# Module – ADC2

# High-Level Description

The ADC2 module shall control sampling and conversion of voltages from the hardware layer into digital signals.

# Figures

## Component Diagram

# Variable Data Dictionary

For details on module input / output variable, refer to the Data Dictionary for the application. Input / output variable names are listed here for reference.

|  |  |
| --- | --- |
| Module Inputs | Module Outputs |
| None | ADC2OffsetComp\_Cnt\_u8p8 |

## Module Internal Variables

This section identifies the name, range and resolutions for module specific data created by this module. If there are no range restrictions on the variable, the term “FULL” is placed into the table for legal range.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Variable Name | Resolution | Legal Range  (min) | Legal Range  (max) | Software Segment |
| None |  |  |  |  |

### User defined typedef definition/declaration

This section documents any user types uniquely used for the module.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Typedef Name | Element Name | User Defined Type | Legal Range  (min) | Legal Range  (max) |
| See Adc\_MDD.doc for Adc subsystem types |  |  |  |  |

# Constant Data Dictionary

## Calibration Constants

This section lists the calibrations used by the module. For details on calibration constants, refer to the Data Dictionary for the application.

|  |
| --- |
| Constant Name |
| None |

## Program(fixed) Constants

### Embedded Constants

All embedded constants whose values are provided in Eng units will be evaluated to the equivalent counts by using the FPM\_InitFixedPoint\_m() macro within the #define statement.

#### Local

| Constant Name | Resolution | Units | Value |
| --- | --- | --- | --- |
| D\_GROUPEV\_CNT\_U8 | 1 | Counts | 0 |
| D\_GROUP1\_CNT\_U8 | 1 | Counts | 1 |
| D\_GROUP2\_CNT\_U8 | 1 | Counts | 2 |
| D\_NTCPARMBIT1\_CNT\_U8 | 1 | Counts | 2 |
| D\_ADC2EVTBUFSZ\_CNT\_U08 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G1BUFSZ\_CNT\_U08 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G2BUFSZ\_CNT\_U08 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2EVSRC\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G1SRC\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G2SRC\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2NUMEVTCH\_CNT\_U08 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2EVTCH\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2NUMG1CH\_CNT\_U08 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G1CH\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2NUMG2CH\_CNT\_U08 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G2CH\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2RSLTBASEADR\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2EVINTENA\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2EVSAMPDISEN\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2EVFIFORESETCR\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2EVDMACR\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G1INTENA\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G1SAMPDISEN\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G1FIFORESETCR\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G1DMACR\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G2INTENA\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G2SAMPDISEN\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G2FIFORESETCR\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2G2DMACR\_CNT\_U32 | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_HWTRGADC2GEVT\_CNT\_LGC | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_HWTRGADC2G1\_CNT\_LGC | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_HWTRGADC2G2\_CNT\_LGC | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ADC2USEDMA\_CNT\_LGC | 1 | Counts | Generated in Adc2\_Cfg.h |
| D\_ISRGROUP\_CNT\_U8 | 1 | Counts | Generated in Adc2\_Cfg.h |

#### Global

This section lists the global constants used by the module. For details on global constants, refer to the Data Dictionary for the application.

|  |
| --- |
| Constant Name |
| None |

### Module specific Lookup Tables Constants

|  |  |  |  |
| --- | --- | --- | --- |
| Constant Name | Resolution | Value | Software Segment |
| T\_Adc2GroupConfigData\_Cnt\_Str[3] | Adc\_GroupConfigDataType | {  D\_ADC2NUMEVTCH\_CNT\_U08,  D\_ADC2EVTCH\_CNT\_U32,  &(((uint32\*)D\_ADC2RSLTBASEADR\_CNT\_U32)[0]),  },  {  D\_ADC2NUMG1CH\_CNT\_U08,  D\_ADC2G1CH\_CNT\_U32,  &(((uint32\*)D\_ADC2RSLTBASEADR\_CNT\_U32)[D\_ADC2EVTBUFSZ\_CNT\_U08]),  },  {  D\_ADC2NUMG2CH\_CNT\_U08,  D\_ADC2G2CH\_CNT\_U32,  &(((uint32\*)D\_ADC2RSLTBASEADR\_CNT\_U32)[D\_ADC2EVTBUFSZ\_CNT\_U08 + D\_ADC2G1BUFSZ\_CNT\_U08]),  } | CONST\_32 |

# Functions/Macros used by the Sub-Modules

## Library Functions / Macros

The library and functions / Macros that are called by the various sub modules are identified below,

1. None

## Data Hiding Functions

None

## Global Functions/Macros Defined by this Module

### ADC2 Init

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | Adc2\_init1 | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | ConfigPtr | Adc\_ConfigType\* | N/A | N/A |  |
| **Return Value** | N/A |  |  |  |  |

This function is used to initialize each of the ADC2 Groups (Event, Group1, and Group2).

#### Initialize Module Internal Variables

None

#### Register Configuration





Adc2\_Write\_ADC2OffsetComp\_Cnt\_u8p8\_Cnt\_T\_u8p8(Adcoffset\_Cnt\_T\_u8p8)

#### Start Initial Conversions



### Start Group Conversion

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | Adc2\_StartGroupConversion | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | group | uint8 | 0 | 2 |  |
| **Return Value** | N/A |  |  |  |  |

This function starts a software triggered conversion of all channels of the requested Adc channel group.

#### Start Conversions



### Enable Group Notification

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | Adc2\_EnableGroupNotification | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | group | uint8 | 0 | 15 |  |
| **Return Value** | N/A |  |  |  |  |

This inline function reads and returns the individual channel conversion result.

#### Enable Notification Conditions



## Local Functions/Macros Used by this MDD only

### Read Channel Conversions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | Adc2\_ReadConversion | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | ConvId | uint16 | 0 | 15 |  |
| **Return Value** | conversion result | uint16 | 0 | 4095 | 0 |

This inline function reads and returns the individual channel conversion result.

#### Read Group



# Software Module Implementation

## Runtime Environment (RTE) Initial Values

None

## Initialization Functions

None

## Periodic Functions

None

## Fault Recovery Functions

None

## Shutdown Functions

None

## Interrupt Functions

None

## Serial Communication Functions

None

## Transition Functions

None

# Execution Requirements

## Execution Sequence of the Module

## Execution Rates for sub-modules called by the Scheduler

This table serves as reference for the Scheduler design

|  |  |  |
| --- | --- | --- |
| Function Name | Calling Frequency | System State(s) in which the function is called |
| Adc2\_Init | Once (at initialization) | STARTUP |

## Execution Requirements for Serial Communication Functions

|  |  |
| --- | --- |
| Function Name | Sub-Module called by (Serial Comm Function Name) |
| N/A |  |

# Memory Map Definition Requirements

## Sub Modules (Functions)

This table identifies the software segments for functions identified in this module.

|  |  |
| --- | --- |
| Name of Sub Module | Software Segment |
| Adc2\_Init | ADC2\_START\_SEC\_CODE |
| Adc2\_StartGroupConversion | ADC2\_START\_SEC\_CODE |
| Adc2\_EnableGroupNotification | ADC2\_START\_SEC\_CODE |
| Adc2\_ReadConversion | inline |

## Local Functions

This table identifies the software segments for local functions identified in this module.

|  |  |
| --- | --- |
| Name of Sub Module | Software Segment |
| None |  |

# Known Limitations With Design

1. INLINE functions defined in “GlobalMacro.h” are not unit tested
2. Buffer pointer initialization could hang if adc conversion did not complete. Consider adding a timeout and possible setting of NTC 0x33 bit 1.
3. ADC component outputs the ADC calibration compensation which is in turn used by Current Measurement to correct its offset volt . This is implementation done across two components. The measured currents from ADC and conversion from counts to volts is not done in this component as described by the FDD but current measurement component.

# Revision Control Log

|  |  |  |  |
| --- | --- | --- | --- |
| **Rev #** | **Change Description** | **Date** | **Author Initials** |
| 1.0 | Initial version based on FDD 33C rev 006 | 05Nov12 | LN |
| 2.0 | Design updates to meet FDD and simplify configuration | 24APR13 | Selva |
| 3.0 | Fixed typo in flowchart | 22MAY13 | LWW |
| 4.0 | Updated for use with DMA. Moved init function scheduling requirements to integration manual. Added design limitation note about buffer pointer initialization. | 10APR14 | KMC |
| 5.0 | Updated for ADC offset compensation | 27JUN14 | Selva |