINCORR register is updated as requested. */
}
else
{
 /* GAINCORR register is not updated. */
}
```

Get the SDCAP value and to restart integrator

The code sequence to read the SDCAP value from the register and to restart integrator is follows:

```
#include ‘Dsadc.h’

Dsadc_ChannelType ChannelId;
Dsadc_GetSdcapValue lSdcapvalue;
Dsadc_ChannelMaskType ChannelIdMask;
Dsadc_DelayType lStmDelay;

/* DsadcChannel_DsadcChannelConfiguration_x is a valid SW channel ID macro Generated in
Dsadc_Cfg.h */
ChannelId = DsadcChannel_DsadcChannelConfiguration_x;

/* Ensure that the Carrier generation is already started */
lSdcapvalue = Dsadc_GetSdcapValue (ChannelId);

/*ChannelIdMask is bit coded information for the DSADC logical channel numbers.
DSADC logical channel enable or disable is indicated by the corresponding bit position.
i.e. Channel 0 is requested by setting the bit position 0 of the input parameter ChannelIdMask
*/
ChannelIdMask = (Dsadc_ChannelMaskType) (1U << ChannelId);

/* lStmDelay is calculated based on the STM frequency used.
Ex: To generate a delay of 12.80 microseconds and STM frequency is 100 MHz
The lStmDelay is calculated as Delay/TstmPeriod = 1280 ticks*/

lRetVal = Dsadc_RestartIntegrator(ChannelIdMask,lStmDelay);

if(lRetVal!=E_NOT_OK)
{
 /* Integrators are restarted as requested. */
}
else
{
 /* Integrators are not restarted */
}
```

1.1.5 Key architectural considerations

1.1.5.1 Mode of operation: result acquisition

The DSADC driver supports two modes of operation related to result data acquisition. The modes can be selectable for each channel by assigning appropriate value to the DsadcTriggerMode parameter. These modes are as follows:

- **Result acquisition using gate signal**

  In this mode of operation, once the `Dsadc_StartModulation()` API is called the conversion results are acquired after the window open event. When the `Dsadc_StopModulation()` API is called or in case of a window close event the results acquisition shall be stopped.

The DsadcGateActiveLevel configuration parameter will define the gate signal level where it needs to acquire the result data. If this parameter is configured as DSADC_GATE_HIGH_LEVEL then the gate signal rising edge will be considered as window open and falling edge will be considered as window close. If this parameter is configured as DSADC_GATE_LOW_LEVEL then the gate signal falling edge will be considered as window open and rising edge will be considered as window close.

If the GTM TOM/ATOM is used as a trigger source, then user has to configure and control the TOM/ATOM channel using the PWM driver. In the DSADC driver, TOM/ATOM CCU0 event is considered as window open event and CCU1 event is considered as a window close event. User has to configure the signal level (SL) bit of the TOM/ATOM accordingly. For example, If the PWM polarity between window open and window close events is configured as 'PWM_HIGH' using the parameter 'PwmPolarity', then the gate level chosen for DSADC driver in the parameter 'DsadcGateActiveLevel' should be configured to 'DSADC_GATE_HIGH_LEVEL' and vice versa.

When the ERU pattern detection is used as a trigger source, If the DsadcGateActiveLevel configuration parameter is configured as DSADC_GATE_HIGH_LEVEL then the pattern match event will be considered as window open and pattern miss event will be considered as window close event. If the DsadcGateActiveLevel configuration parameter is configured as DSADC_GATE_LOW_LEVEL then the pattern miss event will be considered as window open and pattern match event will be considered as window close event.

Configuration settings

DsadcTriggerMode: DSADC_TRIGGER_MODE_WINDOW

DsadcTriggerSelect: Can be GTM or ERU Trigger

DsadcGateActiveLevel: DSADC_GATE_HIGH_LEVEL/DSADC_GATE_LOW_LEVEL

- **Result acquisition without using gate signal**

  In this mode of operation, The conversion results are acquired after invoking the `Dsadc_StartModulation` API. The result acquisition is stopped after invoking the `Dsadc_StopModulation` API.

  Configuration settings

  DsadcTriggerMode: DSADC_TRIGGER_MODE_NORMAL

  DsadcTriggerSelect: TRIGGER_0_NO_DSADC_TRIG

1.1.5.2 Mode of operation: result handling

The DSADC driver supports two modes of operation related to result handling. The modes can be selectable for each channel by assigning appropriate value to DsadcAccessMode configuration parameter. These modes are as follows:

- **Interrupt-based result handling**

  In this mode, the conversion results are transferred from the result registers to the application buffers in the ISR.

  Configuration Settings

  DsadcAccessMode: DSADC_SINGLE_READ or DSADC_CIRCULAR_BUFFER or DSADC_STREAM_LINEAR_BUFFER

- **DMA-based result handling**

  In this mode, the conversion results are transferred from the result registers to the application buffers by a DMA move engine. Each channel may have a flexibility to select DMA or interrupt mode.

  For more information, refer to the DMA support section.

  Configuration settings

  DsadcAccessMode: DSADC_DMA_ACCESS

1.1.5.3 Accessing shared SFR

The DSADC driver updates the SFR related to ERU. These SFR may be updated by application software also. Hence, these updates must be done in a critical section or atomically.

- **Accessing ERU registers:** The DSADC driver updates MODULE_SCU.EICR and MODULE_SCU.IGCR to configure the trigger signal. The update to these registers are done atomically by the driver. Any update to the MODULE_SCU.EICR and MODULE_SCU.IGCR by the application should be performed atomically, if the same register is used by the DSADC driver also. This is required since two ERU channels share a common register. Therefore, update for one channel should not corrupt the ongoing write for another channel.

1.1.5.4 Settling time after the filter chain restart

The filter chain is restarted once the calibration is done. So user has to wait for the settling time before calling `Dsadc_StartModulation ()` API. Settling time is mathematically defined by the step response of the related filter chain configuration. The step response has to be considered only for analog sensor signals which have a DC component. For sinusoidal like signals, only the group delay has to be considered. The step response for the four different possible filter chain configurations are defined by following characteristics:

- When the CIC filter is enabled then the step response is the time taken to generate 4 output samples.
- when the CIC filter and FIR0 filters are enabled then the step response is the time taken to generate 5 output samples.
- when the CIC filter, FIR0 filter and FIR1 filter are enabled then the step response is the time taken to generate 15 output samples.
- When the FIR1 filter is configured with a decimation rate of 1 then the step response is is the time taken to generate 30 output samples.

Every of the above described filter chain configurations can be extended by the integrator stage when it is enabled. The step response of the integrator is related to the configured number of accumulation steps.

1.1.5.5 Timestamp

DSADC driver provides the timestamp information in two cases.

- **Timestamp for trigger mode window**

  DSADC driver provides timestamp for the window open event. In this case timestamp is the time between the last conversion result and the window open event.

  Configuration settings

  DsadcTriggerMode: DSADC_TRIGGER_MODE_WINDOW

  DsadcTimestampFeature: DSADC_TIMESTAMP_ENABLED

  Note: By using the conversion result before the window open event, timestamp and the conversion result after the window open event, user can interpolate the conversion result for the window open event.

  Timestamp with conversion results for different access mode is shown as follows:

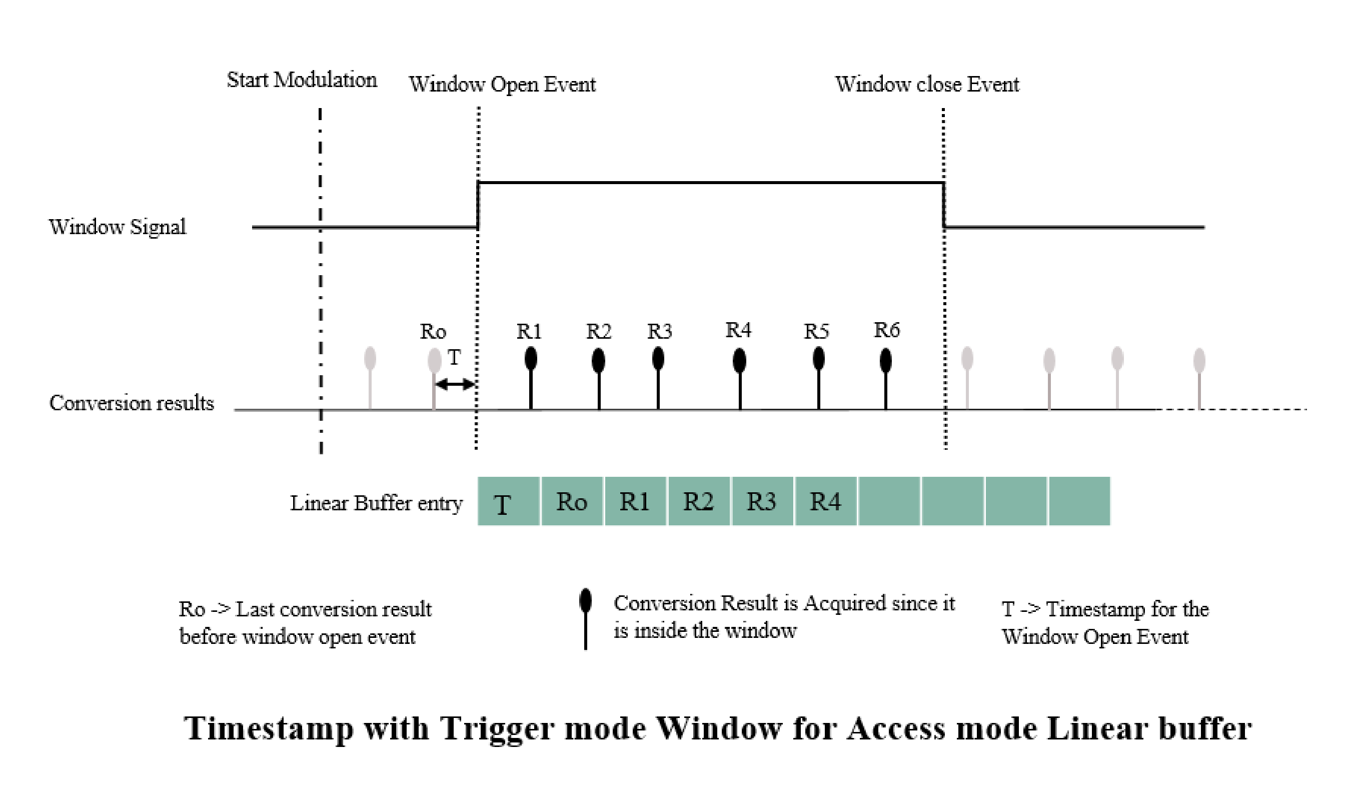

Start Modulation
Window Open Event
Window close Event
Window Signal
Ro
T
R1
R2
R3
R4
R5
R6
Conversion results
Linear Buffer entry
T
Ro
R1
R2
R3
R4
Ro -> Last conversion result before window open event
Conversion Result is Acquired since it is inside the window
T -> Timestamp for the Window Open Event
Timestamp with Trigger mode Window for Access mode Linear buffer

Figure 15 Timestamp with Trigger mode Window for Access mode Linear buffer

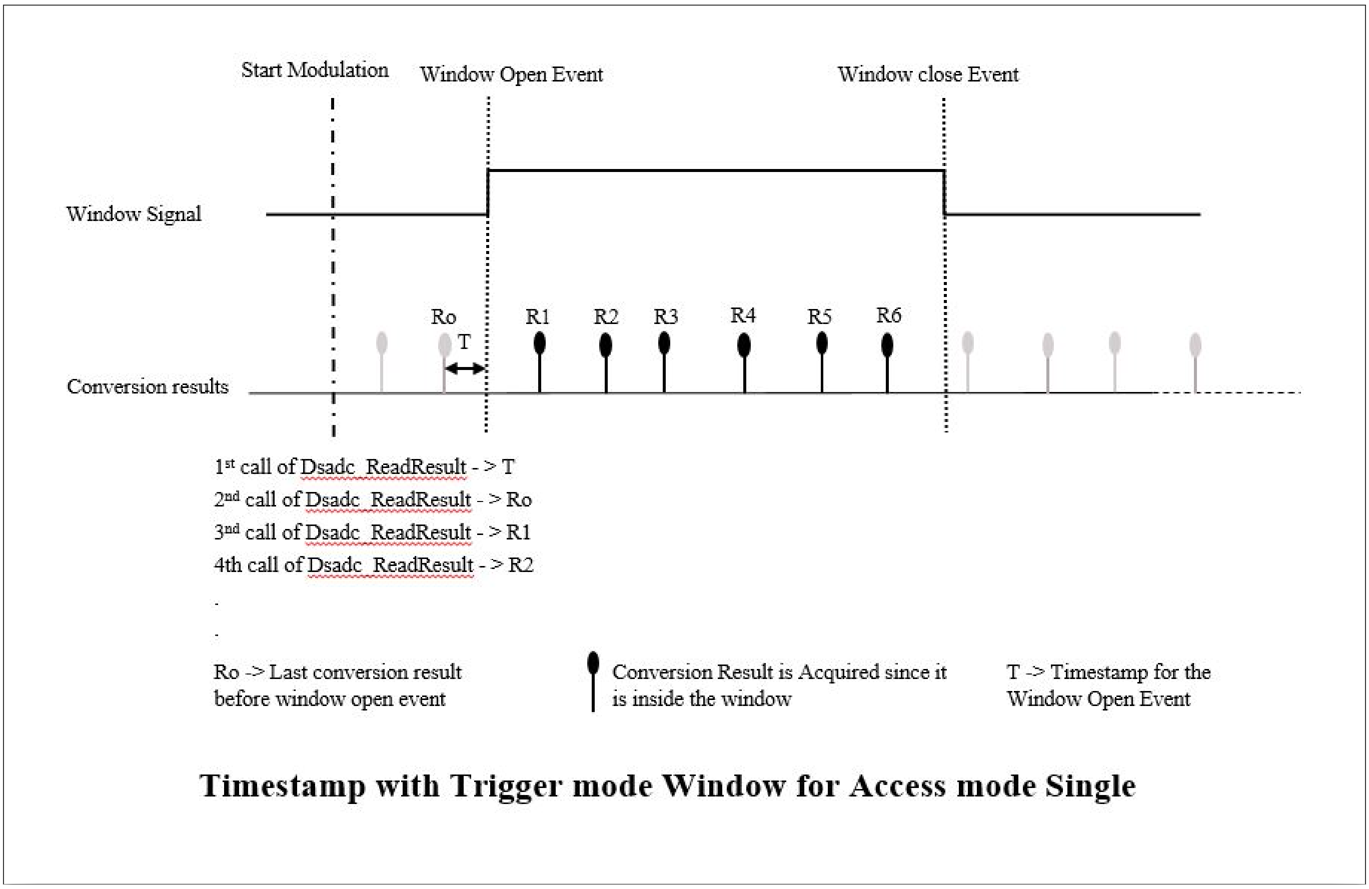

Start Modulation
Window Open Event
Window close Event
Window Signal
Ro
T
R1
R2
R3
R4
R5
R6
Conversion results
1st call of Dsadc_ReadResult - > T
2nd call of Dsadc_ReadResult - > Ro
3nd call of Dsadc_ReadResult - > R1
4th call of Dsadc_ReadResult - > R2
.
.
Ro -> Last conversion result before window open event
Conversion Result is Acquired since it is inside the window
T -> Timestamp for the Window Open Event
Timestamp with Trigger mode Window for Access mode Single

Figure 16 Timestamp with Trigger mode Window for Access mode Single

- **Timestamp for single access mode with trigger mode normal**

  DSADC driver provides the timestamp for the `Dsadc_ReadResult()` API event. In this case timestamp is the time between the conversion result and the `Dsadc_ReadResul()` API reads the result from the hardware result register.

  Configuration settings

  DsadcTriggerMode: DSADC_TRIGGER_MODE_NORMAL

  DsadcTimestampFeature: DSADC_TIMESTAMP_ENABLED

  DsadcAccessMode: DSADC_SINGLE_READ

  Note: By using this timestamp user can interpolate the exact conversion result when the `Dsadc_ReadResult()` API read the result from the result register.

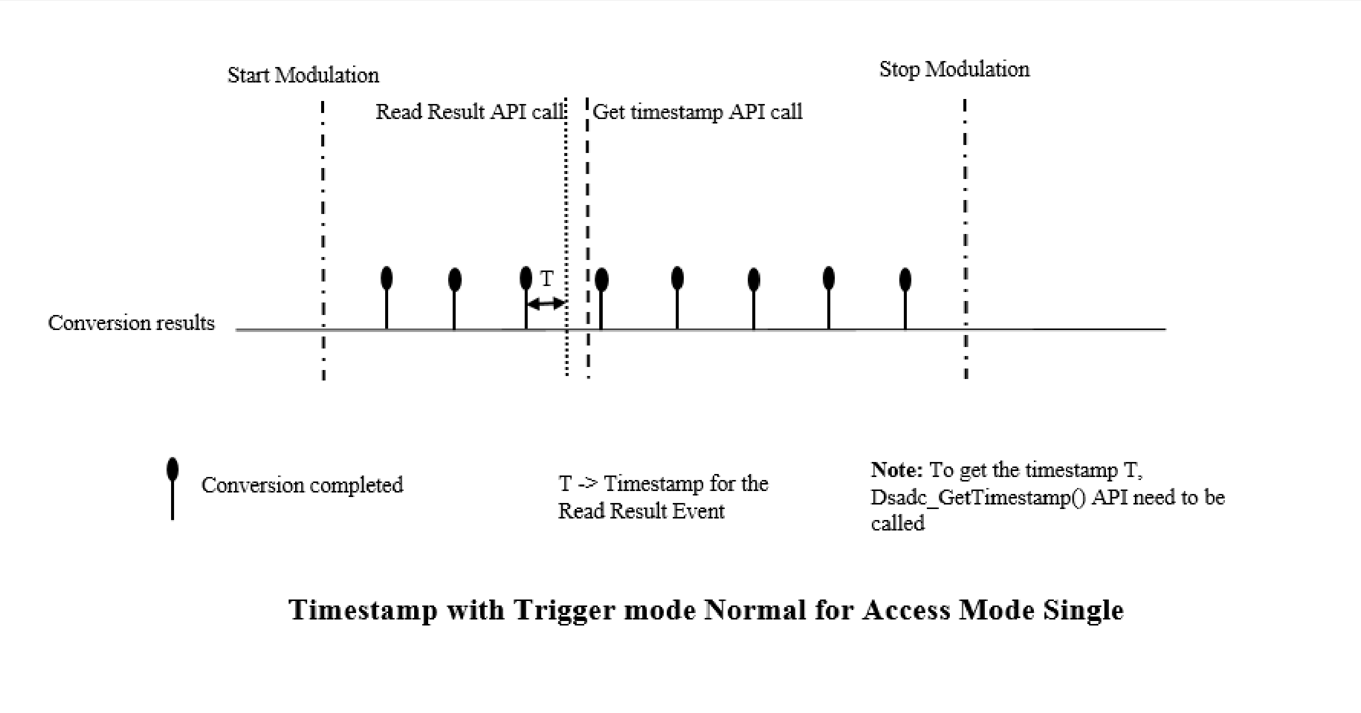

Start Modulation
Read Result API call
Get timestamp API call
Stop Modulation
T
Conversion results
Conversion completed
T -> Timestamp for the Read Result Event
Note: To get the timestamp T, Dsadc_GetTimestamp() API need to be called
Timestamp with Trigger mode Normal for Access Mode Single

Figure 17 Timestamp with Trigger mode Normal for Access Mode Single

1.2 Assumptions of Use (AoU)

The AoU for the DSADC driver are as follows.

- **Call sequence for proper window event handling**

User shall follow the following call sequences when the IFX MCAL PWM driver is used to generate the gate signal:

- Pwm_17_GtmCcu6_SetOutputToIdle API shall be called before invoking Dsadc_StartModulation API.
- Pwm_17_GtmCcu6_SetDutyCycle/Pwm_17_GtmCcu6_SetPeriodAndDuty API shall be called after invoking the Dsadc_StartModulation API.
- Pwm_17_GtmCcu6_EnableNotification API shall be called before invoking the Pwm_17_GtmCcu6_SetDutyCycle/Pwm_17_GtmCcu6_SetPeriodAndDuty API.
- Pwm_17_GtmCcu6_EnableNotification API shall be called after invoking the Pwm_17_GtmCcu6_SetOutputToIdle API.
- Pwm_17_GtmCcu6_SetOutputToIdle API shall be called before invoking the Dsadc_StopModulation API.

[cover parentID DSADC={62B5AE66-5D29-4030-AF77-9E5B0F804888}]

- **Get SDCAP value after carrier generation and demodulators are enabled.**

User shall ensure that the demodulator is enabled and carrier signals are generated before the Dsadc_GetSdcapValue API is called.

[cover parentID DSADC={20944A12-0C6F-4130-BE0F-B057070B7821}]

- **Implausible timestamp value**

User shall aware that Implausible timestamp value(65535) is updated if the buffer is full before the window is closed when the channel is configured for linear buffer with window close timestamp.

[cover parentID DSADC={E36D872D-9ACC-455b-AC50-52AFA496E027}]

- **Restart integrator after demodulator is enabled.**

Users shall call Dsadc_RestartIntegrator API only after the demodulators are enabled.

[cover parentID DSADC={05EDCF64-E55C-4d0f-BBE8-6345AC389C94}]

- **Settling time after Restart demodulator**

User shall use the conversion result after the demanded settling time as when the Dsadc_RestartDemodulator API is called.

Note: Settling time shall be calculated as stated in the HW user manual.

[cover parentID DSADC={5450B855-A3A0-4198-91BF-84991C564F5D}]

- **UCB access for IRMS value**

User shall ensure that UCB sections containing IRMS values are available for read access if Dsadc_GetIrmsValue API is used.

[cover parentID DSADC={8FCD0363-E1A2-42ca-A0B2-AF1450DFC86B}]

- **Wrong duty cycle passed in PWM driver to generate gate signal**

User shall be aware that the duty cycle can not be set to 0 or 100 percentage after the Dsadc_StartModulation API and before the Dsadc_StopModulation API when the IFX MCAL PWM driver is used to generate the gate signal.

[cover parentID DSADC={DD8039D7-C386-4261-8056-0A3A03F2EF23}]

- **Mcu_Init shall be called before Dsadc_Init**

User shall ensure that the MCU driver is initialized before invoking the Dsadc_Init API.

[cover parentID DSADC={526D4F74-EFD6-4c71-B6D2-A71E39407C4A}]

- **Correct configuration pointer for initialization of the DSADC driver**

User shall ensure that the correct configuration pointer is passed for initializing the DSADC driver.

[cover parentID DSADC={5D8325C1-B202-472d-8D8D-429D520FF70B}]

- **User shall pass a pointer different from channel buffer**

User shall ensure that while invoking the Dsadc_ReadStreamResults() API, the passed result buffer pointer shall be different from the channel buffer pointer, which was used during setting up the result buffer.

[cover parentID DSADC={379669C4-790C-4652-8988-4AF381AB63EC}]

- **Stop modulation call sequence**

User shall ensure that the results are read(using Dsadc_ReadResult/ Dsadc_ReadStreamResults) before the Dsadc_StopModulation function is invoked, else the results will be lost.

[cover parentID DSADC={6BD82066-F2AB-4c00-8715-CC6813647BE2}]

- **Setup result buffer call sequence**

User shall ensure that the Dsadc_SetupResultBuffer API is called before invoking the Dsadc_StartModulation API.

[cover parentID DSADC={2456266F-E919-49bf-B217-D90EBD47E7B7}]

- **Past event not notified**

User shall ensure that the Dsadc_EnableNotifications API is invoked before the window close event. Else the notification will be missed and will be provided only for the future window close event.

[cover parentID DSADC={4DB24530-3475-4275-8D1E-57B552D59920}]

- **DMA usage**

User shall use the DMA mode for high output rate (output rate is greater than the ISR execution frequency).

Note: ISR execution frequency = 1/WCET of ISR execution time.

[cover parentID DSADC={AE9B2A95-7D71-40e3-A1DD-D51F2F94FA42}]

- **Correct channel ID passed for runtime APIs**

User shall verify that the macros generated with the symbolic name of channels in the Dsadc_Cfg.h file contain correct values and should use these macros while invoking APIs of the DSADC driver.

[cover parentID DSADC={E34FB178-2046-4013-9F27-FC90430E1C68}]

- **Valid result buffer parameters are passed**

User shall ensure that the passed pointer parameter and the buffer size contain the valid values.

[cover parentID DSADC={AC081E71-E51A-4bcf-840F-15187E0E948E}]

- **Trigger signal priority higher than DSADC ISR**

User shall ensure that the priority of trigger signal ISR is higher than the DSADC conversion completed ISR.

[cover parentID DSADC={AC15A256-9820-4383-9C27-04AE9A3E68A7}]

- **Start modulation call sequence**

User shall ensure that after invoking the Dsadc_Init API, the Dsadc_StartCalibration API is invoked before calling the Dsadc_StartModulation API.

[cover parentID DSADC={5566A5CB-793D-4edc-A6C3-1B82EE0E020E}]

- **Settling time after calibration**

User shall ensure that there is a delay of settling time between the completion of the

Dsadc_StartCalibration API and invocation of the Dsadc_StartModulation API.

[cover parentID DSADC={9BF1B1AD-2960-40e3-91C8-3A96C33C65F8}]

- **Result status check for DMA**

Result ready status is not applicable for the DMA mode and the user should check for the availability of data using the DMA transfer count value.

[cover parentID DSADC={E2E282BE-DD91-4eea-BA7F-79A3F07551D0}]

- **DMA channel initialization sequence**

User shall ensure that the DMA channel is set up before calling the Dsadc_StartModulation API and also DMA channel is stopped/de-initialized after calling the Dsadc_StopModulation API.

[cover parentID DSADC={3F7B0823-89D9-4280-B15D-701DC528CBA6}]

- **Loss of last two samples**

User shall be aware that when the hardware trigger is configured and the timestamp feature is enabled, the last two samples before the window close event are lost.

[cover parentID DSADC={9A59E04C-F733-488b-BB14-73A3D1F00ADB}]

- **Trigger signal start after start modulation call**

User shall ensure that the trigger signal is started after the Dsadc_StartModulation API execution is completed.

[cover parentID DSADC={B161CA40-73AA-4a80-8807-4DB5EEDB39F6}]

- **Correct configuration for calibration parameters**

User shall ensure that the calibration-related configuration parameters are configured correctly for the successful calibration.

[cover parentID DSADC={0B6282A3-B21F-495d-B197-CF1634AEB982}]

- **InitCheck Sequence**

User shall invoke Dsadc_InitCheck to ensure the initialization is done correctly.

The parameter DsadcInitCheckApi shall be enabled and the user of DSADC shall call InitCheck function before the execution of any runtime API (except GetVersionInfo) but after completion of DSADC initialization sequence

[cover parentID DSADC={9B2A932C-AEB3-441d-A87B-E49D630E0E65}]

- **ConfigPtr passed to InitCheck**

User of the DSADC shall ensure that InitCheck is invoked with the same ConfigPtr that was used during initialization.

[cover parentID DSADC={C44CE229-A113-4b63-A987-05F7B6AACF16}]

1.3 Reference information

1.3.1 Configuration interfaces

Supported configuration variant: Post-Build

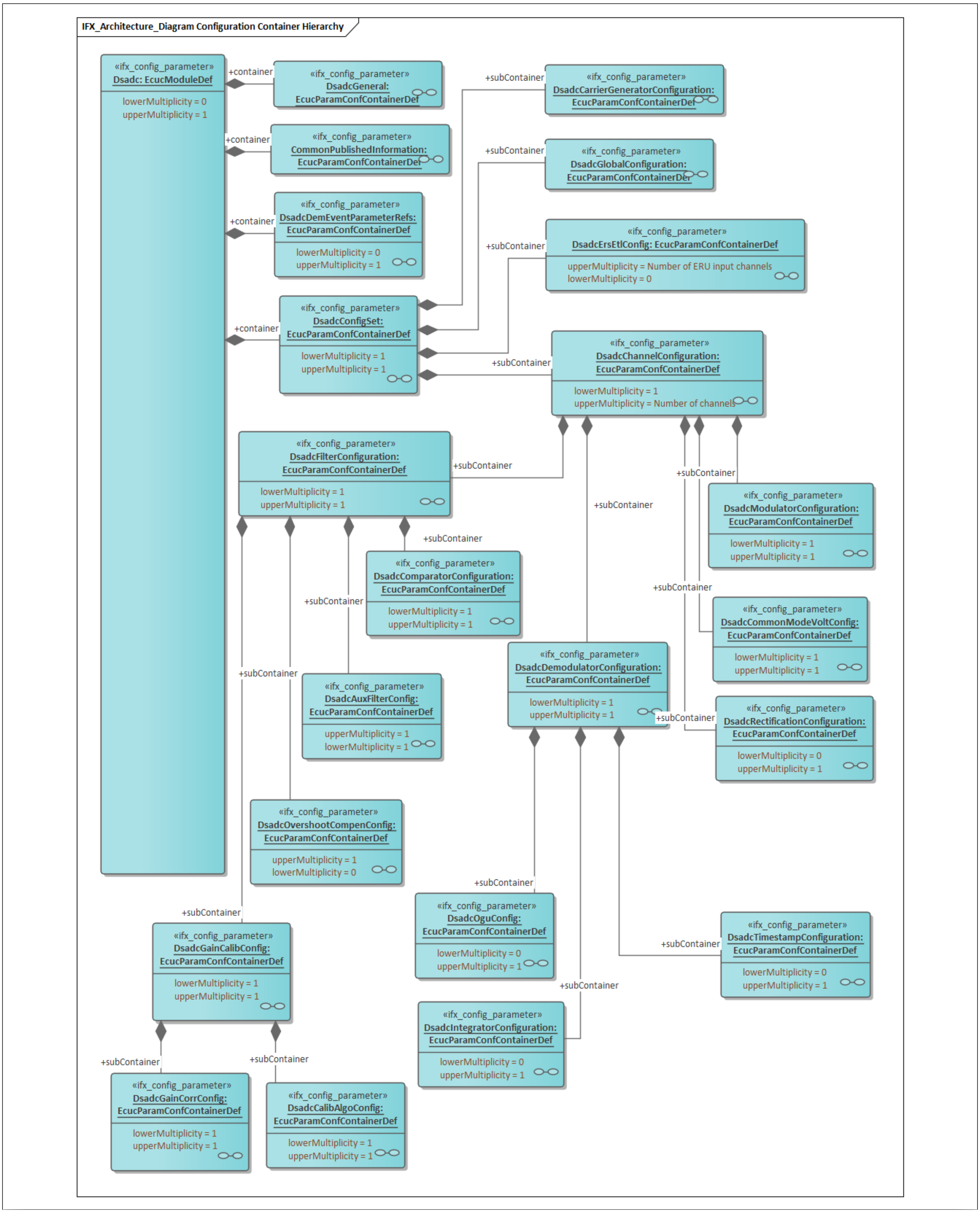


Figure 18 Container hierarchy along with their configuration parameters

1.3.1.1 Container: CommonPublishedInformation

This container contains the published information of the DSADC driver.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.1.1 ArMajorVersion

Table 4 Specification for ArMajorVersion

Name	`ArMajorVersion`		
Description	Major version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	4		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.2 ArMinorVersion

Table 5 Specification for ArMinorVersion

Name	`ArMinorVersion`		
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	2		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.3 ArPatchVersion

Table 6 Specification for ArPatchVersion

Name	`ArPatchVersion`		
Description	Patch version number of AUTOSAR specification on which the appropriate implementation is based on.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	2		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.4 ModuleId

Table 7 Specification for ModuleId

Name	`ModuleId`		
Description	Module ID of this module from Module List		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	255		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.5 Release

Table 8 Specification for Release

Name	`Release`
Description	Aurix2G derivative used for the implementation.

(table continues...)

Table 8 (continued) Specification for Release

Multiplicity	1..1	**Type**	EcucStringParamDef
Range	String		
Default value	As per hardware derivative		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.6 SwMajorVersion

Table 9 Specification for SwMajorVersion

Name	`SwMajorVersion`		
Description	Major version number of the vendor specific implementation of the module. The numbering is vendor specific.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per Driver		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.7 SwMinorVersion

Table 10 Specification for SwMinorVersion

Name	`SwMinorVersion`		
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		

(table continues...)

Table 10 (continued) Specification for SwMinorVersion

Default value	As per Driver		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.8 SwPatchVersion

Table 11 Specification for SwPatchVersion

Name	`SwPatchVersion`		
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per Driver		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.9 VendorId

Table 12 Specification for VendorId

Name	`VendorId`		
Description	Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	17		

(table continues...)

Table 12 (continued) Specification for VendorId

Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2 Container: Dsadc

Configuration of the DSADC (Delta Sigma Analog Digital Conversion) module

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: -

1.3.1.3 Container: DsadcAuxFilterConfig

This configuration container provides parameters related to DSADC auxiliary filter configuration.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.3.1 DsadcAuxCicFilterEnable

Table 13 Specification for DsadcAuxCicFilterEnable

Name	`DsadcAuxCicFilterEnable`		
Description	This parameter defines the availability of DSADC auxiliary filter. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.2 DsadcAuxFilterCicDecimationFactor

Table 14 Specification for DsadcAuxFilterCicDecimationFactor

Name	`DsadcAuxFilterCicDecimationFactor`		
Description	This parameter defines the over sampling rate/decimation factor for the Auxiliary CIC filter. This parameter is set the default value when the parameter DsadcAuxCicFilterEnable is configured as false. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_AUXCIC_OSR_16: DSADC auxiliary CIC filter over sampling rate is 16 DSADC_AUXCIC_OSR_32: DSADC auxiliary CIC filter over sampling rate is 32		
Default value	DSADC_AUXCIC_OSR_16		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcAuxCicFilterEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4 Container: DsadcCalibAlgoConfig

This container provides configuration parameters for Gain calibration.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.4.1 DsadcCICDecimationRate

Table 15 Specification for DsadcCICDecimationRate

Name	`DsadcCICDecimationRate`		
Description	This parameter defines Decimation factor for the CIC filter. This parameter value will be considered only during the execution of the calibration algorithm. The default value of this parameter is set to the reset value of the corresponding SFR. *Note: The decimation factor can be increased when user needs higher precision.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef

(table continues...)

Table 15 (continued) Specification for DsadcCICDecimationRate

Range	DSADC_CIC_DECIMATION_RATE_128: Decimation factor for CIC is 128 during calibration time. DSADC_CIC_DECIMATION_RATE_16: Decimation factor for CIC is 16 during calibration time. DSADC_CIC_DECIMATION_RATE_256: Decimation factor for CIC is 256 during calibration time. DSADC_CIC_DECIMATION_RATE_32: Decimation factor for CIC is 32 during calibration time. DSADC_CIC_DECIMATION_RATE_512: Decimation factor for CIC is 512 during calibration time. DSADC_CIC_DECIMATION_RATE_64: Decimation factor for CIC is 64 during calibration time. DSADC_CIC_DECIMATION_RATE_8: Decimation factor for CIC is 8 during calibration time.		
Default value	DSADC_CIC_DECIMATION_RATE_8		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.2 DsadcCalibAlgoTargetValue

Table 16 Specification for DsadcCalibAlgoTargetValue

Name	`DsadcCalibAlgoTargetValue`		
Description	This parameter defines the full scale target value for calibration algorithm. Maximum target value allowed for this parameter depends on DsadcInputGain. When DsadcInputGain is selected as 2 or 4 then DsadcCalibAlgoTargetValue must not go beyond 22757. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 32767		
Default value	25000		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcInputGain		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.3 DsadcCalibCICFilterOutputShiftPos

Table 17 Specification for DsadcCalibCICFilterOutputShiftPos

Name	`DsadcCalibCICFilterOutputShiftPos`		
Description	This parameter defines the position of the CIC output shifter, which is to be used to select the valid output bits from the CIC filter. This is valid only during execution of calibration algorithm. Default value for this parameter is BITS_0_TO_16. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	BITS_i_TO_j: Valid output bit selection from the CIC filter output. Possible values of i and j are i : 0 to 28 j : 16 to 44		
Default value	BITS_0_TO_16		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.4 DsadcCalibGainCorrMulFactor

Table 18 Specification for DsadcCalibGainCorrMulFactor

Name	`DsadcCalibGainCorrMulFactor`		
Description	This parameter defines multiplication factor for Gain correction. This is valid only during execution of the calibration algorithm. The default value of this parameter is set to the reset value of the corresponding SFR. *Note: When computing the GAINFACTOR using the formula listed below, the user should not include the multiplication factor of 4096 as it is taken care in the DSADC driver itself.* *<GAINFACTOR> = truncate(((2 * AFS / (N^3 * 4 * FM)) * 2^(CICSHIFT-14)) * 4096)* *For more details on the formula and the computations, please refer to the Hardware User Manual.*		
Multiplicity	1..1	**Type**	EcucFloatParamDef
Range	0.0000 - 1.9999		
Default value	1.0000		

(table continues...)

Table 18 (continued) Specification for DsadcCalibGainCorrMulFactor

Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.5 DsadcGainCalibMulFactor

Table 19 Specification for DsadcGainCalibMulFactor

Name	`DsadcGainCalibMulFactor`		
Description	This parameter defines multiplication factor for Gain calibration. This is valid only during the execution of calibration algorithm. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucFloatParamDef
Range	0.0000 - 1.9999		
Default value	1.0000		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5 Container: DsadcCarrierGeneratorConfiguration

This container contains the Carrier Generation related Parameters.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.5.1 DsadcCarrierFrequencyClockDiv

Table 20 Specification for DsadcCarrierFrequencyClockDiv

Name	`DsadcCarrierFrequencyClockDiv`

(table continues...)

Table 20 (continued) Specification for DsadcCarrierFrequencyClockDiv

Description	This parameter defines the divider factor, which is used to define the frequency of the carrier signal generator, which is derived from the selected internal clock source and divider factor. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_CG_CLOCKDIVIDER_DIV10: Input clock is divided by 10 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV12: Input clock is divided by 12 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV14: Input clock is divided by 14 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV16: Input clock is divided by 16 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV18: Input clock is divided by 18 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV20: Input clock is divided by 20 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV22: Input clock is divided by 22 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV24: Input clock is divided by 24 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV26: Input clock is divided by 26 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV28: Input clock is divided by 28 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV2: Input clock is divided by 2 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV30: Input clock is divided by 30 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV32: Input clock is divided by 32 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV4: Input clock is divided by 4 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV6: Input clock is divided by 6 for carrier generator DSADC_CG_CLOCKDIVIDER_DIV8: Input clock is divided by 8 for carrier generator		
Default value	DSADC_CG_CLOCKDIVIDER_DIV2		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.2 DsadcCarrierSignalPolarity

Table 21 Specification for DsadcCarrierSignalPolarity

Name	`DsadcCarrierSignalPolarity`		
Description	This parameter defines the starting polarity of the carrier signal. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef

(table continues...)

Table 21 (continued) Specification for DsadcCarrierSignalPolarity

Range	DSADC_CARR_SIG_INVERTED: Carrier signal begins with -1 i.e. LOW DSADC_CARR_SIG_NORMAL: Carrier signal begins with +1 i.e. HIGH		
Default value	DSADC_CARR_SIG_NORMAL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.3 DsadcCarrierSignalType

Table 22 Specification for DsadcCarrierSignalType

Name	`DsadcCarrierSignalType`		
Description	This parameter determines the carrier signal type to be generated. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_CARR_SIG_SINEWAVE: Carrier Generator generates the Sinewave. DSADC_CARR_SIG_SQUAREWAVE: Carrier Generator generates the Square wave. DSADC_CARR_SIG_STOPPED: Carrier signals are stopped. DSADC_CARR_SIG_TRIANGLE: Carrier Generator generates the Triangle wave		
Default value	DSADC_CARR_SIG_STOPPED		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.4 DsadcPwmGenerationMode

Table 23 Specification for DsadcPwmGenerationMode

Name	`DsadcPwmGenerationMode`

(table continues...)

Table 23 (continued) Specification for DsadcPwmGenerationMode

Description	This parameter defines the mode in which the Carrier Generator signal is generated. In case of Bit Reverse generation mode, it increases the frequency spectrum to yield a smoother induced sine signal. This is done by distributing the 0 and 1 bits over the 32 cycles of a PWM period. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_BIT_REVERSE_MODE: Carrier generated in bit reverse mode DSADC_NORMAL_MODE: Normal Carrier generation		
Default value	DSADC_NORMAL_MODE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6 Container: DsadcChannelConfiguration

This container contains the channel configuration (parameters) depending on the hardware capability.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

1.3.1.6.1 DsadcAccessMode

Table 24 Specification for DsadcAccessMode

Name	`DsadcAccessMode`		
Description	This parameter determines the Result Access Mode selection for a DSADC channel. The available access mode depends on the parameter DsadcTriggerMode. If DsadcTriggerMode is Configured as TRIGGER_MODE_WINDOW then DsadcAccessMode cannot be set as DSADC_CIRCULAR_BUFFER. The default value of this parameter is set to DSADC_SINGLE_READ to minimize the execution time of ISR		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef

(table continues...)

Table 24 **(continued) Specification for DsadcAccessMode**

Range	DSADC_CIRCULAR_BUFFER: Configure buffer as circular buffer (i.e. the DSADC Driver wraps around if the end of the stream buffer is reached) DSADC_DMA_ACCESS: Result data has to be transferred via DMA DSADC_SINGLE_READ: Hardware result register value is read and returned to the user without buffering. DSADC_STREAM_LINEAR_BUFFER: Configure buffer as linear buffer (i.e. Once the Buffer is full, the subsequent results are discarded.).		
Default value	DSADC_SINGLE_READ		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcTriggerMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.2 DsadcBufferFullNotification

Table 25 **Specification for DsadcBufferFullNotification**

Name	`DsadcBufferFullNotification`		
Description	Callback function for buffer full event. This parameter is set to the default value when the parameter DsadcAccessMode is not configured as DSADC_STREAM_LINEAR_BUFFER Dsadc_NotifyFnPtrType is the data type for this callback function. By default, the notification parameter will be NULL_PTR , to remove the dependency from the user defined functions. The DSADC driver does not validate the configured function name or address for correctness and hence the responsibility falls on the user.		
Multiplicity	1..1	**Type**	EcucFunctionNameDef
Range	String		
Default value	NULL_PTR		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL

(table continues...)

Table 25 (continued) Specification for DsadcBufferFullNotification

Dependency	DsadcAccessMode
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.1.6.3 DsadcChannelId

Table 26 Specification for DsadcChannelId

Name	`DsadcChannelId`		
Description	Unique number to identify the channel. The Default value for this parameter is the index value for the container.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - (Number of channels - 1) where Number of channels depends on derivative		
Default value	Index value		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.4 DsadcGateActiveLevel

Table 27 Specification for DsadcGateActiveLevel

Name	`DsadcGateActiveLevel`		
Description	This parameter is used to define the active level for the gate signal. If this Parameter is configured as DSADC_GATE_HIGH_LEVEL then high level of the gate signal is considered as active phase. If this Parameter Configured as DSADC_GATE_LOW_LEVEL then low level of the gate signal is considered as the active phase. This Parameter is configurable only if the Parameter DsadcTriggerMode is configured as DSADC_TRIGGER_MODE_WINDOW. The default value of this parameter is set to DSADC_GATE_HIGH_LEVEL since many applications want to acquire result when the signal is High		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_GATE_HIGH_LEVEL: The Gate is active when the signal level is High DSADC_GATE_LOW_LEVEL: The Gate is active when the signal level is Low		

(table continues...)

Table 27 **(continued) Specification for DsadcGateActiveLevel**

Default value	DSADC_GATE_HIGH_LEVEL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcTriggerMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.5 DsadcHwChannelNum

Table 28 **Specification for DsadcHwChannelNum**

Name	`DsadcHwChannelNum`		
Description	Hardware EDSADC channel number. The default value for this parameter is set to DSADC_CHANNEL_0, since it is the first physical channel number.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_CHANNEL_x: where x stands for the channel number.		
Default value	DSADC_CHANNEL_0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.6 DsadcNewResultNotification

Table 29 **Specification for DsadcNewResultNotification**

Name	`DsadcNewResultNotification`

(table continues...)

Table 29 (continued) Specification for DsadcNewResultNotification

Description	Callback function for new result event. This parameter is set to the default value when the parameter DsadcAccessMode is configured as DSADC_DMA_ACCESS. Dsadc_NotifyFnPtrType is the data type for this callback function. By default, the notification parameter will be NULL_PTR , to remove the dependency from the user defined functions. The DSADC driver does not validate the configured function name or address for correctness and hence the responsibility falls on the user.		
Multiplicity	1..1	**Type**	EcucFunctionNameDef
Range	String		
Default value	NULL_PTR		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcAccessMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.7 DsadcTimestampFeature

Table 30 Specification for DsadcTimestampFeature

Name	`DsadcTimestampFeature`
Description	This parameter is used to define the timestamp function availability for the DSADC Channel. The Timestamp Functionality varies depending upon the access mode. If the parameter DsadcTriggerMode is configured as DSADC_TRIGGER_MODE_WINDOW then the timestamp is the, Timestamp count from the last HW result event till the gate event. The gate event can be window open or close event depends on the configuration parameter DsadcTimestampTriggerMode. If the Parameter DsadcAccessMode is configured as DSADC_SINGLE_READ and the DsadcTriggerMode is configured as DSADC_TRIGGER_MODE_NORMAL then the timestamp is the, Timestamp count from the result event till the Dsadc_ReadResult API reads the result value from the HW result register. Error will be raised when this parameter is configured as DSADC_TIMESTAMP_ENABLED and the DsadcTriggerMode is configured as DSADC_TRIGGER_MODE_NORMAL and DsadcAccessMode is not configured as DSADC_SINGLE_READ. The default value for this parameter is set to DSADC_TIMESTAMP_DISABLED to avoid the configuration dependency error for the default configuration.

(table continues...)

Table 30 **(continued) Specification for DsadcTimestampFeature**

Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_TIMESTAMP_DISABLED: Disable the timestamp functionality DSADC_TIMESTAMP_ENABLED: Enable the timestamp functionality		
Default value	DSADC_TIMESTAMP_DISABLED		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcTriggerMode , DsadcAccessMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.8 DsadcTriggerMode

Table 31 **Specification for DsadcTriggerMode**

Name	`DsadcTriggerMode`		
Description	This parameter is used to define the trigger mode. Possible Result data acquisition modes are DSADC_TRIGGER_MODE_NORMAL and DSADC_TRIGGER_MODE_WINDOW. The default value of this parameter is set to DSADC_TRIGGER_MODE_NORMAL to avoid the configuration dependency error for the default configuration.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_TRIGGER_MODE_NORMAL: Result data acquisition starts after calling Dsadc_StartModulation() API. DSADC_TRIGGER_MODE_WINDOW: Result data acquisition starts after the gate signal (GTM, ERU) is open.		
Default value	DSADC_TRIGGER_MODE_NORMAL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.9 DsadcWindowCloseNotification

Table 32 Specification for DsadcWindowCloseNotification

Name	`DsadcWindowCloseNotification`		
Description	Callback function for window close event. This Parameter is set to the default value if the Parameter DsadcTriggerMode is configured as DSADC_TRIGGER_MODE_NORMAL. Dsadc_NotifyFnPtrType is the data type for this callback function. By default, the notification parameter will be NULL_PTR , to remove the dependency from the user defined functions. The DSADC driver does not validate the configured function name or address for correctness and hence the responsibility falls on the user.		
Multiplicity	1..1	**Type**	EcucFunctionNameDef
Range	String		
Default value	NULL_PTR		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcTriggerMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7 Container: DsadcCommonModeVoltConfig

This container provides configuration parameters related to common mode voltage application to the inputs of the channel modulator.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.7.1 DsadcComModeVoltNegAEnable

Table 33 Specification for DsadcComModeVoltNegAEnable

Name	`DsadcComModeVoltNegAEnable`
Description	This parameter defines if the negative analog line connected to position A in the MUX needs to be connected to the Common Mode Voltage. This parameter is set to the default value if the Parameter DsadcCommonModeVoltageEnable is configured as False or the selected Hardware Channel DsadcHwChannelNum does not have the connection. The default value of this parameter is set to the reset value of the corresponding SFR.

(table continues...)

Table 33 (continued) Specification for DsadcComModeVoltNegAEnable

Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcCommonModeVoltageEnable, DsadcHwChannelNum		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7.2 DsadcComModeVoltNegBEnable

Table 34 Specification for DsadcComModeVoltNegBEnable

Name	`DsadcComModeVoltNegBEnable`		
Description	This parameter defines if the negative analog line connected to position B in the MUX needs to be connected to the common mode voltage. This Parameter is set to the default value if the Parameter DsadcCommonModeVoltageEnable is configured as False or the selected Hardware Channel DsadcHwChannelNum does not have the connection. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcCommonModeVoltageEnable, DsadcHwChannelNum		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7.3 DsadcComModeVoltNegCEnable

Table 35 Specification for DsadcComModeVoltNegCEnable

Name	`DsadcComModeVoltNegCEnable`		
Description	This parameter defines if the negative analog line connected to position C in the MUX needs to be connected to the Common Mode Voltage. This Parameter is set to the default value if the Parameter DsadcCommonModeVoltageEnable is configured as False or the selected Hardware Channel DsadcHwChannelNum does not have the connection. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcCommonModeVoltageEnable, DsadcHwChannelNum		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7.4 DsadcComModeVoltNegDEnable

Table 36 Specification for DsadcComModeVoltNegDEnable

Name	`DsadcComModeVoltNegDEnable`		
Description	This parameter defines if the negative analog line connected to position D in the MUX needs to be connected to the Common Mode Voltage. This Parameter is set to the default value if the Parameter DsadcCommonModeVoltageEnable is configured as False or the selected Hardware Channel DsadcHwChannelNum does not have the connection. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-

(table continues...)

Table 36 (continued) Specification for DsadcComModeVoltNegDEnable

Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcCommonModeVoltageEnable, DsadcHwChannelNum		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7.5 DsadcComModeVoltPosAEnable

Table 37 Specification for DsadcComModeVoltPosAEnable

Name	`DsadcComModeVoltPosAEnable`		
Description	This parameter defines if the positive analog line connected to position A in the MUX needs to be connected to the Common Mode Voltage. This Parameter is set to the default value if the Parameter DsadcCommonModeVoltageEnable is configured as False or the selected Hardware Channel DsadcHwChannelNum does not have the connection. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcCommonModeVoltageEnable, DsadcHwChannelNum		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7.6 DsadcComModeVoltPosBEnable

Table 38 Specification for DsadcComModeVoltPosBEnable

Name	`DsadcComModeVoltPosBEnable`

(table continues...)

Table 38 (continued) Specification for DsadcComModeVoltPosBEnable

Description	This parameter defines if the positive analog line connected to position B in the MUX needs to be connected to the Common Mode Voltage. This Parameter is set to the default value if the Parameter DsadcCommonModeVoltageEnable is configured as False or the selected Hardware Channel DsadcHwChannelNum does not have the connection. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcCommonModeVoltageEnable, DsadcHwChannelNum		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7.7 DsadcComModeVoltPosCEnable

Table 39 Specification for DsadcComModeVoltPosCEnable

Name	`DsadcComModeVoltPosCEnable`		
Description	This parameter defines if the positive analog line connected to position C in the MUX needs to be connected to the Common Mode Voltage. This Parameter is set to the default value if the Parameter DsadcCommonModeVoltageEnable is configured as False or the selected Hardware Channel DsadcHwChannelNum does not have the connection. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-

(table continues...)

Table 39 (continued) Specification for DsadcComModeVoltPosCEnable

Origin	IFX	**Scope**	LOCAL
Dependency	DsadcCommonModeVoltageEnable, DsadcHwChannelNum		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7.8 DsadcComModeVoltPosDEnable

Table 40 Specification for DsadcComModeVoltPosDEnable

Name	`DsadcComModeVoltPosDEnable`		
Description	This parameter defines if the positive analog line connected to position D in the MUX needs to be connected to the Common Mode Voltage. This Parameter is set to the default value if the Parameter DsadcCommonModeVoltageEnable is configured as False or the selected Hardware Channel DsadcHwChannelNum does not have the connection. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcCommonModeVoltageEnable, DsadcHwChannelNum		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7.9 DsadcCommonModeVoltageEnable

Table 41 Specification for DsadcCommonModeVoltageEnable

Name	`DsadcCommonModeVoltageEnable`		
Description	This parameter defines the availability of Common Mode voltage to the input pins. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		

(table continues...)

Table 41 (continued) Specification for DsadcCommonModeVoltageEnable

Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7.10 DsadcCommonModeVoltageSelect

Table 42 Specification for DsadcCommonModeVoltageSelect

Name	`DsadcCommonModeVoltageSelect`		
Description	This parameter defines the voltage level configured as Common Mode voltage, which can be connected to the input pins. This Parameter is set to the Default value if the Parameter DsadcCommonModeVoltageEnable is configured as False. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_VCM_VREFX_16: Common mode voltage is configured as (VREFX/16) DSADC_VCM_VREFX_2: Common mode voltage is configured as (VREFX/2) DSADC_VCM_VREFX_4: Common mode voltage is configured as (VREFX/4) DSADC_VCM_VREFX_8: Common mode voltage is configured as (VREFX/8)		
Default value	DSADC_VCM_VREFX_2		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcCommonModeVoltageEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.8 Container: DsadcComparatorConfiguration

This container provides configuration parameters for comparator configuration.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.8.1 DsadcComparatorEventSelect

Table 43 Specification for DsadcComparatorEventSelect

Name	`DsadcComparatorEventSelect`		
Description	This parameter defines the comparator mode selected to generate an alarm event (and also a service request, if the alternate service request is enabled). The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_RESULT_ALWAYS: Event is generated on each new result generated. DSADC_RESULT_INSIDE_RANGE: Event is generated if the result is within the boundaries defined. DSADC_RESULT_OUTSIDE_RANGE: Event is generated if the result is outside the boundaries defined.		
Default value	DSADC_RESULT_ALWAYS		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.8.2 DsadcLowerBoundaryValue

Table 44 Specification for DsadcLowerBoundaryValue

Name	`DsadcLowerBoundaryValue`		
Description	This parameter defines the lower boundary used for limit checking. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	-32768 - +32767		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL

(table continues...)

Table 44 (continued) Specification for DsadcLowerBoundaryValue

Dependency	-
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.1.8.3 DsadcUpperBoundaryValue

Table 45 Specification for DsadcUpperBoundaryValue

Name	`DsadcUpperBoundaryValue`		
Description	This parameter defines the upper boundary used for limit checking. The value of this Parameter should be greater than the parameter DsadcLowerBoundaryValue. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	-32768 - +32767		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcLowerBoundaryValue		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.9 Container: DsadcConfigSet

This is the base container that contains Dsadc module related parameters

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.10 Container: DsadcDemEventParameterRefs

Container list down the production errors supported by the DSADC driver. This container must be present when safety check is enabled.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

1.3.1.10.1 DsadcClcFailureNotification

Table 46 Specification for DsadcClcFailureNotification

Name	`DsadcClcFailureNotification`

(table continues...)

Table 46 (continued) Specification for DsadcClcFailureNotification

Description	Parameter defines whether CLC failure Production Error notification is enabled or not. This parameter must be present when safety check is enabled. The default value of this parameter is set to NULL to minimize the executable code size.		
Multiplicity	0..1	**Type**	EcucReferenceDef
Range	Reference to Node: DemEventParameter		
Default value	NULL		
Post-build variant value	FALSE	**Post-build variant multiplicity**	FALSE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	Pre-Compile
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcSafetyEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.10.2 DsadcFifoFailureNotification

Table 47 Specification for DsadcFifoFailureNotification

Name	`DsadcFifoFailureNotification`		
Description	Parameter defines whether HW FIFO failure Production Error notification is enabled or not. This parameter must be present when safety check is enabled. The default value of this parameter is set to NULL to minimize the executable code size.		
Multiplicity	0..1	**Type**	EcucReferenceDef
Range	Reference to Node: DemEventParameter		
Default value	NULL		
Post-build variant value	FALSE	**Post-build variant multiplicity**	FALSE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	Pre-Compile
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcSafetyEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.11 Container: DsadcDemodulatorConfiguration

This container contains configuration parameters related to the de-modulator, input data stream and trigger selection.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.11.1 DsadcIntegratorTriggerMode

Table 48 Specification for DsadcIntegratorTriggerMode

Name	`DsadcIntegratorTriggerMode`		
Description	This parameter defines the Integrator trigger mode. If DsadcTriggerMode is configured as TRIGGER_MODE_WINDOW and the DsadcGateActiveLevel is configures as HIGH, then DsadcIntegratorTriggerMode should be DSADC_INTR_RISING_EDGE or DSADC_INTR_BYPASSED. If DsadcTriggerMode is configured as TRIGGER_MODE_WINDOW and the DsadcGateActiveLevel is configures as LOW, then DsadcIntegratorTriggerMode should be DSADC_INTR_FALLING_EDGE or DSADC_INTR_BYPASSED. If DsadcTriggerMode is configured as TRIGGER_MODE_NORMAL then DsadcIntegratorTriggerMode should be Configured as DSADC_INTR_BYPASSED or DSADC_INTR_ALWAYS_ACTIVE The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_INTR_ALWAYS_ACTIVE: No trigger required, integrator is active all the time DSADC_INTR_BYPASSED: No integration trigger, integrator bypassed DSADC_INTR_FALLING_EDGE: Trigger event upon a falling edge i.e. Integrator is activated on the falling edge of the trigger. DSADC_INTR_RISING_EDGE: Trigger event upon a rising edge i.e. Integrator is activated on the rising edge of the trigger.		
Default value	DSADC_INTR_BYPASSED		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcTriggerMode , DsadcGateActiveLevel		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.11.2 DsadcResultDisplayMode

Table 49 Specification for DsadcResultDisplayMode

Name	`DsadcResultDisplayMode`		
Description	This parameter defines the ranges of the result values i.e. result display modes. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef

(table continues...)

Table 49 **(continued) Specification for DsadcResultDisplayMode**

Range	DSADC_RES_SIGNED_MODE: The result values range is -32768 to +32767 DSADC_RES_UNSIGNED_MODE: The result values range is 0 to +65535		
Default value	DSADC_RES_SIGNED_MODE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.11.3 DsadcTriggerSelect

Table 50 **Specification for DsadcTriggerSelect**

Name	`DsadcTriggerSelect`		
Description	This parameter defines EDSADC channel trigger source. The Trigger source is depends on the DSADC Hardware Channel selected by the parameter DsadcHwChannelNum. If the Parameter DsadcTriggerMode is configured as DSADC_TRIGGER_MODE_WINDOW then the parameter DsadcTriggerSelect should not be configured as TRIGGER_0_NO_DSADC_TRIG. If the Parameter DsadcTriggerMode is configured as DSADC_TRIGGER_MODE_NORMAL then the parameter DsadcTriggerSelect should be configured as TRIGGER_0_NO_DSADC_TRIG. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	TRIGGER_0_NO_DSADC_TRIG: No trigger is selected for DSADC channel. TRIGGER_x_y: x: The available number of trigger selection. y: depends on the DSADC channel selected and trigger input selected from the available selection.		
Default value	TRIGGER_0_NO_DSADC_TRIG		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcTriggerMode , DsadcHwChannelNum		

(table continues...)

Table 50 **(continued) Specification for DsadcTriggerSelect**

Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.1.12 Container: DsadcErsEtlConfig

This container configures the parameters for ERU input triggering.

If the Parameter DsadcTriggerSelect selects the ERU as a Trigger source then only this container should be present.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Post-Build

1.3.1.12.1 DsadcEruErsInputPin

Table 51 **Specification for DsadcEruErsInputPin**

Name	`DsadcEruErsInputPin`		
Description	This parameter determines the input pin for the selected ERS. The default value for this parameter is set depends on the configured ERS channel and the first physical input pin for the multiplexer.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	ERS_X_Y: X stands for the input connection number Y stands for the input source		
Default value	ERS_X_Y		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.12.2 DsadcEruErsRef

Table 52 **Specification for DsadcEruErsRef**

Name	`DsadcEruErsRef`

(table continues...)

Table 52 (continued) Specification for DsadcEruErsRef

Description	This parameter is a reference to the ERU container in the MCU. It lists down all the ERU-ERS channels available. If referred ERU-ERS channel is not marked as ERU_CHANNEL_INP_USED_BY_DSADC_DRIVER in MCU an error message will be raised. If the ERU ERS input channel referenced in one container is already referenced in another container, then an error message is provided. The default value for this parameter is set to NULL to avoid the configuration dependency error for the default configuration		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: McuEruChannelInputLineConf		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.12.3 DsadcEruStatusFlagConfig

Table 53 Specification for DsadcEruStatusFlagConfig

Name	`DsadcEruStatusFlagConfig`		
Description	This parameters defines the condition on which the status flag in ETL block of ERU is set. On the inverse of the edge selected, the status flag is cleared. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_ETL_FALLING_EDGE: Status flag of ERU channel is set on the detection of a falling edge on input channel DSADC_ETL_RISING_EDGE: Status flag of ERU channel is set on the detection of a rising edge on input channel		
Default value	DSADC_ETL_FALLING_EDGE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-

(table continues...)

Table 53 (continued) Specification for DsadcEruStatusFlagConfig

Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.13 Container: DsadcFilterConfiguration

This container provides configuration parameter related to main filter chain.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.13.1 DsadcAlternateServiceReq

Table 54 Specification for DsadcAlternateServiceReq

Name	`DsadcAlternateServiceReq`		
Description	This parameter is used to generate alternate service request for any one of the following events, comparator event, time stamp event or alternate source. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_ALT_SERVICE_DISABLE: Service request is disabled. DSADC_COMPARATOR_EVENT: comparator event generates the service request. DSADC_RESOLVER_EVENT: Alternate source (capturing of a sign delay value to register carrier generator synchronization register) Service request. DSADC_TIMESTAMP_EVENT: Timestamp event generates service request.		
Default value	DSADC_ALT_SERVICE_DISABLE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.13.2 DsadcCICFilterDecimationFactor

Table 55 Specification for DsadcCICFilterDecimationFactor

Name	`DsadcCICFilterDecimationFactor`

(table continues...)

Table 55 (continued) Specification for DsadcCICFilterDecimationFactor

Description	This parameter defines the oversampling rate/Decimation factor for the CIC filter. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	4 - 512		
Default value	4		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.13.3 DsadcCICFilterStartValue

Table 56 Specification for DsadcCICFilterStartValue

Name	`DsadcCICFilterStartValue`		
Description	This parameter defines the starting value of decimation counter, when the CIC filter is started/restarted. If the value of DsadcCICFilterStartValue is set higher than the value of DsadcCICFilterDecimationFactor, then an error message is provided. Starting value exceeding the value specified in DsadcCICFilterDecimationFactor can lead to overflow of CIC filter. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	4 - 512		
Default value	4		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcCICFilterDecimationFactor		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.13.4 DsadcFIR0FilterEnable

Table 57 Specification for DsadcFIR0FilterEnable

Name	`DsadcFIR0FilterEnable`		
Description	This parameter defines the availability of FIR0 filter in the filter chain of the DSADC channel. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.13.5 DsadcFIR1FilterDecimationEnable

Table 58 Specification for DsadcFIR1FilterDecimationEnable

Name	`DsadcFIR1FilterDecimationEnable`		
Description	This parameter defines the decimation rate of FIR1 filter. If selected as TRUE, then the filter decimates with a ratio 2:1. This Parameter takes the default value when the parameter DsadcFIR1FilterEnable set to FALSE. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL

(table continues...)

Table 58 (continued) Specification for DsadcFIR1FilterDecimationEnable

Dependency	DsadcFIR1FilterEnable
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.1.13.6 DsadcFIR1FilterEnable

Table 59 Specification for DsadcFIR1FilterEnable

Name	`DsadcFIR1FilterEnable`		
Description	This parameter defines the availability of FIR1 filter in the filter chain of the DSADC channel. This Parameter takes the default value when the parameter DsadcFIR0FilterEnable set to FALSE. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcFIR0FilterEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.13.7 DsadcOffsetCompFilterEnable

Table 60 Specification for DsadcOffsetCompFilterEnable

Name	`DsadcOffsetCompFilterEnable`		
Description	This parameter indicates availability of an Offset Compensation (IIR) filter (and its various operation modes) in Offset Compensation block. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef

(table continues...)

Table 60 (continued) Specification for DsadcOffsetCompFilterEnable

Range	DSADC_OFFCOMP_FILTER_DISABLE: Offset compensation filter is disabled. DSADC_OFFCOMP_FILTER_RATE_1: Offset compensation filter is enabled and it adjusts OFFCOMP register with offset Compensation filter with Rate 1 DSADC_OFFCOMP_FILTER_RATE_2: Offset compensation filter is enabled and it adjusts OFFCOMP register with offset Compensation filter with Rate 2 DSADC_OFFCOMP_FILTER_RATE_3: Offset compensation filter is enabled and it adjusts OFFCOMP register with offset Compensation filter with Rate 3 DSADC_OFFCOMP_FILTER_RATE_4: Offset compensation filter is enabled and it adjusts OFFCOMP register with offset Compensation filter with Rate 4 DSADC_OFFCOMP_FILTER_RATE_5: Offset compensation filter is enabled and it adjusts OFFCOMP register with offset Compensation filter with Rate 5 DSADC_OFFCOMP_FILTER_RATE_6: Offset compensation filter is enabled and it adjusts OFFCOMP register with offset Compensation filter with Rate 6 DSADC_OFFCOMP_FILTER_RATE_7: Offset compensation filter is enabled and it adjusts OFFCOMP register with offset Compensation filter with Rate 7		
Default value	DSADC_OFFCOMP_FILTER_DISABLE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.13.8 DsadcOffsetCompValue

Table 61 Specification for DsadcOffsetCompValue

Name	`DsadcOffsetCompValue`		
Description	This parameter defines the offset component value to be removed (subtracted) from each result from the filter chain. This Parameter is set to the default value '0' if the parameter DsadcOffsetCompFilterEnable is not configured as DSADC_OFFCOMP_FILTER_DISABLE . The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	-32768 - +32767		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-

(table continues...)

Table 61 (continued) Specification for DsadcOffsetCompValue

Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcOffsetCompFilterEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.13.9 DsadcOffsetCompValueProtect

Table 62 Specification for DsadcOffsetCompValueProtect

Name	`DsadcOffsetCompValueProtect`		
Description	This parameter defines the protection of the Offset Compensation register from the calibration algorithm. This Parameter should not set to false when the parameter DsadcOffsetCompFilterEnable is not configured as DSADC_OFFCOMP_FILTER_DISABLE. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcOffsetCompFilterEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.13.10 DsadcOvershootCompensationEn

Table 63 Specification for DsadcOvershootCompensationEn

Name	`DsadcOvershootCompensationEn`		
Description	This parameter defines the availability of Overshoot compensation block in the filter chain of the DSADC channel. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef

(table continues...)

Table 63 (continued) Specification for DsadcOvershootCompensationEn

Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.13.11 DsadcPreFilterEnable

Table 64 Specification for DsadcPreFilterEnable

Name	`DsadcPreFilterEnable`		
Description	This parameter defines the availability of Prefilter in the filter chain of the DSADC channel The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.14 Container: DsadcGainCalibConfig

This container provides configuration parameters for Gain calibration.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.15 Container: DsadcGainCorrConfig

This container provides configuration parameters for Gain correction.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.15.1 DsadcCICFilterOutputShiftPos

Table 65 Specification for DsadcCICFilterOutputShiftPos

Name	`DsadcCICFilterOutputShiftPos`		
Description	This parameter defines the position of the CIC output shifter, which is to be used to select the valid output bits from the CIC filter. Default value for this parameter is BITS_0_TO_16. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	BITS_i_TO_j: Valid output bits selection from the CIC Filter output. Possible values of i and j are i : 0 to 28 j : 16 to 44		
Default value	BITS_0_TO_16		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.15.2 DsadcGainCorrMulFactor

Table 66 Specification for DsadcGainCorrMulFactor

Name	`DsadcGainCorrMulFactor`		
Description	This parameter defines multiplication factor for Gain correction. The default value of this parameter is set to the reset value of the corresponding SFR. *Note: When computing the GAINFACTOR using the formula listed below, the user should not include the multiplication factor of 4096 as it is taken care in the DSADC driver itself.* *<GAINFACTOR> = truncate(((2 * AFS / (N^3 * 4 * FM)) * 2^(CICSHIFT-14)) * 4096)* *For more details on the formula and the computations, please refer to the Hardware User Manual.*		
Multiplicity	1..1	**Type**	EcucFloatParamDef
Range	0.0000 - 1.9999		

(table continues...)

Table 66 **(continued) Specification for DsadcGainCorrMulFactor**

Default value	1.0000		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.16 Container: DsadcGeneral

This Container contains all the general configuration parameters for the DSADC driver

Post-Build Variant Multiplicity: -
Multiplicity Configuration Class: -

1.3.1.16.1 DsadcDeInitApi

Table 67 **Specification for DsadcDeInitApi**

Name	`DsadcDeInitApi`		
Description	This Parameter adds or removes the service Dsadc_DeInit() API from the code. The default value of this parameter is set to FALSE to minimize the executable code size.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.16.2 DsadcDevErrorDetect

Table 68 Specification for DsadcDevErrorDetect

Name	`DsadcDevErrorDetect`		
Description	This Parameter Enables/Disables the Development Error Detection and reporting in the DSADC Driver. The default value of this parameter is set to FALSE to minimize the executable code size.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.16.3 DsadcInitCheckApi

Table 69 Specification for DsadcInitCheckApi

Name	`DsadcInitCheckApi`		
Description	This Parameter adds or removes the service Dsadc_InitCheck() API from the code. The default value of this parameter is set to FALSE to minimize the executable code size.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.16.4 DsadcInitDeInitApiMode

Table 70 Specification for DsadcInitDeInitApiMode

Name	`DsadcInitDeInitApiMode`		
Description	This parameter defines the privilege mode in which the Initialization and De-initialization APIs would operate. Since the DSADC driver accesses the SFRs, it is efficient to operate the DSADC driver in supervisory mode than the USER1 mode. Hence, the default mode of operation is the supervisory mode.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_MCAL_SUPERVISOR: Operating mode used is Supervisory. DSADC_MCAL_USER1: Operating mode used is USER1.		
Default value	DSADC_MCAL_SUPERVISOR		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.16.5 DsadcRestartIntegratorApi

Table 71 Specification for DsadcRestartIntegratorApi

Name	`DsadcRestartIntegratorApi`		
Description	This Parameter adds or removes the service Dsadc_RestartIntegrator() from the code. Note: Carrier signal is also restarted along with the integrator.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	FALSE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL

(table continues...)

Table 71 (continued) Specification for DsadcRestartIntegratorApi

Dependency	-
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.1.16.6 DsadcRuntimeApiMode

Table 72 Specification for DsadcRuntimeApiMode

Name	`DsadcRuntimeApiMode`		
Description	This Parameter defines the privilege mode in which the runtime APIs would operate. Since the DSADC driver accesses the SFRs, it is efficient to operate the DSADC driver in supervisory mode. Hence, the default mode of operation is supervisory mode. DsadcRuntimeApiMode must be configured as User-1 mode if DsadcInitDeInitApiMode is configured as User-1 mode.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_MCAL_SUPERVISOR: Operating mode used is Supervisory. DSADC_MCAL_USER1: Operating mode used is USER1.		
Default value	DSADC_MCAL_SUPERVISOR		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcInitDeInitApiMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.16.7 DsadcSafetyEnable

Table 73 Specification for DsadcSafetyEnable

Name	`DsadcSafetyEnable`		
Description	This Parameter determines whether to Enable/Disable the safety check and reporting. The default value of this parameter is set TRUE to ensure that safety issues are addressed during the product lifecycle.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	TRUE		

(table continues...)

Table 73 (continued) Specification for DsadcSafetyEnable

Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.16.8 DsadcVersionInfoApi

Table 74 Specification for DsadcVersionInfoApi

Name	`DsadcVersionInfoApi`		
Description	This Parameter adds or removes the Dsadc_GetVersionInfo() API from the code. When set to TRUE, the API is available at runtime. The default value of this parameter is set to FALSE to minimize the executable code size.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17 Container: DsadcGlobalConfiguration

This container contains the parameters to configure DSADC IP global configuration.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.17.1 DsadcDitheringTrimValue

Table 75 Specification for DsadcDitheringTrimValue

Name	`DsadcDitheringTrimValue`

(table continues...)

Table 75 (continued) Specification for DsadcDitheringTrimValue

Description	This Parameter defines the trimming value for internal dithering function. This trimming value is used for all the modulators of DSADC. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_DITHERING_HIGH_400_MILVLT: Dithering intensity is high volts which is 400 millivolts. DSADC_DITHERING_LOW_100_MILVLT: Dithering intensity is Low volts which is 100 millivolts. DSADC_DITHERING_MED_200_MILVLT: Dithering intensity is medium volts which is 200 millivolts DSADC_DITHERING_MIN_50_MILVLT: Dithering intensity is minimum volts which is 50 millivolts.		
Default value	DSADC_DITHERING_MIN_50_MILVLT		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.2 DsadcSleepMode

Table 76 Specification for DsadcSleepMode

Name	`DsadcSleepMode`		
Description	This Parameter defines EDSADC reaction to the Sleep mode requests. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	SLEEP_DISABLE: Disable the sleep mode for DSADC module SLEEP_ENABLE: Enable the sleep mode for DSADC module		
Default value	SLEEP_ENABLE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL

(table continues...)

Table 76 **(continued) Specification for DsadcSleepMode**

Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.3 DsadcSupplyVoltageLevel

Table 77 **Specification for DsadcSupplyVoltageLevel**

Name	`DsadcSupplyVoltageLevel`		
Description	This Parameter defines the supply voltage level to be used for DSADC internal operations. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	VOLTAGESUPPLY_3_3V: 3.3V power supply is connected. VOLTAGESUPPLY_5V: 5V power supply is connected VOLTAGESUPPLY_AUTO: The voltage range is controlled by the power supply		
Default value	VOLTAGESUPPLY_AUTO		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.4 DsadcSyncClockGen

Table 78 **Specification for DsadcSyncClockGen**

Name	`DsadcSyncClockGen`		
Description	This Parameter defines the influence of Analog Phase synchronizer on the clock generated. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	SYNCHRONIZED_MODE: Rising clock edge is defined by the phase synchronizer. UNSYNCHRONIZED_MODE: Modulator clock is generated without the influence of phase synchronizer		
Default value	UNSYNCHRONIZED_MODE		

(table continues...)

Table 78 (continued) Specification for DsadcSyncClockGen

Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.18 Container: DsadcIntegratorConfiguration

This container provides configuration parameter related to the Integrator functional block.

If the Parameter DsadcIntegratorTriggerMode is set to DSADC_INTR_BYPASSED, then this container cannot be added.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Post-Build

1.3.1.18.1 DsadcDiscardCount

Table 79 Specification for DsadcDiscardCount

Name	`DsadcDiscardCount`		
Description	This parameter defines the number of result values to be discarded before the start of the integration cycle. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 63		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.18.2 DsadcIntegrationCount

Table 80 Specification for DsadcIntegrationCount

Name	`DsadcIntegrationCount`		
Description	This parameter defines the number of result values to be integrated. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	2 - 64		
Default value	2		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.19 Container: DsadcModulatorConfiguration

This container contains configuration parameters related to the On-chip modulator, input pin selection and modulator clock configuration.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.19.1 DsadcAnalogClockSyncDelay

Table 81 Specification for DsadcAnalogClockSyncDelay

Name	`DsadcAnalogClockSyncDelay`		
Description	This parameter defines the delay in clock cycles after the sync signal provided from Phase synchronizer. This Parameter is set to the default value when the parameter DsadcSyncClockGen is set to UNSYNCHRONIZED_MODE. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 7		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-

(table continues...)

Table 81 (continued) Specification for DsadcAnalogClockSyncDelay

Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcSyncClockGen		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.19.2 DsadcClockDivider

Table 82 Specification for DsadcClockDivider

Name	`DsadcClockDivider`		
Description	This parameter defines the divider factor, which defines the frequency of the modulator clock, which is derived from the peripheral clock. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_CLOCKDIVIDER_DIV10: Input clock is divided by 10 for modulator DSADC_CLOCKDIVIDER_DIV12: Input clock is divided by 12 for modulator DSADC_CLOCKDIVIDER_DIV14: Input clock is divided by 14 for modulator DSADC_CLOCKDIVIDER_DIV16: Input clock is divided by 16 for modulator DSADC_CLOCKDIVIDER_DIV18: Input clock is divided by 18 for modulator DSADC_CLOCKDIVIDER_DIV4: Input clock is divided by 4 for modulator DSADC_CLOCKDIVIDER_DIV6: Input clock is divided by 6 for modulator DSADC_CLOCKDIVIDER_DIV8: Input clock is divided by 8 for modulator		
Default value	DSADC_CLOCKDIVIDER_DIV4		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.19.3 DsadcDitheringEnable

Table 83 Specification for DsadcDitheringEnable

Name	`DsadcDitheringEnable`

(table continues...)

Table 83 (continued) Specification for DsadcDitheringEnable

Description	This parameter defines the availability of internal dithering functionality for the modulator. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.19.4 DsadcInputGain

Table 84 Specification for DsadcInputGain

Name	`DsadcInputGain`		
Description	This parameter defines multiplication Gain factor for the analog input signal. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_INPUT_GAIN_FACTOR_1: Gain factor is 1 DSADC_INPUT_GAIN_FACTOR_2: Gain factor is 2 DSADC_INPUT_GAIN_FACTOR_4: Gain factor is 4		
Default value	DSADC_INPUT_GAIN_FACTOR_1		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.19.5 DsadcInputMuxActionMode

Table 85 Specification for DsadcInputMuxActionMode

Name	`DsadcInputMuxActionMode`		
Description	This parameter defines the control mechanism for input multiplexer. It defines the action to be taken for the input multiplexer upon a trigger event for pin selection. This parameter is set to the default value if the Parameter DsadcTriggerMode is configured as DSADC_TRIGGER_MODE_NORMAL. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_INPUTMUX_PRESET_MODE: Load INMUX upon a trigger DSADC_INPUTMUX_SINGLE_STEP_MODE: Decrement INMUX value upon a trigger and wrap around the value specified in parameter DsadcInputPinSelection.		
Default value	DSADC_INPUTMUX_PRESET_MODE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcTriggerMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.19.6 DsadcInputMuxControlMode

Table 86 Specification for DsadcInputMuxControlMode

Name	`DsadcInputMuxControlMode`		
Description	This parameter defines the condition for a trigger event to control the input multiplexer. This parameter is set to the default value If the Parameter DsadcTriggerSelect is configured as DSADC_TRIGGER_MODE_NORMAL. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_INMUX_SOFTWARE_CONTROL: Multiplexer controlled by software DSADC_INMUX_TRIG_EVENT_BOTH_EDGES: Trigger event upon an any edge DSADC_INMUX_TRIG_EVENT_FALLING_EDGE: Trigger event upon a falling edge DSADC_INMUX_TRIG_EVENT_RISING_EDGE: Trigger event upon a raising edge		
Default value	DSADC_INMUX_SOFTWARE_CONTROL		

(table continues...)

Table 86 (continued) Specification for DsadcInputMuxControlMode

Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcTriggerMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.19.7 DsadcInputPinSelection

Table 87 Specification for DsadcInputPinSelection

Name	`DsadcInputPinSelection`		
Description	This parameter defines the initial/permanent setting for the input multiplexer, based on the operating mode selected. The Input pin selection is depends on the DSADC Hardware Channel selected by the parameter DsadcHwChannelNum. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	INPUT_PIN_[x]: where [x] stands for the pin selected for the selected channel		
Default value	INPUT_PIN_[x]		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcHwChannelNum		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.19.8 DsadcIntegratorResetEnable

Table 88 Specification for DsadcIntegratorResetEnable

Name	`DsadcIntegratorResetEnable`
Description	This parameter defines the modulator overload handling. This Parameter is set to the Default value when the Parameter DsadcIntegratorTriggerMode is configured as DSADC_INTR_BYPASSED The default value of this parameter is set to the reset value of the corresponding SFR.

(table continues...)

Table 88 (continued) Specification for DsadcIntegratorResetEnable

Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcIntegratorTriggerMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.19.9 DsadcNegativeInputLine

Table 89 Specification for DsadcNegativeInputLine

Name	`DsadcNegativeInputLine`		
Description	This parameter defines the modulator internal connection of the negative input. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_NEG_INPUT_PIN: Modulator negative input is connected to Input pin. DSADC_NEG_IN_COMMON_MODE_VOLT: Modulator negative input is connected to Common mode voltage V_{REFX}. DSADC_NEG_IN_REFERENCE_GROUND: Modulator negative input is connected to reference ground V_{AGND} DSADC_NEG_IN_SUPPLY_VOLT: Modulator negative input is connected to Supply voltage V_{AREF}.		
Default value	DSADC_NEG_INPUT_PIN		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.19.10 DsadcPositiveInputLine

Table 90 Specification for DsadcPositiveInputLine

Name	`DsadcPositiveInputLine`		
Description	This parameter defines the modulator internal connection of the positive input. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_POS_INPUT_PIN: Modulator negative input is connected to Input pin DSADC_POS_IN_COMMON_MODE_VOLT: Modulator negative input is connected to Common mode voltage V_{REFX}. DSADC_POS_IN_REFERENCE_GROUND: Modulator negative input is connected to reference ground V_{AGND} DSADC_POS_IN_SUPPLY_VOLT: Modulator negative input is connected to Supply voltage V_{AREF}		
Default value	DSADC_POS_INPUT_PIN		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.20 Container: DsadcOguConfig

This container configures the parameters for ERU Pattern Detection output.

This container can be added only if the Parameter DsadcTriggerSelect selecting the ERU signal as a trigger source.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Post-Build

1.3.1.20.1 DsadcEruErsCh0PatternFlagEnable

Table 91 Specification for DsadcEruErsCh0PatternFlagEnable

Name	`DsadcEruErsCh0PatternFlagEnable`
Description	This parameter determines of the ERU ERS channel 0 is used for pattern detection for the selected OGU channel The default value of this parameter is set to FALSE to avoid the configuration dependency error for the default configuration.

(table continues...)

Table 91 (continued) Specification for DsadcEruErsCh0PatternFlagEnable

Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.20.2 DsadcEruErsCh1PatternFlagEnable

Table 92 Specification for DsadcEruErsCh1PatternFlagEnable

Name	`DsadcEruErsCh1PatternFlagEnable`		
Description	This parameter determines of the ERU ERS channel 1 is used for pattern detection for the selected OGU channel. The default value of this parameter is set to FALSE to avoid the configuration dependency error for the default configuration.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.20.3 DsadcEruErsCh2PatternFlagEnable

Table 93 Specification for DsadcEruErsCh2PatternFlagEnable

Name	`DsadcEruErsCh2PatternFlagEnable`		
Description	This parameter determines of the ERU ERS channel 2 is used for pattern detection for the selected OGU channel. The default value of this parameter is set to FALSE to avoid the configuration dependency error for the default configuration.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.20.4 DsadcEruErsCh3PatternFlagEnable

Table 94 Specification for DsadcEruErsCh3PatternFlagEnable

Name	`DsadcEruErsCh3PatternFlagEnable`		
Description	This parameter determines of the ERU ERS channel 3 is used for pattern detection for the selected OGU channel. The default value of this parameter is set to FALSE to avoid the configuration dependency error for the default configuration.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL

(table continues...)

Table 94 (continued) Specification for DsadcEruErsCh3PatternFlagEnable

Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.20.5 DsadcEruErsCh4PatternFlagEnable

Table 95 Specification for DsadcEruErsCh4PatternFlagEnable

Name	`DsadcEruErsCh4PatternFlagEnable`		
Description	This parameter determines of the ERU ERS channel 4 is used for pattern detection for the selected OGU channel. The default value of this parameter is set to FALSE to avoid the configuration dependency error for the default configuration.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.20.6 DsadcEruErsCh5PatternFlagEnable

Table 96 Specification for DsadcEruErsCh5PatternFlagEnable

Name	`DsadcEruErsCh5PatternFlagEnable`		
Description	This parameter determines of the ERU ERS channel 5 is used for pattern detection for the selected OGU channel. The default value of this parameter is set to FALSE to avoid the configuration dependency error for the default configuration.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		

(table continues...)

Table 96 (continued) Specification for DsadcEruErsCh5PatternFlagEnable

Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.20.7 DsadcEruErsCh6PatternFlagEnable

Table 97 Specification for DsadcEruErsCh6PatternFlagEnable

Name	`DsadcEruErsCh6PatternFlagEnable`		
Description	This parameter determines of the ERU ERS channel 6 is used for pattern detection for the selected OGU channel. The default value of this parameter is set to FALSE to avoid the configuration dependency error for the default configuration.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.20.8 DsadcEruErsCh7PatternFlagEnable

Table 98 Specification for DsadcEruErsCh7PatternFlagEnable

Name	`DsadcEruErsCh7PatternFlagEnable`
Description	This parameter determines of the ERU ERS channel 7 is used for pattern detection for the selected OGU channel. The default value of this parameter is set to FALSE to avoid the configuration dependency error for the default configuration.

(table continues...)

Table 98 (continued) Specification for DsadcEruErsCh7PatternFlagEnable

Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.20.9 DsadcEruOguRef

Table 99 Specification for DsadcEruOguRef

Name	`DsadcEruOguRef`		
Description	This parameter is a reference to the ERU container in the MCU. It lists down all the ERU-OGU channels available. The OGUs available is dependent on the DsadcHwChannelNum. The Trigger source ERU PD_OUT selected by the DsadcTriggerSelect should be connected to the corresponding DSADC Channel.		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: McuEruChannelOutputUnitConf		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcHwChannelNum, DsadcTriggerSelect		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.21 Container: DsadcOvershootCompenConfig

This container provides configuration parameters for the Overshoot Compensation block.

If the Parameter DsadcOvershootCompensationEn is set to FALSE, then this container cannot be added
Post-Build Variant Multiplicity: TRUE
Multiplicity Configuration Class: Post-Build

1.3.1.21.1 DsadcSlewRateFilterRunTime

Table 100 Specification for DsadcSlewRateFilterRunTime

Name	`DsadcSlewRateFilterRunTime`		
Description	This parameter defines the time constant for the slew rate filter. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_SLEWRATE_FILTR_RUNTIME_16: Slew Rate Filter runs for 16 Input cycle DSADC_SLEWRATE_FILTR_RUNTIME_2: Slew Rate Filter runs for 2 Input cycle DSADC_SLEWRATE_FILTR_RUNTIME_4: Slew Rate Filter runs for 4 Input cycle DSADC_SLEWRATE_FILTR_RUNTIME_8: Slew Rate Filter runs for 8 Input cycle		
Default value	DSADC_SLEWRATE_FILTR_RUNTIME_2		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.21.2 DsadcSlewRateFilterStrength

Table 101 Specification for DsadcSlewRateFilterStrength

Name	`DsadcSlewRateFilterStrength`		
Description	This parameter defines the filter strength for the slew rate filter. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_MAXIMUM_FILTER_EFFECT: Maximum filter effect for slew rate Filter DSADC_MEDIUM_FILTER_EFFECT: Medium filter effect for slew rate Filter DSADC_MINIMUM_FILTER_EFFECT: Minimum filter effect for slew rate Filter DSADC_WEAK_FILTER_EFFECT: Weak filter effect for slew rate Filter		
Default value	DSADC_MINIMUM_FILTER_EFFECT		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-

(table continues...)

Table 101 (continued) Specification for DsadcSlewRateFilterStrength

Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.21.3 DsadcStepDetectionMode

Table 102 Specification for DsadcStepDetectionMode

Name	`DsadcStepDetectionMode`		
Description	This parameter defines the when the slew rate filter has to be activated. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_STEP_DETECT_CMP_LAST: Compare threshold to difference of current and last input to activate the slew rate filter DSADC_STEP_DETECT_CMP_SEC_LAST: Compare threshold to difference of current and second last input to activate the slew rate filter		
Default value	DSADC_STEP_DETECT_CMP_LAST		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.21.4 DsadcStepDetectionThreshold

Table 103 Specification for DsadcStepDetectionThreshold

Name	`DsadcStepDetectionThreshold`		
Description	This parameter defines the threshold value (magnitude) used for step detection. The threshold value is DsadcStepDetctionthreshold multiplied with 32. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 2047		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-

(table continues...)

Table 103 (continued) Specification for DsadcStepDetectionThreshold

Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.22 Container: DsadcRectificationConfiguration

This container provides configuration parameters of the rectifier block, used for resolver support, to determine position of the motor.

This Container is added only when the Parameter DsadcIntegratorTriggerMode is not configured as DSADC_INTR_BYPASSED.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Post-Build

1.3.1.22.1 DsadcNegSignDelayValue

Table 104 Specification for DsadcNegSignDelayValue

Name	`DsadcNegSignDelayValue`		
Description	This parameter determines the value of Sign Delay counter (number of result values), for which to generate a negative delayed sign signal. If the Parameter DsadcRectificationEnable is set to FALSE then this parameter is set to the Default value. This Parameter value needs to be configured greater than the parameter DsadcPosSignDelayValue. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcPosSignDelayValue, DsadcRectificationEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.22.2 DsadcPosSignDelayValue

Table 105 Specification for DsadcPosSignDelayValue

Name	`DsadcPosSignDelayValue`		
Description	This parameter determines the value of Sign Delay counter (number of result values), for which to generate a positive delayed sign signal. If the Parameter DsadcRectificationEnable is set to FALSE then this parameter takes the Default value. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcRectificationEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.22.3 DsadcRectificationEnable

Table 106 Specification for DsadcRectificationEnable

Name	`DsadcRectificationEnable`		
Description	This parameter controls the action of the rectifier circuit on the input data. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.22.4 DsadcSignSignalChannel

Table 107 Specification for DsadcSignSignalChannel

Name	`DsadcSignSignalChannel`		
Description	This parameter selects the DSADC channel which provides the source of the sign signal, which is to be delayed for the purpose of rectification. The Selection of DSADC Channel as a Sign Signal source is possible only when the Parameter DsadcSignSignalSource is set to SRC_1_SIGNRESULT_FROM_DSADC_CHANNEL and DsadcRectificationEnable is set to TRUE. The default value for this parameter is DSADC_CHANNEL_0 and it is the result value for the SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_CHANNEL_x: where x varies for channel number		
Default value	DSADC_CHANNEL_0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcSignSignalSource, DsadcRectificationEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.22.5 DsadcSignSignalSource

Table 108 Specification for DsadcSignSignalSource

Name	`DsadcSignSignalSource`		
Description	This parameter selects the source of the sign signal, which is to be delayed for the purpose of rectification. If the Parameter DsadcRectificationEnable is set to FALSE then this parameter takes the Default value. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef

(table continues...)

Table 108 (continued) Specification for DsadcSignSignalSource

Range	SRC_0_ON_CHIP_CARRIER_GENERATOR: Sign signal is generated from the On-chip carrier generator SRC_1_SIGNRESULT_FROM_DSADC_CHANNEL: The Sign signal is generated from one of the DSADC channels SRC_2_EXTERNAL_SIGN_SIGNAL_PORT_A: Sign signal is provide from external source, through Port Pin A 'A': Port pin number depends on the device. refer device property files SRC_3_EXTERNAL_SIGN_SIGNAL_PORT_B: Sign signal is provide from external source, through Port Pin B 'B': Port pin number depends on the device. refer device property files		
Default value	SRC_0_ON_CHIP_CARRIER_GENERATOR		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcRectificationEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.23 Container: DsadcTimestampConfiguration

This container provides configuration parameters regarding the timestamp counter configuration.

This Container is added only when the Parameter DsadcTimestampFeature is configured as DSADC_TIMESTAMP_ENABLED.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Post-Build

1.3.1.23.1 DsadcInputMuxSetCopyEnable

Table 109 Specification for DsadcInputMuxSetCopyEnable

Name	`DsadcInputMuxSetCopyEnable`		
Description	This parameter defines the availability of Analog MUX setting in the timestamp information. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		

(table continues...)

Table 109 (continued) Specification for DsadcInputMuxSetCopyEnable

Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.23.2 DsadcTimestampCounterClockSel

Table 110 Specification for DsadcTimestampCounterClockSel

Name	`DsadcTimestampCounterClockSel`		
Description	This parameter defines the divider factor, which defines the clock used for Timestamp counter. The default value of this parameter is set to the reset value of the corresponding SFR.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	CLOCKDIVIDER_DIV1: Modulator clock is divided by 1 for timestamp counter increment CLOCKDIVIDER_DIV2: Modulator clock is divided by 2 for timestamp counter increment CLOCKDIVIDER_DIV4: Modulator clock is divided by 4 for timestamp counter increment CLOCKDIVIDER_DIV8: Modulator clock is divided by 8 for timestamp counter increment		
Default value	CLOCKDIVIDER_DIV1		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.23.3 DsadcTimestampTriggerMode

Table 111 Specification for DsadcTimestampTriggerMode

Name	`DsadcTimestampTriggerMode`

(table continues...)

Table 111 (continued) Specification for DsadcTimestampTriggerMode

Description	This parameter is used to define whether the timestamp has to be captured for window open event or window close event. Error will be raised when this parameter is configured as DSADC_TIMESTAMP_WINDOWCLOSE and DsadcAccessMode is configured as DSADC_SINGLE_READ. The default value of this parameter is set to DSADC_TIMESTAMP_WINDOWOPEN since many applications want to acquire timestamp for the window open event.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DSADC_TIMESTAMP_WINDOWCLOSE: Timestamp shall be captured for window close event. DSADC_TIMESTAMP_WINDOWOPEN: Timestamp shall be captured for window open event.		
Default value	DSADC_TIMESTAMP_WINDOWOPEN		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	DsadcAccessMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.2 Functions - Type definitions

This section lists all the data type of the DSADC driver.

1.3.2.1 Dsadc_CalibrationStatusType

Table 112 Specification for Dsadc_CalibrationStatusType

Syntax	`Dsadc_CalibrationStatusType`	
Type	uint8	
File	`Dsadc.h`	
Range	0 - DSADC_CALIBRATION_NOT_STARTED	The DSADC channel Calibration activity not started.
	1 - DSADC_CALIBRATION_RUNNING	Calibration algorithm is currently running
	2 - DSADC_CALIBRATION_DONE	Calibration is completed. Normal operations is possible
	3 - DSADC_CALIBRATION_ERROR	Calibration terminated incorrectly.
Description	This datatype is used to define the various states of the calibration algorithm for the DSADC channel.	

(table continues...)

Table 112 (continued) Specification for Dsadc_CalibrationStatusType

Source	IFX
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.2.2 Dsadc_ChannelMaskType

Table 113 Specification for Dsadc_ChannelMaskType

Syntax	`Dsadc_ChannelMaskType`	
Type	uint32	
File	`Dsadc.h`	
Range	0 to 4294967295	
Description	Bit coded information for the DSADC logical channel numbers. DSADC logical channel is indicated by the corresponding bit position. For example if the bit position 0 is set means logical channel ID 0 is requested.	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.3 Dsadc_ChannelStatusType

Table 114 Specification for Dsadc_ChannelStatusType

Syntax	`Dsadc_ChannelStatusType`	
Type	uint8	
File	`Dsadc.h`	
Range	0 - DSADC_IDLE	DSADC channel is in idle state
	1- DSADC_BUSY	DSADC channel is in busy state
	2-DSADC_RESULT_READY	Result is available for DSADC channel
Description	Gives the status of DSADC channel	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.4 Dsadc_ChannelType

Table 115 Specification for Dsadc_ChannelType

Syntax	`Dsadc_ChannelType`	
Type	uint8	
File	`Dsadc.h`	
Range	0 to 255	
Description	Numeric identifier of DSADC channel.	

(table continues...)

Table 115 **(continued) Specification for Dsadc_ChannelType**

Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.5 Dsadc_ConfigType

Table 116 **Specification for Dsadc_ConfigType**

Syntax	`Dsadc_ConfigType`	
Type	Structure	
File	`Dsadc.h`	
Range	--	None
Description	This type defines the data structure used to store the configuration root for the DSADC driver	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.6 Dsadc_DelayType

Table 117 **Specification for Dsadc_DelayType**

Syntax	`Dsadc_DelayType`	
Type	uint32	
File	`Dsadc.h`	
Range	0 to 2147483647	
Description	This type defines the data structure used to store the STM delay used for the creating the delay. Note: The delay should be a non-zero value.	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.7 Dsadc_GainCorrType

Table 118 **Specification for Dsadc_GainCorrType**

Syntax	`Dsadc_GainCorrType`	
Type	uint32	
File	`Dsadc.h`	
Range	0 to 4294967295	
Description	This type defines the data structure used to store the gain correction value to be written to GAINCORR register.	

(table continues...)

Table 118 (continued) Specification for Dsadc_GainCorrType

Source	IFX
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.2.8 Dsadc_IrmsValueType

Table 119 Specification for Dsadc_IrmsValueType

Syntax	`Dsadc_IrmsValueType`	
Type	uint16	
File	`Dsadc.h`	
Range	0 to 65535	
Description	This data type is used to return the current IRMS value read from the UCB section.	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.9 Dsadc_NotifyFnPtrType

Table 120 Specification for Dsadc_NotifyFnPtrType

Syntax	`Dsadc_NotifyFnPtrType`
Type	Pointer to a function of type void Function_Name (void)
File	`Dsadc.h`
Description	Defines the function pointer type for callback functions.
Source	IFX
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.2.10 Dsadc_ResultType

Table 121 Specification for Dsadc_ResultType

Syntax	`Dsadc_ResultType`	
Type	sint16	
File	`Dsadc.h`	
Range	-32768 to 32767	
Description	Data type used for the result value generated from the DSADC channel.	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.11 Dsadc_SdcapValueType

Table 122 Specification for Dsadc_SdcapValueType

Syntax	`Dsadc_SdcapValueType`	
Type	uint8	
File	`Dsadc.h`	
Range	0 to 255	
Description	This type defines the data structure used to store the SDCAP value read from register.	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.12 Dsadc_SizeType

Table 123 Specification for Dsadc_SizeType

Syntax	`Dsadc_SizeType`	
Type	uint16	
File	`Dsadc.h`	
Range	0 to 65535	
Description	Datatype used to define the size of the buffer and also the same type is used to return the valid size entries	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.13 Dsadc_TimeStampType

Table 124 Specification for Dsadc_TimeStampType

Syntax	`Dsadc_TimeStampType`	
Type	uint16	
File	`Dsadc.h`	
Range	0 to 65535	
Description	This data type is used to return the current timestamp value.	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3 Functions - APIs

This section lists all the APIs of the DSADC driver.

1.3.3.1 Dsadc_Init

Table 125 Specification for `Dsadc_Init` API

Syntax	`void  Dsadc_Init`
	`(`
	`    const Dsadc_ConfigType * const ConfigPtr`
	`)`	

Service ID	0x1A	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	ConfigPtr	Pointer to the DSADC driver configuration structure
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	This API initializes the EDSADC hardware as per the configuration pointer passed and sets all the global variables to their required initial values. The SFRs of EDSADC hardware are reset to their default values and then they are configured with the given configuration data. This API will set the DSADC module state to DSADC_INITIALIZED if the initialization is successful. It also enables the DSADC module by writing into the CLC register and enable the modulator and demodulator for the configured channels.	
Source	IFX	
Error handling	DSADC_E_PARAM_CONFIG, DSADC_E_ALREADY_INITIALIZED, DSADC_E_CLC_FAILURE	
Configuration dependencies	-	
User hints	1. DSADC driver does not perform a NULL_PTR check on ConfigPtr, when DET is off. 2. Dsadc_StartCalibration API must be invoked by the user after the initialization is completed and before calling the Dsadc_StartModulation API . 3. Mcu_Init() should be called before calling this API. 4. Interrupts should be in a disabled state before calling this API.	

(table continues...)

Table 125 (continued) Specification for `Dsadc_Init` API

SFR accessed	CPU_COMPAT(w), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), EDSADC_CGCFG(w), EDSADC_CH_BOUNDSEL(w), EDSADC_CH_CGSYNC(w), EDSADC_CH_DICFG(w), EDSADC_CH_FCFGA(w), EDSADC_CH_FCFGC(w), EDSADC_CH_FCFGM(w), EDSADC_CH_GAINCAL(w), EDSADC_CH_GAINCORR(w), EDSADC_CH_GAINCTR(w), EDSADC_CH_IWCTR(w), EDSADC_CH_MODCFG(w), EDSADC_CH_OFFCOMP(w), EDSADC_CH_OVSCFG(w), EDSADC_CH_RECTCFG(w), EDSADC_CH_RFC(w), EDSADC_CH_TSCNT(w), EDSADC_CH_VCM(w), EDSADC_CLC(rw), EDSADC_EVFLAGCLR(w), EDSADC_GLOBCFG(w), EDSADC_GLOBRC(rw), SCU_CCUCON0(r), SCU_EICON0(rw), SCU_EICR(w), SCU_IGCR(w), SCU_OSCCON(r), SCU_SEICON0(rw), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.2 Dsadc_DeInit

Table 126 Specification for `Dsadc_DeInit` API

Syntax	`void Dsadc_DeInit ( void )`	
Service ID	0x1B	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	This API resets all SFRs of the EDSADC configured during initialization to their reset values. It sets DSADC module's state to DSADC_UNINIT_COMPLETED. It also disables the EDSADC hardware by writing into the CLC register. This API is available only when DsadcDeInitApi is configured as TRUE.	
Source	IFX	
Error handling	DSADC_E_UNINIT, DSADC_E_CLC_FAILURE	

(table continues...)

Table 126 (continued) Specification for `Dsadc_DeInit` API

Configuration dependencies	DsadcDeInitApi
User hints	None
SFR accessed	CPU_COMPAT(w), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), EDSADC_CGCFG(w), EDSADC_CH_BOUNDSEL(w), EDSADC_CH_CGSYNC(w), EDSADC_CH_DICFG(w), EDSADC_CH_FCFGA(w), EDSADC_CH_FCFGC(w), EDSADC_CH_FCFGM(w), EDSADC_CH_GAINCAL(w), EDSADC_CH_GAINCORR(w), EDSADC_CH_GAINCTR(w), EDSADC_CH_IWCTR(w), EDSADC_CH_MODCFG(w), EDSADC_CH_OFFCOMP(w), EDSADC_CH_OVSCFG(w), EDSADC_CH_RECTCFG(w), EDSADC_CH_RFC(w), EDSADC_CH_TSCNT(w), EDSADC_CH_VCM(w), EDSADC_CLC(rw), EDSADC_EVFLAGCLR(w), EDSADC_GLOBCFG(w), EDSADC_GLOBRC(w), SCU_CCUCON0(r), SCU_EICON0(rw), SCU_EICR(w), SCU_IGCR(w), SCU_OSCCON(r), SCU_SEICON0(rw), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.3 Dsadc_StartModulation

Table 127 Specification for `Dsadc_StartModulation` API

Syntax	`Std_ReturnType  Dsadc_StartModulation`
	`(`
	`    const Dsadc_ChannelType ChannelId`
	`)`	

Service ID	0x1C	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant for same channel. Reentrant for other channels	
Parameters (in)	ChannelId	Numeric Id of the requested DSADC Channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: DSADC channel result data acquisition is enabled E_NOT_OK: DSADC channel result data acquisition is disabled
Description	This API enables the result data acquisition for the given channel and enables the trigger source if it is configured. It sets the channel status to DSADC_BUSY.	
Source	IFX	

(table continues...)

Table 127 (continued) Specification for `Dsadc_StartModulation` API

Error handling	DSADC_E_PARAM_CHANNEL, DSADC_E_BUSY, DSADC_E_UNINIT, DSADC_E_CALIB_RUNNING
Configuration dependencies	-
User hints	In case of Trigger mode configured as Window, 1. The data acquisition not started immediately after calling this function. 2. Data acquisition starts only after the window open event. 3. During window close event the result data acquisition will be stopped. After calling the APIs Dsadc_Init or Dsadc_StartCalibration there must be a delay of 2 x Group delay shall be added before invoking the API Dsadc_StartModulation.
SFR accessed	CPU_COMPAT(w), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), EDSADC_CH_DICFG(r), EDSADC_CH_FCFGM(rw), EDSADC_CH_FCNTC(r), EDSADC_CH_RFC(rw), EDSADC_EVFLAGCLR(w), SCU_CCUCON0(r), SCU_IGCR(w), SCU_OSCCON(r), SCU_SEICON0(rw), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.4 Dsadc_StopModulation

Table 128 Specification for `Dsadc_StopModulation` API

Syntax	`Std_ReturnType  Dsadc_StopModulation`
	`(`
	`    const Dsadc_ChannelType ChannelId`
	`)`	

Service ID	0x1D	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant for same channel, Reentrant for other channels	
Parameters (in)	ChannelId	Numeric Id of the requested DSADC Channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: DSADC Channel Result data acquisition is stopped E_NOT_OK: DSADC Channel Result data acquisition is not stopped

(table continues...)

Table 128 (continued) Specification for `Dsadc_StopModulation` API

Description	This API Disables the Result data acquisition for the given channel and disable the Trigger source if it is configured. It sets the Channel status to DSADC_IDLE.
Source	IFX
Error handling	DSADC_E_UNINIT, DSADC_E_PARAM_CHANNEL
Configuration dependencies	-
User hints	After calling this API, Result data acquisition is stopped immediately even though the window is in active state in case of Trigger mode window.
SFR accessed	EDSADC_CH_FCFGM(rw), SCU_IGCR(w) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.5 Dsadc_ReadStreamResults

Table 129 Specification for `Dsadc_ReadStreamResults` API

Syntax	`Dsadc_SizeType  Dsadc_ReadStreamResults`
	`(`
	`    const Dsadc_ChannelType ChannelId,`
	`    Dsadc_ResultType * const ResultLinearBufferPtr`
	`)`	

Service ID	0x1E	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant for same channel, Reentrant for other channels	
Parameters (in)	ChannelId	Numeric Id of the requested DSADC Channel
Parameters (out)	-	-
Parameters (in - out)	ResultLinearBufferPtr	Location to store the requested channel result.
Return	Dsadc_SizeType	65535 -> Read stream result failed. 0-> No Failure in the ReadStream Result, but no data available for read Other than 0 and 65535 -> Read stream result is successful and the return value indicate the size of valid data in the result Buffer.

(table continues...)

Table 129 (continued) Specification for `Dsadc_ReadStreamResults` API

Description	This API reads the conversion results stored in the linear buffer. If the ResultLinearBufferPtr is not the channel Buffer what was configured in the Dsadc_SetupResultBuffer then this API copies the conversion results from channel buffer to the ResultLinearBufferPtr. This API returns the number of valid conversion results. The starting location of the conversion result is always 0 in the buffer. If the Parameter DsadcTimestampFeature is configured as DSADC_TIMESTAMP_ENABLED and the DsadcTriggerMode configured as TRIGGER_MODE_WINDOW then the first location of the buffer always contains the timestamp count for the window open event and the next value is the conversion result prior to window opening. *Note: This API is available only if the DsadcAccessMode is not configured as DSADC_DMA_ACCESS for all the configured channels.*
Source	IFX
Error handling	DSADC_E_UNINIT, DSADC_E_PARAM_CHANNEL, DSADC_E_PARAM_POINTER, DSADC_E_INV_LINEAR_BUFFER_CONFIG, DSADC_E_FIFO_FAILURE, DSADC_E_INVALID_BUFFER_POINTER
Configuration dependencies	DsadcAccessMode
User hints	When the channel is configured for linear buffer with window close timestamp and if the Dsadc_ReadStreamResults API is invoked before the Buffer full or window close event then the timestamp value will be invalid. Always call the API only after the window close or Buffer full notifications to get the correct timestamp value. Call this Function only when the Access Mode is configured as Linear buffer.
SFR accessed	-
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.6 Dsadc_ReadResult

Table 130 Specification for `Dsadc_ReadResult` API

Syntax	`Std_ReturnType  Dsadc_ReadResult`
	`(`
	`    const Dsadc_ChannelType ChannelId,`
	`    Dsadc_ResultType * const ResultPtr`
	`)`	

Service ID	0x1F	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant for same channel, Reentrant for other channels	
Parameters (in)	ChannelId	Numeric Id of the requested DSADC Channel

(table continues...)

Table 130 (continued) Specification for `Dsadc_ReadResult` API

Parameters (out)	-	-
Parameters (in - out)	ResultPtr	Access mode single: Pointer to result data from the DSADC hardware result register Access mode circular Buffer: Pointer to result data from the circular buffer current read pointer location. In both modes if any error present data pointed by this pointer will not be updated. If there is no error but there is no data to read then data pointed by pointer will be updated with value 0.
Return	Std_ReturnType	E_OK: Requested DSADC channel result is read E_NOT_OK: Failed to read requested DSADC channel result.
Description	This API reads the result data for given DSADC channel. If the DsadcAccessMode is configured as DSADC_SINGLE_READ then this API read the DSADC hardware result register to update location pointed by ResultPtr. If the DsadcAccessMode is configured as DSADC_CIRCULAR_BUFFER then this API update the location pointed by ResultPtr with the circular Buffer data. *Note: This API is available only if the DsadcAccessMode is not configured as DSADC_DMA_ACCESS for all the configured channels.*	
Source	IFX	
Error handling	DSADC_E_UNINIT, DSADC_E_PARAM_CHANNEL, DSADC_E_PARAM_POINTER, DSADC_E_INV_CIRCULAR_BUFFER_CONFIG, DSADC_E_FIFO_FAILURE, DSADC_E_INVALID_BUFFER_POINTER	
Configuration dependencies	DsadcAccessMode	
User hints	None	
SFR accessed	EDSADC_CH_DICFG(r), EDSADC_CH_RESM(r), EDSADC_CH_RFC(rw) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.7 Dsadc_GetStatus

Table 131 Specification for `Dsadc_GetStatus` API

Syntax	`Dsadc_ChannelStatusType Dsadc_GetStatus`
	`(`
	`    const Dsadc_ChannelType ChannelId`
	`)`

Service ID	0x20
Sync/Async	Synchronous

(table continues...)

Table 131 **(continued) Specification for `Dsadc_GetStatus` API**

Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant for same channel, Reentrant for other channels	
Parameters (in)	ChannelId	Numeric Id of the requested DSADC Channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Dsadc_ChannelStatusType	DSADC_IDLE: DSADC driver is in idle state. No action is performed. DSADC_BUSY: DSADC driver is processing the input signal. DSADC_RESULT_READY: DSADC driver is in ready state to read the converted results.
Description	This API returns the current status of the requested DSADC channel. In case of DET or SAFETY error the channel status is returned as DSADC_IDLE.	
Source	IFX	
Error handling	DSADC_E_UNINIT, DSADC_E_PARAM_CHANNEL	
Configuration dependencies	-	
User hints	when the DsadcResultHandlingImplementation is configured as DSADC_DMA_MODE for a channel, then the status DSADC_RESULT_READY will not be set by DSADC Driver. After the Dsadc_StartModulation API call and till the Dsadc_StopModulation API call, the channel status stays only at DSADC_BUSY	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.8 Dsadc_SetupResultBuffer

Table 132 **Specification for `Dsadc_SetupResultBuffer` API**

Syntax	`Std_ReturnType Dsadc_SetupResultBuffer`
	`(`
	`    const Dsadc_ChannelType ChannelId,`
	`    const Dsadc_ResultType * const DataBufferPtr,`
	`    const Dsadc_SizeType Size`
	`)`

Service ID	0x21
Sync/Async	Synchronous
Safety Level	Refer to the release notes for the safety related info
Re-entrancy	Non Reentrant for same channel, Reentrant for other channels

(table continues...)

Table 132 (continued) Specification for `Dsadc_SetupResultBuffer` API

Parameters (in)	ChannelId DataBufferPtr Size	Numeric Id of the requested DSADC Channel Pointer to the start of result buffer(Channel buffer) for the requested channel. Result buffer size which defines the number of result values that can be stored in the result buffer. Maximum size of the buffer should be 65534.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: Result buffer(Channel Buffer) initialization is successful E_NOT_OK: Result buffer Initialization failed
Description	This API sets up the start address of Channel specific result buffers, where the conversion results will be stored. This API sets up the buffer only if the DsadcAccessMode is configured as DSADC_STREAM_LINEAR_BUFFER or DSADC_CIRCULAR_BUFFER. This API returns E_OK on successful initialization of result buffer. *Note: This API is available only if the DsadcAccessMode is not configured as DSADC_DMA_ACCESS for all the configured channels.*	
Source	IFX	
Error handling	DSADC_E_PARAM_POINTER, DSADC_E_UNINIT, DSADC_E_PARAM_CHANNEL, DSADC_E_UNIDLE, DSADC_E_INVALID_BUFFER_CONFIG, DSADC_E_INV_BUFFER_SIZE	
Configuration dependencies	DsadcAccessMode	
User hints	None	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.9 Dsadc_StartCarrierSignal

Table 133 Specification for `Dsadc_StartCarrierSignal` API

Syntax	`Std_ReturnType  Dsadc_StartCarrierSignal`
	`(`
	`  void`
	`)`

Service ID	0x22
Sync/Async	Synchronous
Safety Level	Refer to the release notes for the safety related info
Re-entrancy	Non Reentrant

(table continues...)

Table 133 (continued) Specification for `Dsadc_StartCarrierSignal` API

Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: Carrier signal generation started successfully. E_NOT_OK: Failed to start the carrier signal generation.
Description	This API is used to start the generation of the carrier signal from the carrier generator based on the configured waveform properties for exciting the resolver coils. This interface returns E_OK on successful starting of the carrier signal.	
Source	IFX	
Error handling	DSADC_E_UNINIT, DSADC_E_CARRIER_ALREADY_RUNNING	
Configuration dependencies	-	
User hints	None	
SFR accessed	EDSADC_CGCFG(rw) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.10 Dsadc_StopCarrierSignal

Table 134 Specification for `Dsadc_StopCarrierSignal` API

Syntax	`Std_ReturnType Dsadc_StopCarrierSignal`
	`(`
	`  void`
	`)`	

Service ID	0x23	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-

(table continues...)

Table 134 (continued) Specification for `Dsadc_StopCarrierSignal` API

Return	Std_ReturnType	E_OK: Carrier signal generation stopped successfully. E_NOT_OK: Failed to stop the carrier signal generation.
Description	The interface is used to stop the generation of carrier signal from the carrier generator. This interface returns E_OK on successfully stopping of carrier signal	
Source	IFX	
Error handling	DSADC_E_UNINIT	
Configuration dependencies	-	
User hints	Stopping of the carrier generator terminates the PWM output, after the completion of the current period.	
SFR accessed	EDSADC_CGCFG(rw) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.11 Dsadc_EnableNotifications

Table 135 Specification for `Dsadc_EnableNotifications` API

Syntax	`void Dsadc_EnableNotifications`
	`(`
	`    const Dsadc_ChannelType ChannelId`
	`)`	

Service ID	0x24	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant for same channel, Reentrant for other channels	
Parameters (in)	ChannelId	Numeric Id of the requested DSADC Channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	This API enables the notification mechanism for the requested DSADC Channel.	
Source	IFX	
Error handling	DSADC_E_UNINIT, DSADC_E_PARAM_CHANNEL, DSADC_E_NOTIF_CAPABILITY	

(table continues...)

Table 135 (continued) Specification for `Dsadc_EnableNotifications` API

Configuration dependencies	-
User hints	Any one of the Channel notification function should not be a NULL Pointer
SFR accessed	-
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.12 Dsadc_DisableNotifications

Table 136 Specification for `Dsadc_DisableNotifications` API

Syntax	`void  Dsadc_DisableNotifications`
	`(`
	`    const Dsadc_ChannelType ChannelId`
	`)`	

Service ID	0x25	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant for same channel, Reentrant for other channels	
Parameters (in)	ChannelId	Numeric Id of the requested DSADC Channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	This API disables the notification mechanism for the requested DSADC Channel.	
Source	IFX	
Error handling	DSADC_E_UNINIT, DSADC_E_PARAM_CHANNEL, DSADC_E_NOTIF_CAPABILITY	
Configuration dependencies	-	
User hints	Any one of the Channel notification function should not be a NULL Pointer	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.13 Dsadc_GetTimestamp

Table 137 Specification for `Dsadc_GetTimestamp` API

Syntax	`Dsadc_TimeStampType  Dsadc_GetTimestamp`
	`(`
	`    const Dsadc_ChannelType ChannelId`
	`)`	

Service ID	0x26	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant for same channel, Reentrant for other channels	
Parameters (in)	ChannelId	Numeric Id of the requested DSADC Channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Dsadc_TimeStampType	Timestamp count for the last read result event. In case of error value 0 will be returned.
Description	This API returns the timestamp count value for the Dsadc_ReadResult API read result event. This timestamp count is the time from the HW result event till the Dsadc_ReadResult API reads the result value from the HW Result register.	
Source	IFX	
Error handling	DSADC_E_UNINIT, DSADC_E_PARAM_CHANNEL, DSADC_E_SINGLE_ACCESSMODE_TIMESTAMP	
Configuration dependencies	-	
User hints	None	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.14 Dsadc_StartCalibration

Table 138 Specification for `Dsadc_StartCalibration` API

Syntax	`Std_ReturnType  Dsadc_StartCalibration`
	`(`
	`    const Dsadc_ChannelType ChannelId`
	`)`

Service ID	0x27
Sync/Async	Synchronous

(table continues...)

Table 138 (continued) Specification for `Dsadc_StartCalibration` API

Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant for same channel, Reentrant for other channels	
Parameters (in)	ChannelId	Numeric Id of the requested DSADC Channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: Calibration algorithm started successfully E_NOT_OK: Failed to start the calibration algorithm
Description	This API triggers the calibration algorithm. The calibration algorithm will be triggered only when the current status of the channel is DSADC_IDLE. This interface returns E_OK on successful start of the calibration algorithm	
Source	IFX	
Error handling	DSADC_E_UNINIT, DSADC_E_PARAM_CHANNEL, DSADC_E_UNIDLE, DSADC_E_CALIB_RUNNING	
Configuration dependencies	-	
User hints	The Calibration related parameters must be valid for the successful calibration.	
SFR accessed	EDSADC_CH_FCFGM(rw) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.15 Dsadc_GetCalibrationStatus

Table 139 Specification for `Dsadc_GetCalibrationStatus` API

Syntax	`Dsadc_CalibrationStatusType  Dsadc_GetCalibrationStatus`
	`(`
	`    const Dsadc_ChannelType ChannelId`
	`)`	

Service ID	0x28	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant for same channel, Reentrant for other channels	
Parameters (in)	ChannelId	Numeric Id of the requested DSADC Channel

(table continues...)

Table 139 (continued) Specification for `Dsadc_GetCalibrationStatus` API

Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Dsadc_CalibrationStatusType	DSADC_CALIBRATION_NOT_STARTED: DSADC driver calibration is not yet started for the given channel. DSADC_CALIBRATION_DONE: DSADC driver calibration is done for the given channel. DSADC_CALIBRATION_RUNNING: DSADC driver calibration is currently running for the given channel. DSADC_CALIBRATION_ERROR: DSADC driver calibration is failed for the given channel.
Description	This API returns the current calibration status for the given channel. In case of DET or SAFETY error the status DSADC_CALIBRATION_NOT_STARTED will be returned. When the Calibration status is DSADC_CALIBRATION_DONE then the status will be changed to DSADC_CALIBRATION_NOT_STARTED. So that when this function is called next time DSADC_CALIBRATION_NOT_STARTED is returned.	
Source	IFX	
Error handling	DSADC_E_PARAM_CHANNEL, DSADC_E_UNINIT	
Configuration dependencies	-	
User hints	Once the calibration is done and if it is success then this API will return DSADC_CALIBRATION_DONE. If this API is called again before calling Dsadc_StartCalibration API then it will return the calibration status as DSADC_CALIBRATION_NOT_STARTED.	
SFR accessed	EDSADC_CH_FCNTC(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.16 Dsadc_InitCheck

Table 140 Specification for `Dsadc_InitCheck` API

Syntax	`Std_ReturnType Dsadc_InitCheck`
	`(`
	`    const Dsadc_ConfigType * const ConfigPtr`
	`)`

Service ID	0x29
Sync/Async	Synchronous
Safety Level	Refer to the release notes for the safety related info

(table continues...)

Table 140 **(continued) Specification for `Dsadc_InitCheck` API**

Re-entrancy	Non Reentrant	
Parameters (in)	ConfigPtr	Pointer to the DSADC Driver configuration structure
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK : Initialization check passed E_NOT_OK : In Case of - Driver is not initialized - ConfigPtr input is NULL - Global Variables or SFR is not set as expected
Description	This API verifies the DSADC Module Initialization. This API returns E_NOT_OK when configuration pointer is NULL or DSADC driver is not initialized. *Note: This API is available only when the parameter DsadcInitCheckApi is configured as TRUE.*	
Source	IFX	
Error handling	-	
Configuration dependencies	DsadcInitCheckApi	
User hints	The DSADC module environment should ensure the following calling sequence for this API. 1. Dsadc_Init API is called. 2. Dsadc_Initcheck API is called.	
SFR accessed	EDSADC_CGCFG(r), EDSADC_CH_BOUNDSEL(r), EDSADC_CH_CGSYNC(r), EDSADC_CH_DICFG(r), EDSADC_CH_FCFGA(r), EDSADC_CH_FCFGC(r), EDSADC_CH_FCFGM(r), EDSADC_CH_GAINCAL(r), EDSADC_CH_GAINCORR(r), EDSADC_CH_GAINCTR(r), EDSADC_CH_IWCTR(r), EDSADC_CH_MODCFG(r), EDSADC_CH_OFFCOMP(r), EDSADC_CH_OVSCFG(r), EDSADC_CH_RECTCFG(r), EDSADC_CH_RFC(r), EDSADC_CH_TSCNT(r), EDSADC_CH_VCM(r), EDSADC_CLC(r), EDSADC_GLOBCFG(r), EDSADC_GLOBRC(r), SCU_EICR(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.17 Dsadc_GetVersionInfo

Table 141 **Specification for `Dsadc_GetVersionInfo` API**

Syntax	`void  Dsadc_GetVersionInfo`
	`(`
	`    Std_VersionInfoType * const versioninfo`
	`)`	

Service ID	0x2C	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	versioninfo	Pointer to where to store the version information of the DSADC driver.
Parameters (in - out)	-	-
Return	void	-
Description	API returns the version information of this driver. *Note: This API is available only when DsadcVersionInfoApi is configured as TRUE.*	
Source	IFX	
Error handling	DSADC_E_PARAM_POINTER	
Configuration dependencies	DsadcVersionInfoApi	
User hints	None	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.18 Dsadc_RestartDemodulator

Table 142 **Specification for `Dsadc_RestartDemodulator` API**

Syntax	`Std_ReturnType  Dsadc_RestartDemodulator`
	`(`
	`    const Dsadc_ChannelMaskType ChannelId`
	`)`

Service ID	0x2D
Sync/Async	Synchronous
Safety Level	Refer to the release notes for the safety related info
Re-entrancy	Non Reentrant for same channel. Reentrant for other channels

(table continues...)

Table 142 (continued) Specification for `Dsadc_RestartDemodulator` API

Parameters (in)	ChannelId	Bit coded information for the DSADC logical channel numbers. DSADC logical channel is indicated by the corresponding bit position. For example if bit 0 is set means logical channel ID 0 is requested for the demodulator restart.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: Demodulator is restarted for all the DSADC channels requested. E_NOT_OK: Restart of the demodulator is failed.
Description	This API restart the demodulators for all the requested DSADC channels. The DSADC channels are requested as a bit coded information in the input parameter (i.e. Channel 0 is requested by setting the bit position 0 of the input parameter).	
Source	IFX	
Error handling	DSADC_E_PARAM_CHANNEL, DSADC_E_UNINIT	
Configuration dependencies	-	
User hints	The result buffer (circular buffer or linear buffer) would hold the results acquired by DSADC. When the demodulator is restarted using Dsadc_RestartDemodulator API, the new results would be appended in the results buffer along with the older results. If the user wishes to flush the older results stored in the result buffer, the user can do the following : - If linear buffer is used, make a call to the Dsadc_ReadStreamResults API which would read and empty the results buffer. - If circular buffer is used, make calls to Dsadc_ReadResult API till the results buffer is emptied.	
SFR accessed	EDSADC_GLOBRC(rw) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.19 Dsadc_GetIrmsValue

Table 143 Specification for `Dsadc_GetIrmsValue` API

Syntax	`Dsadc_IrmsValueType  Dsadc_GetIrmsValue`
	`(`
	`    const Dsadc_ChannelType ChannelId`
	`)`

Service ID	0x2E
Sync/Async	Synchronous

(table continues...)

Table 143 (continued) Specification for `Dsadc_GetIrmsValue` API

Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant.	
Parameters (in)	ChannelId	Numeric ID of requested DSADC channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Dsadc_IrmsValueType	The IRMS value read from UCB section for DSADC channel.
Description	This API reads the IRMS value from the UCB section for the requested DSADC channel. The resolution of irms value is 0.01 micro ampere.	
Source	IFX	
Error handling	DSADC_E_PARAM_CHANNEL, DSADC_E_UNINIT	
Configuration dependencies	-	
User hints	-	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.20 Dsadc_SetGainCorrRegValue

Table 144 Specification for `Dsadc_SetGainCorrRegValue` API

Syntax	`Std_ReturnType  Dsadc_SetGainCorrRegValue`
	`(`
	`    const Dsadc_ChannelType ChannelId,`
	`    const Dsadc_GainCorrType RegValue`
	`)`	

Service ID	0xF0	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non-Reentrant for same channel, Reentrant for other channels.	
Parameters (in)	ChannelId RegValue	Numeric ID of requested DSADC channel. Value to be written into Gain correction register(GAINCORR)
Parameters (out)	-	-
Parameters (in - out)	-	-

(table continues...)

Table 144 (continued) Specification for `Dsadc_SetGainCorrRegValue` API

Return	Std_ReturnType	E_OK : DSADC channel Gain correction is updated based on user request E_NOT_OK: DSADC channel Gain correction is not updated
Description	This API writes the gain correction to the GAINCORR register. The CICSHIFT field has a valid range from value 0x00 to 0x1C, the values 0x1D to 0x1F are reserved and shall not be used. User shall configure the values within the range. The reserve bits in the register shall be configured with value 0 only.	
Source	IFX	
Error handling	DSADC_E_UNINIT, DSADC_E_PARAM_CHANNEL, DSADC_E_INVALID_PARAM_VALUE, DSADC_E_BUSY	
Configuration dependencies	-	
User hints	-	
SFR accessed	EDSADC_CH_GAINCORR(w) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.21 Dsadc_RestartIntegrator

Table 145 Specification for `Dsadc_RestartIntegrator` API

Syntax	`Std_ReturnType  Dsadc_RestartIntegrator`
	`(`
	`    const Dsadc_ChannelMaskType ChannelId,`
	`    const Dsadc_DelayType Delay`
	`)`	

Service ID	0xF1	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	ChannelId	Bit coded information for the DSADC logical channel numbers. DSADC logical channel is indicated by the corresponding bit position. For example if bit 0 is set means logical channel ID 0 is requested for the integrator restart. Similarly if bit position 2 is set then logical channel ID 2 is requested for the integrator restart. Customer shall pass value 0x5U to restart channel 0 and channel 2.
	Delay	The delay to be inserted between the restart of integrator for selected channels. The delay parameter is in STM ticks.

(table continues...)

Table 145 (continued) Specification for `Dsadc_RestartIntegrator` API

Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK: Integrator is restarted for all the DSADC channels requested. E_NOT_OK: Restart of the Integrator is failed.
Description	This API restarts the integrators for all the DSADC channels requested. The DSADC channels are requested as a bit coded information in the input parameter (If the input parameter value is 0x5U (bit position 0 and 2 are set), this indicates DSADC logical Channels 0 and 2 are requested for integrator restart). The delay shall be passed as ticks count for STM. Example for STM ticks calculation: Considering the STM frequency of 100 MHz and a delay of 12.800 microseconds. STM frequency = 100 MHz TstmPeriod = 1/100 MHz = 10 ns = 0.01 microseconds. Ticks = Delay/TstmPeriod = 1280 ticks. **Note:** 1. Carrier signal is restarted along with integrator in this API. 2. For channels which have bypassed integrators the restart integrator API does not have effect. User shall not pass the channels configured with integrator bypassed as input parameter for the API.	
Source	IFX	
Error handling	DSADC_E_PARAM_CHANNEL, DSADC_E_UNINIT, DSADC_E_INVALID_PARAM_DELAY	
Configuration dependencies	DsadcRestartIntegratorApi	
User hints	Carrier signal is also restarted along with the integrator.	
SFR accessed	EDSADC_CGCFG(w), EDSADC_CH_DICFG(w) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.22 Dsadc_GetSdcapValue

Table 146 Specification for `Dsadc_GetSdcapValue` API

Syntax	`Dsadc_SdcapValueType  Dsadc_GetSdcapValue`
	`(`
	`    const Dsadc_ChannelType ChannelId`
	`)`

(table continues...)

Table 146 **(continued) Specification for `Dsadc_GetSdcapValue` API**

Service ID	0xF2	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant	
Parameters (in)	ChannelId	Numeric Id of the requested DSADC Channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Dsadc_SdcapValueType	The SDCAP value read from register.
Description	This API reads the SDCAP value for the requested DSADC channel.	
Source	IFX	
Error handling	DSADC_E_PARAM_CHANNEL, DSADC_E_UNINIT	
Configuration dependencies	-	
User hints	-	
SFR accessed	EDSADC_CH_CGSYNC(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.4 Notifications and Callbacks

This section lists all the notifications and callbacks of the DSADC driver.

1.3.4.1 Dsadc_TimerIsr

Table 147 **Specification for `Dsadc_TimerIsr` API**

Syntax	`void  Dsadc_TimerIsr`
	`(`
	`    const Dsadc_ChannelMaskType ChannelId,`
	`    const uint32 StatusFlags`
	`)`

Service ID	0x2B
Sync/Async	Synchronous
Safety Level	Refer to the release notes for the safety related info
Re-entrancy	Reentrant for different channel

(table continues...)

Table 147 (continued) Specification for `Dsadc_TimerIsr` API

Parameters (in)	ChannelId StatusFlags	Bit coded information for the DSADC logical channel numbers. For example if bit 0 is enabled means the channel 0 is requested. Source of the ISR: For GTM: CCU0 (1) or CCU1 (2) For ERU: Pattern match (1) or Pattern miss (0)
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	Handles the interrupt from GTM and ERU for window events(Open and Close). If the DSADC channel is using GTM as a Trigger source then all the DSADC channel requested using the parameter ChannelId as a bit coded information will be serviced one by one. *Note: This Callback function is available only if the DsadcTriggerSelect is configured as GTM or ERU resource for at least any one of the DSADC channel.*	
Source	IFX	
Error handling	DSADC_SE_INVALID_ISR, DSADC_SE_EARLY_WINDOW_ISR	
Configuration dependencies	DsadcTriggerSelect	
User hints	Setting of SRC Register for the corresponding ERU and GTM resource must be handled by the OS/Application. This functon shall be called by application/user during the CCU0 and CCU1 interrupt event if the GTM timer for the DSADC trigger is handled by application. User shall not call this function if the IFX MCAL PWM driver is used to handle the GTM timer for the DSADC trigger.	
SFR accessed	EDSADC_CH_DICFG(r), EDSADC_CH_RESM(r), EDSADC_CH_RFC(rw), EDSADC_CH_TSTMP(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.5 Scheduled functions

The DSADC driver does not provide any scheduled functions.

1.3.6 Interrupt service routines

This section lists all the interrupt handlers of the DSADC driver.

1.3.6.1 Dsadc_Isr

Table 148 Specification for `Dsadc_Isr` API

Syntax	`void Dsadc_Isr`
	`(`
	`const Dsadc_ChannelType HwChannelId`
	`)`	

Service ID	0x2A	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	HwChannelId	Hardware channel number
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	Handles the interrupts from Main Service Request for the given DSADC HW Channel Id. *Note: This ISR is available only if the DsadcAccessMode is not configured as DSADC_DMA_ACCESS for all the configured channels.*	
Source	IFX	
Error handling	DSADC_SE_INVALID_ISR, DSADC_SE_PARAM_HW_CHANNEL	
Configuration dependencies	DsadcAccessMode	
User hints	User must call this interrupt handler from the ISR of DSADCxSRGM of each channel and pass the HW Channel Id as the parameter.	
SFR accessed	EDSADC_CH_DICFG(r), EDSADC_CH_RESM(r), EDSADC_CH_RFC(rw), EDSADC_EVFLAG(r), EDSADC_EVFLAGCLR(w) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.7 Callout

The DSADC driver does not provide any callout.

1.3.8 Errors Handling

This section describes the various error types reported by the DSADC driver.

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
DSADC_E_CLC_FAILURE: Error is reported when enabling/ disabling of CLC (module clock) fails.	IFX	Assigned by DEM	Production Error	Assigned by DEM	Production Error
DSADC_E_FIFO_FAILURE: Error is reported when hardware FIFO failure is detected.	IFX	Assigned by DEM	Production Error	Assigned by DEM	Production Error
DSADC_E_INVALID_BUFFER_POINTER: Error code is reported if the result buffer range is within the channel buffer range.	IFX	0x10	DET_SAFETY	0x10	DET_SAFETY
DSADC_E_INVALID_PARAM_DELAY: Error code is reported if the API service is invoked with wrong delay value.	IFX	0x14	DET_SAFETY	0x14	DET_SAFETY
DSADC_E_INVALID_PARAM_VALUE: Error code is reported if the Dsadc_SetGainCorrRegValue API service is invoked with wrong CICSHIFT value as input parameter.	IFX	0x15	DET_SAFETY	0x15	DET_SAFETY
DSADC_E_INV_BUFFER_SIZE: Error code is reported when the requested size of the buffer is zero or greater than the maximum buffer size.	IFX	0x0F	DET_SAFETY	0x0F	DET_SAFETY
DSADC_E_SINGLE_ACCESSMODE_TIMESTAMP: Error code is reported When Dsadc_GetTimestamp API is invoked and if the DsadcAccessMode is not configured as DSADC_SINGLE_READ or if the DsadcTimestampFeature is DSADC_TIMESTAMP_DISABLED or if the DsadcTriggerMode is DSADC_TRIGGER_MODE_WINDOW.	IFX	0x0D	DET_SAFETY	0x0D	DET_SAFETY
DSADC_SE_EARLY_WINDOW_ISR: Safety Error shall be reported when Timer Isr is Called before the Dsadc_StartModulation invoked.	IFX	0x13	SAFETY	0x13	SAFETY

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
DSADC_SE_PARAM_HW_CHANNEL: Error code is reported if the HW Channel ID passed is not configured.	IFX	0x12	SAFETY	0x12	SAFETY
DSADC_E_PARAM_CONFIG: Error code is reported if Dsadc_Init has been called with incorrect configuration parameter (configuration pointer is NULL_PTR).	IFX	0x01	DET_SAFETY	0x01	DET_SAFETY
DSADC_SE_INVALID_ISR: Error is reported if unintended interrupt is triggering the ISR.	IFX	0x11	SAFETY	0x11	SAFETY
DSADC_E_ALREADY_INITIALIZED: Error code is reported if the Dsadc_Init API is called while the DSADC driver is already in initialized state.	IFX	0x02	DET_SAFETY	0x02	DET_SAFETY
DSADC_E_UNINIT: Error code is reported if the API service is invoked before the module initialization.	IFX	0x03	DET_SAFETY	0x03	DET_SAFETY
DSADC_E_BUSY: Error code is reported when the Dsadc_StartModulation API is called while the result acquisition is already started for the given channel or Dsadc_SetGainCorrRegValue API is called while the channel is busy with conversion.	IFX	0x04	DET_SAFETY	0x04	DET_SAFETY
DSADC_E_PARAM_CHANNEL: Error code is reported if the passed input channel ID is not configured.	IFX	0x05	DET_SAFETY	0x05	DET_SAFETY
DSADC_E_PARAM_POINTER: Error code is reported if the API is invoked with null-pointer as a parameter.	IFX	0x06	DET_SAFETY	0x06	DET_SAFETY
DSADC_E_INV_LINEAR_BUFFER_CONFIG: Error code is reported when the access mode is not configured as Linear buffer and the API Dsadc_ReadStreamResults is called.	IFX	0x07	DET_SAFETY	0x07	DET_SAFETY

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
DSADC_E_INV_CIRCULAR_BUFFER_CONFIG: Error code is reported when the DsadcAccessMode is configured as DSADC_STREAM_LINEAR_BUFFER or DSADC_DMA_ACCESS	IFX	0x08	DET_SAFETY	0x08	DET_SAFETY
DSADC_E_INVALID_BUFFER_CONFIG: Error code is reported when the configured DsadcAccessMode is neither DSADC_STREAM_LINEAR_BUFFER nor DSADC_CIRCULAR_BUFFER.	IFX	0x09	DET_SAFETY	0x09	DET_SAFETY
DSADC_E_UNIDLE: Error code is reported when the current channel status is not DSADC_IDLE.	IFX	0x0A	DET_SAFETY	0x0A	DET_SAFETY
DSADC_E_CARRIER_ALREADY_RUNNING: Error code is reported when the Dsadc_StartCarrierSignal API is called but the carrier signal is already running.	IFX	0x0B	DET_SAFETY	0x0B	DET_SAFETY
DSADC_E_NOTIF_CAPABILITY: Error code is reported when the enable/disable notification function for a channel is called but there is no notification function configured for that channel.	IFX	0x0C	DET_SAFETY	0x0C	DET_SAFETY
DSADC_E_CALIB_RUNNING: Error code is reported when the calibration algorithm is still running for the requested channel.	IFX	0xE	DET_SAFETY	0xE	DET_SAFETY

1.3.9 Deviations and limitations

This section describes the deviations and limitations of the DSADC driver.

1.3.9.1 Deviations

This section describes the deviation of the DSADC driver

1.3.9.1.1 Software specification deviations

This section describes the deviations from software specification.

Table 149 Known deviations

Reference	Deviation
Address and Data CRC in the DMA mode	Since the Data CRC and Address CRC features of DMA are not used for DSADC driver, the user shall ensure that, while using the DMA mode a plausibility check of the conversion result is performed either by redundancy or by other means.
Spurious interrupt reporting when the channel is in inactive state	The DSADC driver cannot report the occurrence of a spurious interrupt when the channel is in the inactive state [that is, before invoking the Dsadc_StartModulation() API and after invoking the Dsadc_StopModulation() API].

1.3.9.1.2 AMDC Violations

The DSADC driver does not have any AMDC violations.

1.3.9.1.3 VSMD Violations

The DSADC driver does not have any VSMD violations.

1.3.9.2 Limitations

This section describes the limitations of the DSADC driver.

Table 150 Known limitations

Reference	Limitation
Channel has trigger mode window and timestamp enabled	The last two conversion results before the window close events are lost due to the hardware limitation.
Input parameter of Dsadc_InitCheck is used only to check CLC register.	Input parameter of Dsadc_Initcheck() API is used only to check CLC register. To check all other registers, parameter used during Dsadc_Init() API is used for Initcheck evaluation.

Revision history

Table 151 Revision History

Date	Version	Description
2023-06-07	8.0	Document is released.
2023-05-23	7.1	In Integration hints, the following points are modified: -DEM Module section has been removed. -Mcal_Wrapper Module section has been added. The Assumption of Use has been added for 'ConfigPtr passed to InitCheck' and 'InitCheck Sequence' in section 1.2 DEM has been modified to Production Error wherever applicable. Updated the C File Structure to remove Dem.h and added Mcal_Wrapper.h. Removed DEM Module and added Mcal_Wrapper module in Hw-Sw Interface Diagram in Figure 1. Dsadc_InitCheck() API description is updated for return type E_NOT_OK case information. ASIL level has been updated to Safety Level and the description is updated for Safety Level.
2021-11-08	7.0	Document is released.
2021-11-03	6.1	‘Mapping of hardware-software interfaces’ figure is corrected.
2021-10-27	6.0	Document is released.
2021-10-04	5.1	Added details for the new APIs - Dsadc_GetIrmsValue, Dsadc_SetGainCorrRegValue, Dsadc_GetSdcapValue, Dsadc_RestartIntegrator. Config variant attribute table information is removed and added this information in 'Configuration interfaces' section.
2021-03-25	5.0	Document is released.
2021-03-24	4.1	Added note in the description for the following parameters : - DsadcCalibGainCorrMulFactor - DsadcGainCorrMulFactor
2021-03-09	4.0	Document is released.
2021-03-03	3.1	File name updated for Dsadc_PBcfg.c.
2020-11-30	3.0	Document is released.
2020-11-27	2.1	Added an example use case for Key architectural considerations - Result acquisition using gate signal.
2020-08-18	2.0	Document is released.
2020-08-18	1.1	User hints updated for Dsadc_RestartDemodulator API.
2020-08-13	1.0	Document is released.

(table continues...)

Table 151 (continued) Revision History

2020-08-06	0.1	- DSADC driver chapter moved from MC-ISAR_TC3xx_UM_CD to this document - Unsupported hardware features section removed. - Hardware software interface section updated. - Deviations moved to the Software Specification deviations section.

Trademarks

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

Edition 2023-07-07

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
IFX-ocr1484806431059

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.

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.