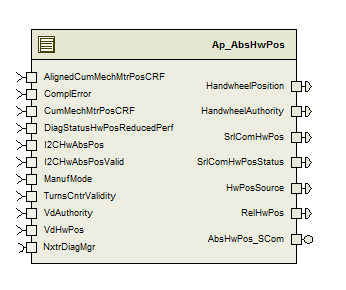
# Module -- Absolute Handwheel Position TcI2cVd

# High-Level Description

The Absolute Hand Wheel Position Function is responsible for determining the steering wheel hand wheel position using either a Turns Counter estimate of motor position during key off and learnt internal hw position (for eg. Sensorless Vehicle Dynamics) or I2C digital hw position sensor and learnt internal hw position (for eg. Sensorless Vehicle Dynamics).

# Figures



# 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.

(Note: Full variable names required in table.)

(Note: All global variables including End Of Line data used should be shown here)

|  |  |  |
| --- | --- | --- |
| Module Inputs | Module Outputs | |
| CumMechMtrPosCRF\_Deg\_f32 | | HandwheelPosition\_HwDeg\_f32 |
| AlignedCumMechMtrPosCRF\_Deg\_f32 | | HandwheelAuthority\_Uls\_f32 |
| TurnsCntrValidity\_Cnt\_u08 | | RelHwPos\_HwDeg\_f32 |
| I2CHwAbsPos\_HwDeg\_f32 | | HwPosSource\_Cnt\_u16 |
| I2CHwAbsPosValid\_Cnt\_lgc | | SrlComHwPos\_HwDeg\_f32 |
| VdHwPos\_HwDeg\_f32 | | SrlComHwPosStatus\_Cnt\_u16 |
| VdAuthority\_Uls\_f32 | |  |
| ComplError\_HwDeg\_f32 | |  |
| DiagStatusHwPosReducedPerf\_Cnt\_lgc | |  |
| ManufMode\_Cnt\_enum | |  |

## 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 |
| AbsHwPos\_HwPosState\_Cnt\_M\_enum | 1 | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_UNSPECIFIED |
| AbsHwPos\_HandwheelPositionLPF\_M\_str.SV\_Uls\_f32 | Single precision Float | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_UNSPECIFIED |
| AbsHwPos\_HandwheelPositionLPF\_M\_str.K\_Uls\_f32 | Single precision Float | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_UNSPECIFIED |
| AbsHwPos\_SrlComHwPosStatus\_Cnt\_M\_u16 | 1 | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_16 |
| AbsHwPos\_VehCntrValid\_Cnt\_M\_lgc | n/a | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_BOOLEAN |
| AbsHwPos\_VehCntrOfstLearn\_Cnt\_M\_lgc | n/a | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_BOOLEAN |
| AbsHwPos\_HwAtoMtrAFltAcc\_Cnt\_M\_u16 | 1 | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_16 |
| AbsHwPos\_HwPosSource\_Cnt\_M\_u16 | 1 | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_16 |
| AbsHwPos\_VehCntrOffset\_HwDeg\_M\_f32 | Single Precision Float | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_32 |
| AbsHwPos\_AlignedHwPos\_HwDeg\_M\_f32 | Single precision Float | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_32 |
| AbsHwPos\_AlignedCumMtrPos\_HwDeg\_M\_f32 | Single precision Float | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_32 |
| AbsHwPos\_PrevHandwheelPosition\_HwDeg\_M\_f32 | Single precision Float | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_32 |
| AbsHwPos\_HandwheelAuthority\_Uls\_M\_f32 | Single precision Float | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_32 |
| AbsHwPos\_TempHwPos\_HwDeg\_M\_f32 | Single precision Float | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_32 |
| AbsHwPos\_SrlComHwPos\_HwDeg\_M\_f32 | Single precision Float | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_32 |
| AbsHwPos\_TargetHwAuthority\_Uls\_M\_f32 | Single precision Float | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_32 |
| AbsHwPos\_RelHwPos\_HwDeg\_M\_f32 | Single precision Float | See Data Dictionary | See Data Dictionary | ABSHWPOS\_START\_SEC\_VAR\_CLEARED\_32 |

### 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) |
| HwPosStateType | HWPOS\_STATE\_INIT  HWPOS\_STATE\_SENSORINVALID  HWPOS\_STATE\_SENSORVALID  HWPOS\_STATE\_INPUTINVALID | uint8  uint8  uint8  uint8 | 8  4  2  14 | 8  4  2  14 |

# 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\_GearRatio\_Uls\_f32 |
| k\_UseTurnsCntr\_Cnt\_lgc |
| k\_HwPosAuthorityStep\_Uls\_f32 |
| k\_HwPosOutputLPFCoeffFc\_Hz\_f32 |
| k\_HwPosOutputLPFError\_HwDeg\_f32 |
| k\_TurnsCntrAuthority\_Uls\_f32 |
| k\_I2CHwAuthority\_Uls\_f32 |
| k\_VdAuthority\_Uls\_f32 |
| k\_MaxVehCntrOffDiff\_HwDeg\_f32 |
| k\_KinmIntDiagMaxRackLimit\_HwDeg\_f32 |
| k\_HWAtoMtrADiffLimit\_HwDeg\_f32 |
| k\_HwAtoMtrAError\_str |
| k\_VehCntrOffValidLimit\_HwDeg\_f32 |
| k\_LoSpdVDAuthority\_Uls\_f32 |
| k\_HiSpdVDAuthority\_Uls\_f32 |

## 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\_HWPOSMAX\_HWDEG\_F32 | Single Precision Float | HwDeg | 1600 |
| D\_RELHWPOSMAX\_HWDEG\_F32 | Single Precision Float | HwDeg | 3200 |
| D\_INVALIDOFFSET\_HWDEG\_F32 | Single Precision Float | HwDeg | 65535 |
| D\_NOAUTHORITY\_ULS\_F32 | Single Precision Float | Uls | 0 |
| D\_MAXAUTHORITY\_ULS\_F32 | Single Precision Float | Uls | 1 |
| D\_SRCUNKNOWN\_CNT\_U16 | 1 | Cnt | 65534 |
| D\_SRCTURNSCNTR\_CNT\_U16 | 1 | Cnt | 1 |
| D\_SRCI2CSENSOR\_CNT\_U16 | 1 | Cnt | 2 |
| D\_SRCLOSPDVD\_CNT\_U16 | 1 | Cnt | 3 |
| D\_SRCHISPDVD\_CNT\_U16 | 1 | Cnt | 4 |
| D\_HWPOSSTATUSFAULT\_CNT\_U16 | 1 | Cnt | 0xFFFF |
| D\_HWPOSSTATUSUNKNOWN\_CNT\_U16 | 1 | Cnt | 0xFFFE |
| D\_HWPOSSTATUSVALID\_CNT\_U16 | 1 | Cnt | 0x5555 |
| D\_TCVCOMPUTING\_CNT\_U08 | 1 | Cnt | 0 |
| D\_TCVVALID\_CNT\_U08 | 1 | Cnt | 100 |
| D\_TCVINVALID\_CNT\_U08 | 1 | Cnt | 255 |
| D\_TRIMPERFORMED\_CNT\_U16 | 1 | Cnt | 0xAAAA |
| D\_TRIMNOTPERFORMED\_CNT\_U16 | 1 | Cnt | 0 |

#### 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 |
| D\_2MS\_SEC\_F32 |

### Module specific Lookup Tables Constants

This is for lookup tables (arrays) with fixed values, same name as other tables.

|  |  |  |  |
| --- | --- | --- | --- |
| Constant Name | Resolution | Value | Software Segment |
| None |  |  |  |

# 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. LPF\_KUpdate\_f32\_m
2. LPF\_OpUpdate\_f32\_m
3. Abs\_f32\_m
4. Limit\_m
5. DiagPStep\_m
6. DiagNStep\_m
7. DiagFailed\_m

## Data Hiding Functions

1. Rte\_Pim\_EOLVehCntrOffset()->EOLVehCntrOffset\_HwDeg\_f32
2. Rte\_Pim\_EOLVehCntrOffset()->EOLHwPosTrimPerformed\_Cnt\_u16

## Global Functions/Macros Defined by this Module

None

## Local Functions/Macros Used by this MDD only

### Trim Not Performed Diagnostic

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | TrimNotPerfDiag | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | ManufMode\_Cnt\_T\_enum | ManufModeType | 0 | 2 | n/a |
| **Return Value** | n/a |  |  |  |  |

#### Description



# Software Module Implementation

## Runtime Environment (RTE) Initial Values

This section lists the initial values of data written by this module but controlled by the RTE. After RTE initialization, the data in this table will contain these values.

|  |  |
| --- | --- |
| Data | Value |
| Rte\_InitValue\_AlignedCumMechMtrPosCRF\_Deg\_f32 | 0 |
| Rte\_InitValue\_ComplError\_HwDeg\_f32 | 0 |
| Rte\_InitValue\_CumMechMtrPosCRF\_Deg\_f32 | 0 |
| Rte\_InitValue\_DiagStatusHwPosReducedPerf\_Cnt\_lgc | FALSE |
| Rte\_InitValue\_HandwheelAuthority\_Uls\_f32 | 0 |
| Rte\_InitValue\_HandwheelPosition\_HwDeg\_f32 | 0 |
| Rte\_InitValue\_HwPosSource\_Cnt\_u16 | 0 |
| Rte\_InitValue\_I2CHwAbsPos\_HwDeg\_f32 | 0 |
| Rte\_InitValue\_I2CHwAbsPosValid\_Cnt\_lgc | FALSE |
| Rte\_InitValue\_ManufMode\_Cnt\_enum | 0 |
| Rte\_InitValue\_RelHwPos\_HwDeg\_f32 | 0 |
| Rte\_InitValue\_SrlComHwPos\_HwDeg\_f32 | 0 |
| Rte\_InitValue\_SrlComHwPosStatus\_Cnt\_u16 | 0 |
| Rte\_InitValue\_TurnsCntrValidity\_Cnt\_u08 | 0 |
| Rte\_InitValue\_VdAuthority\_Uls\_f32 | 0 |
| Rte\_InitValue\_VdHwPos\_HwDeg\_f32 | 0 |

## Initialization Functions

### Init: AbsHwPos\_Init

#### Design Rationale

None

#### Store Module Inputs to Local copies

ManufMode\_Cnt\_T\_enum = Rte\_IRead\_AbsHwPos\_Init1\_ManufMode\_Cnt\_enum()

#### Description



#### Store Local copy of outputs into Module Outputs

Rte\_IWrite\_AbsHwPos\_Init1\_SrlComHwPosStatus\_Cnt\_u16 (AbsHwPos\_SrlComHwPosStatus\_Cnt\_M\_u16)

Rte\_IWrite\_AbsHwPos\_Init1\_HwPosSource\_Cnt\_u16 (AbsHwPos\_HwPosSource\_Cnt\_M\_u16)

## Periodic Functions

### Per: Per1

#### Design Rationale

AbsHwPos\_Per1 function does the RelHwPos output, and the remaining functionality that runs at 2ms is done in AbsHwPos\_Per2. The 2ms is split into two periodics so that the Vehicle Dynamics 2ms periodic can be called in between them, since the two components each use output(s) from the other.

#### Program Flow Start

Rte\_Call\_AbsHwPos\_Per1\_CP0\_CheckpointReached()

#### Store Module Inputs to Local copies

AlignedCumMechMtrPosCRF\_Deg\_T\_f32 = Rte\_IRead\_AbsHwPos\_Per1\_AlignedCumMechMtrPosCRF\_Deg\_f32()

ComplError\_HwDeg\_T\_f32 = Rte\_IRead\_AbsHwPos\_Per1\_ComplError\_HwDeg\_f32()

CumMechMtrPosCRF\_Deg\_T\_f32 = Rte\_IRead\_AbsHwPos\_Per1\_CumMechMtrPosCRF\_Deg\_f32()

#### Scale and Compensate Mtr Pos Signals used in Calculation



#### Store Local copy of outputs into Module Outputs

Rte\_IWrite\_AbsHwPos\_Per1\_RelHwPos\_HwDeg\_f32(RelHwPosLimited\_HwDeg\_T\_f32)

#### Program Flow End

Rte\_Call\_AbsHwPos\_Per1\_CP1\_CheckpointReached()

### Per: Per2

#### Design Rationale

Initial implementation used SetRamBlockStatus in place of WriteBlock to minimize writes to EEPROM, however, Aparna made it clear that all writes to EOLVehCntrOffset\_HwDeg\_f32 must happen immediately and not at power down thus necessitating the WriteBlock call.

#### Program Flow Start

Rte\_Call\_AbsHwPos\_Per2\_CP0\_CheckpointReached()

#### Store Module Inputs to Local copies

DiagStatusHwPosReducedPerf\_Cnt\_T\_lgc = Rte\_IRead\_AbsHwPos\_Per2\_DiagStatusHwPosReducedPerf\_Cnt\_lgc()

I2CHwAbsPosValid\_Cnt\_T\_lgc = Rte\_IRead\_AbsHwPos\_Per2\_I2CHwAbsPosValid\_Cnt\_lgc()

I2CHwAbsPos\_HwDeg\_T\_f32 = Rte\_IRead\_AbsHwPos\_Per2\_I2CHwAbsPos\_HwDeg\_f32()

TurnsCntrValidity\_Cnt\_T\_u08 = Rte\_IRead\_AbsHwPos\_Per2\_TurnsCntrValidity\_Cnt\_u08()

VdAuthority\_Uls\_T\_f32 = Rte\_IRead\_AbsHwPos\_Per2\_VdAuthority\_Uls\_f32()

VdHwPos\_HwDeg\_T\_f32 = Rte\_IRead\_AbsHwPos\_Per2\_VdHwPos\_HwDeg\_f32()

ManufMode\_Cnt\_T\_enum = Rte\_IRead\_AbsHwPos\_Per2\_ManufMode\_Cnt\_enum()

#### Determine Which State Sub-function to Use



#### Input Invalid State



#### Init State



#### Sensor Valid State



#### Sensor Invalid State



#### Calculate and Update VehCntr\_Offset in EEPROM



#### Output Smoothing Low Pass Filter



#### Store Local copy of outputs into Module Outputs

Rte\_IWrite\_AbsHwPos\_Per2\_HandwheelAuthority\_Uls\_f32(HandwheelAuthorityLimited\_Uls\_T\_f32)

Rte\_IWrite\_AbsHwPos\_Per2\_HandwheelPosition\_HwDeg\_f32(HandwheelPosition\_HwDeg\_T\_f32)

Rte\_IWrite\_AbsHwPos\_Per2\_HwPosSource\_Cnt\_u16(AbsHwPos\_HwPosSource\_Cnt\_M\_u16)

Rte\_IWrite\_AbsHwPos\_Per2\_SrlComHwPosStatus\_Cnt\_u16(AbsHwPos\_SrlComHwPosStatus\_Cnt\_M\_u16)

Rte\_IWrite\_AbsHwPos\_Per2\_SrlComHwPos\_HwDeg\_f32(SrlComHwPosLimited\_HwDeg\_T\_f32)

#### Program Flow End

Rte\_Call\_AbsHwPos\_Per2\_CP1\_CheckpointReached()

### Per: Per3

#### Design Rationale

None

#### Program Flow Start

Rte\_Call\_AbsHwPos\_Per3\_CP0\_CheckpointReached()

#### Store Module Inputs to Local copies

I2CHwAbsPosValid\_Cnt\_T\_lgc = Rte\_IRead\_AbsHwPos\_Per3\_I2CHwAbsPosValid\_Cnt\_lgc()

I2CHwAbsPos\_HwDeg\_T\_f32 = Rte\_IRead\_AbsHwPos\_Per3\_I2CHwAbsPos\_HwDeg\_f32()

#### HWA to Motor Angle Correlation Diagnostic



#### Store Local copy of outputs into Module Outputs

None

#### Program Flow End

Rte\_Call\_AbsHwPos\_Per3\_CP1\_CheckpointReached()

### Per: Per4

#### Design Rationale

None

#### Program Flow Start

Rte\_Call\_AbsHwPos\_Per4\_CP0\_CheckpointReached()

#### Store Module Inputs to Local copies

None

#### Kinematic Integrity Diagnostic



#### Store Local copy of outputs into Module Outputs

None

#### Program Flow End

Rte\_Call\_AbsHwPos\_Per4\_CP1\_CheckpointReached()

## Fault Recovery Functions

None

## Shutdown Functions

None

## Interrupt Functions

None

## Serial Communication Functions

### SComm: AbsHwPos\_SCom\_CustClrTrim

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | AbsHwPos\_SCom\_CustClrTrim | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | n/a | n/a | n/a | n/a | n/a |
| **Return Value** | n/a |  |  |  |  |

#### Design Rationale

For clarity and to protect against possible changes to MECCounter definition, using ManufMode\_Cnt\_Enum which is an enumeration value derived from the MECCounter, instead of using MECCounter directly as in the FDD.

#### Program Flow Start

None

#### Store Module Inputs to Local copies

Rte\_Read\_ManufMode\_Cnt\_enum(&ManufMode\_Cnt\_T\_enum)

#### Customer Clear Trim Service



#### Store Local copy of outputs into Module Outputs

None

#### Program Flow End

None

### SComm: AbsHwPos\_SCom\_CustSetTrim

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | AbsHwPos\_SCom\_CustSetTrim | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | n/a | n/a | n/a | n/a | n/a |
| **Return Value** |  | Std\_ReturnType | 0 | 34 | 0 |

#### Design Rationale

For clarity and to protect against possible changes to MECCounter definition, using ManufMode\_Cnt\_Enum which is an enumeration value derived from the MECCounter, instead of using MECCounter directly as in the FDD.

#### Program Flow Start

None

#### Store Module Inputs to Local copies

RespCode\_Cnt\_T\_u08 = 0

Rte\_Read\_ManufMode\_Cnt\_enum(&ManufMode\_Cnt\_T\_enum)

Rte\_Read\_TurnsCntrValidity\_Cnt\_u08(&TurnsCntrValidity\_Cnt\_T\_u08)

#### Customer Set Trim Service



#### Store Local copy of outputs into Module Outputs

None

#### Program Flow End

None

### SComm: AbsHwPos\_SCom\_NxtClearTrim

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | AbsHwPos\_SCom\_NxtClearTrim | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | n/a | n/a | n/a | n/a | n/a |
| **Return Value** | n/a |  |  |  |  |

#### Design Rationale

None

#### Program Flow Start

None

#### Store Module Inputs to Local copies

None

#### Nexteer Clear Trim Service



#### Store Local copy of outputs into Module Outputs

None

#### Program Flow End

None

### SComm: AbsHwPos\_SCom\_NxtSetTrim ()

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | AbsHwPos\_SCom\_NxtSetTrim | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | Offset\_HwDeg\_T\_f32 | Float | FULL | FULL | n/a |
| **Return Value** |  | Std\_ReturnType | 0 | 49 | 0 |

#### Design Rationale

None

#### Program Flow Start

None

#### Store Module Inputs to Local copies

RespCode\_Cnt\_T\_u08 = 0

Rte\_Read\_TurnsCntrValidity\_Cnt\_u08(&TurnsCntrValidity\_Cnt\_T\_u08)

#### Nexteer Set Trim Service



#### Store Local copy of outputs into Module Outputs

None

#### Program Flow End

None

# Execution Requirements

## Execution Sequence of the Module

(Describe in words relevant details about the execution sequence of the different sub modules.)

## 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 |
| AbsHwPos\_Init1 | On Init | Cold Init |
| AbsHwPos\_Per1 | 2 ms | All |
| AbsHwPos\_Per2 | 2 ms | All |
| AbsHwPos\_Per3 | 4 ms | All |
| AbsHwPos\_Per4 | 10 ms | All |

## Execution Requirements for Serial Communication Functions

|  |  |
| --- | --- |
| Function Name | Sub-Module called by (Serial Comm Function Name) |
| AbsHwPos\_SCom\_CustClrTrim | EPS\_DiagSrvcs\_ISO |
| AbsHwPos\_SCom\_CustSetTrim | EPS\_DiagSrvcs\_ISO |
| AbsHwPos\_SCom\_NxtClearTrim | EPS\_DiagSrvcs\_ISO |
| AbsHwPos\_SCom\_NxtSetTrim | EPS\_DiagSrvcs\_ISO |

# 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 |
| AbsHwPos\_Init1 | RTE\_START\_SEC\_AP\_ABSHWPOS\_APPL\_CODE |
| AbsHwPos\_Per1 | RTE\_START\_SEC\_AP\_ABSHWPOS\_APPL\_CODE |
| AbsHwPos\_Per2 | RTE\_START\_SEC\_AP\_ABSHWPOS\_APPL\_CODE |
| AbsHwPos\_Per3 | RTE\_START\_SEC\_AP\_ABSHWPOS\_APPL\_CODE |
| AbsHwPos\_Per4 | RTE\_START\_SEC\_AP\_ABSHWPOS\_APPL\_CODE |
| AbsHwPos\_SCom\_CustClrTrim | RTE\_START\_SEC\_AP\_ABSHWPOS\_APPL\_CODE |
| AbsHwPos\_SCom\_CustSetTrim | RTE\_START\_SEC\_AP\_ABSHWPOS\_APPL\_CODE |
| AbsHwPos\_SCom\_NxtClearTrim | RTE\_START\_SEC\_AP\_ABSHWPOS\_APPL\_CODE |
| AbsHwPos\_SCom\_NxtSetTrim | RTE\_START\_SEC\_AP\_ABSHWPOS\_APPL\_CODE |

## Local Functions

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

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

# Known Issues / Limitations With Design

1. INLINE functions in GlobalMacro.h are not unit tested
2. Because NVM updates are made from a periodic function, it is possible that the previous write will still be pending when a new write is desired. When this happens, the new NVM write is not performed.

# Revision Control Log

|  |  |  |  |
| --- | --- | --- | --- |
| **Rev #** | **Change Description** | **Date** | **Author Initials** |
| 1 | Initial component creation w/ FDD v002 | 21-Jun-13 | Jared |
| 2 | Updated port interface types to match StdDef | 7-Aug-13 | Jared |
| 3 | Unit test range corrections | 5-Sep-13 | Jared |
| 4 | Updated embedded constant values and EOLHwPosTrimPerformed logic to fix anomaly 5641 and 5643. Replaced module level variable ranges with references to Data Dictionary. Added limiting on global outputs. Updated to show that the previous AbsHwPos\_Per1 function has now been split into two functions, AbsHwPos\_Per1 and AbsHwPos\_Per2, which also caused the renaming of AbsHwPos\_Per2 and AbsHwPos\_Per3 to AbsHwPos\_Per3 and AbsHwPos\_Per4, respectively. | 22-Sep-13 | KMC |
| 5 | Updated component diagram to match current component; changed name TurnsCounterValidity to TurnsCntrValidity to match FDD; added Design Rationale regarding use of ManufMode instead of MECCounter; changed names of module level variables to meet naming conventions; updated flowcharts to show checks for NVM writes pending; added note in design limitations about periodic NVM writes; minor flowchart and text corrections; updated “sensor invalid” state logic for fix of anomaly 5864; added periodic and cold init processing of “trim not performed” diagnostic for fix of anomaly 5863. | 25-Nov-13 | KMC |
| 6 | A#6439 fix – Change in Section 6.3.2.8 flow chart and 6.3.2.9 flow chart where <FLT\_EPSILON hase been replaced by >= FLT\_EPSILON in both charts. | 04/17/2014 | LK |
| 7 | Anomaly Fix – Anom# 6633 – Modified the Flow Chart for “Trim Not Performed Diagnostic” | 05/01/2014 | SB |
| 8 | Update NTC 0x75 algorithm per ES-056 v6 | 07/01/2014 | BWL |