# Module -- ADC

# High-Level Description

The ADC module shall control sampling and conversion of voltages from the hardware layer into digital signals. The ADC Register definition has been captured from the TMS570LS31x/21x 16/32-Bit RISC Flash Microcontroller Technical Reference Manual (SPNU499 – September 2011).

# 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 | ADC1OffsetComp\_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) |
| Adc\_GroupType | N/A | uint8 | 0 | 2 |
| Adc\_GroupConfigDataType | uint8 NumOfChannels,  uint32 ChannelSelect,  uint32\* ResultBuf | struct | FULL | FULL |
| Adc\_StatusType | ADC\_IDLE,  ADC\_BUSY,  ADC\_COMPLETED,  ADC\_STREAM\_COMPLETED | enum | 0 | 3 |
| Adc\_ValueGroupType | N/A | uint16 | 0 | 4095 |
| Adc\_ValueGroupType | N/A | uint16\* | FULL | FULL |

# 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 |
| k\_VbattOVTransIntConfig\_Cnt\_u32 |

## 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\_NUMOFGROUPS\_CNT\_U8 | 1 | Counts | 3 |
| D\_GROUPEV\_CNT\_U8 | 1 | Counts | 0 |
| D\_GROUP1\_CNT\_U8 | 1 | Counts | 1 |
| D\_GROUP2\_CNT\_U8 | 1 | Counts | 2 |
| D\_NUMOFADCCALREADS\_CNT\_U8 | 1 | Counts | 8 |
| D\_NTCPARMBIT0\_CNT\_U8 | 1 | Counts | 1 |
| D\_NTCPARMBIT1\_CNT\_U8 | 1 | Counts | 2 |
| D\_NTCPARMBIT2\_CNT\_U8 | 1 | Counts | 4 |
| D\_ADC1RSLTBASEADR\_CNT\_U32 | 1 | Counts | 0xFF3E0000U |
| D\_ADC1NUMRSLTBUF\_CNT\_U08 | 1 | Counts | 64 |
| D\_ADC1CURRENTMODE\_ULS\_LGC | 1 | BOOLEAN | Generated in Adc\_Cfg.h. |
| D\_ADC1MAGINTMASK1\_CNT\_U16 | 1 | Counts | 0/0\* |
| D\_ADC1MAGINTASET\_CNT\_U16 | 1 | Counts | 1/0\* |
| D\_ADC1MAGINTCR1\_CNT\_U32 | 1 | Counts | k\_VbattOVTransIntConfig\_Cnt\_u32/ 0\* |
| D\_ADC1THRINTENACLR\_CNT\_U16 | 1 | Counts | 6/7\* |
| D\_ADC1EVTBUFSZ\_CNT\_U08 | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_ADC1G1BUFSZ\_CNT\_U08 | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_ADC1G2BUFSZ\_CNT\_U08 | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_GROUPEND\_CNT\_U32 | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_GROUPBUSY\_CNT\_U32 | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_ADC1GEVTSRC\_CNT\_U32 | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_ADC1GEVTDMACR\_CNT\_U32 | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_ADC1G1SRC\_CNT\_U32 | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_ADC1G1DMACR\_CNT\_U32 | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_ADC1G2SRC\_CNT\_U32 | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_ADC1G2DMACR\_CNT\_U32 | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_ADC1USEDMA\_CNT\_LGC | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_HWTRGADC1GEVT\_CNT\_LGC | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_HWTRGADC1G1\_CNT\_LGC | 1 | Counts | Generated in Adc\_Cfg.h. |
| D\_HWTRGADC1G2\_CNT\_LGC | 1 | Counts | Generated in Adc\_Cfg.h. |

#### D\_ADC1CURRENTMODE\_ULS\_LGC if True value 1 is selected to proceed. If not value 2 will be used,

#### 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\_Adc1GroupConfigData\_Cnt\_Str[3] | Adc\_GroupConfigDataType | {  D\_ADC1NUMEVTCH\_CNT\_U08,  D\_ADC1EVTCH\_CNT\_U32,  &(((uint32\*)D\_ADC1RSLTBASEADR\_CNT\_U32)[0]),  },  {  D\_ADC1NUMG1CH\_CNT\_U08,  D\_ADC1G1CH\_CNT\_U32,  &(((uint32\*)D\_ADC1RSLTBASEADR\_CNT\_U32)[D\_ADC1EVTBUFSZ\_CNT\_U08]),  },  {  D\_ADC1NUMG2CH\_CNT\_U08,  D\_ADC1G2CH\_CNT\_U32,  &(((uint32\*)D\_ADC1RSLTBASEADR\_CNT\_U32)[D\_ADC1EVTBUFSZ\_CNT\_U08 + D\_ADC1G1BUFSZ\_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

1. <None>

## Global Functions/Macros Defined by this Module

### ADC Init

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | Adc\_Init\_FixedCfg  ( macro aliased is Adc\_Init(x) ) | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | None | N/A | N/A | N/A |  |
| **Return Value** | N/A |  |  |  |  |

This function is used to initialize each of the ADC1 Groups (Event, Group1, and Group2).

#### Initialize Module Internal Variables

None

#### Register Configuration





Adc1\_Write\_ADC1OffsetComp\_Cnt\_u8p8\_Cnt\_T\_u8p8(Adcoffset\_Cnt\_T\_u8p8)

### Setup Result Buffer

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | Adc\_SetupResultBuffer | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | Group | Adc\_GroupType | 0 | 2 |  |
|  | DataBufferPtr | Adc\_ValueGroupRefType | FULL | FULL |  |
| **Return Value** | N/A | Std\_ReturnType | 0 | 0 | 0 |

This function is the Adc group result buffer setup service. It is responsible for setting up the result buffer pointer of the group.

This Api is provided to support for the standard Autosar API, however the design of this driver does not buffer an intermediate copy of the results and therefore does not require a result buffer. The hardware provides a sufficient Adc results buffer.

For optimization purposes, this function could be defined as a macro.

#### Setup Group’s Result Buffer

return E\_OK

### Start Group Conversion

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | Adc\_StartGroupConversion | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | Group | Adc\_GroupType | 0 | 2 |  |
|  | DataBufferPtr | Adc\_ValueGroupRefType | FULL | FULL |  |
| **Return Value** | N/A | Std\_ReturnType | 0 | 0 | 0 |

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

In order to provide optimized execution of this single line function, this API should be defined as a macro, however, due to timing it is at the moment defined as a function. The need for minimal runtime is driven by a Nexteer use case which executes this in the context of the MtrCtrl ISR in some EPS systems.

#### Select Channels to Convert

adcREG1->GxSEL[Group] = T\_Adc1GroupConfigData\_Cnt\_Str[Group].ChannelSelect

### Get Group Status

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | Adc\_GetGroupStatus | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | Group | Adc\_GroupType | 0 | 2 |  |
| **Return Value** | ReturnVal\_Cnt\_T\_Enum | Adc\_StatusType | 0 | 3 | 0 |

Returns the conversion status of the requested ADC1 channel group.

#### Get Status



### Read Group Conversions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | Adc\_ReadGroup | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | Group | Adc\_GroupType | 0 | 2 |  |
| **Return Value** | ReturnVal\_Cnt\_T\_Enum | Adc\_StatusType | 0 | 3 | 0 |

Reads the group conversion result of the last completed conversion round of the requested group and

stores the channel values starting at the DataBufferPtr address. The group channel values are stored in ascending channel number order (in contrast to the storage layout of the result buffer if streaming access is configured).

Per the design direction in FDD33, direct ADC result buffer acces is used instead of using the hardware FIFO access register.

#### Read Group



## Local Functions/Macros Used by this MDD only

None

# 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 |
| Adc\_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 |
| Adc\_Init | ADC\_START\_SEC\_CODE |
| Adc\_SetupResultBuffer | ADC\_START\_SEC\_CODE |
| Adc\_StartGroupConversion | ADC\_START\_SEC\_CODE |
| Adc\_ReadGroup | ADC\_START\_SEC\_CODE |
| Adc\_GetGroupStatus | ADC\_START\_SEC\_CODE |
| ADCOffsetCalibration | ADC\_START\_SEC\_CODE |

## Local Functions

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

|  |  |
| --- | --- |
| Name of Sub Module | Software Segment |
|  |  |

# 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 0x32 bit 1.

# Revision Control Log

|  |  |  |  |
| --- | --- | --- | --- |
| **Rev #** | **Change Description** | **Date** | **Author Initials** |
| 1.0 | Initial version based on FDD 33C rev 006 | 05Nov12 | LN |
| 2 | Constant confiuguration tables renamed to start with “T\_”  Rewored Adc\_ReadGroup to provide direct fixed memory access as required in the FDD | 15Jan13 | JJW |
| 3 | Updated FDD 33C rev 007 and FDD33E rev2 | 25-April-13 | Selva |
| 4 | Updated for use with DMA. Moved init function scheduling requirements to integration manual. Updated/corrected flowcharts and module specific lookup table contants. Added design limitation note about buffer pointer initialization. | 10-Apr-14 | KMC |
| 5 | Updated for ADC offset compensation | 27-Jun-14 | Selva |