# Module --

# High-Level Description

This module calculates the cumulative and corrected motor positions. It also outputs the sine and cosine position signals used for diagnostics and motor electrical and mechanical position.

# Figures

#### Component Diagram

mtrpos.emf

# Variable Data Dictionary

|  |  |  |
| --- | --- | --- |
| Module Inputs | Module Outputs | |
|  | | MtrPos\_MechMtrPos\_Rev\_G\_u0p16 |
|  | | MtrPos\_CorrectedMtrPos\_Rev\_G\_u0p16 |
|  | | MtrPos\_SinTheta1\_Volts\_G\_s2p13 |
|  | | MtrPos\_CosTheta1\_Volts\_G\_s2p13 |
|  | | MtrPos\_SampleTime\_uS\_G\_u32 |

## Module Internal Variables

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Variable Name | Resolution | Legal Range  (min) | Legal Range  (max) | Software Segment |
| SinTheta2\_Uls\_D\_s2p13 | 2-13 | -2 | 2 | MTRPOS\_START\_SEC\_VAR\_CLEARED\_16 |
| MtrPos\_CorrectedMtrPos\_Rev\_M\_u0p16 [D\_MTRPOSDBLBUFFSZ\_CNT\_U08] | 2-16 | 0 | 0.99998 | MTRPOS\_START\_SEC\_VAR\_CLEARED\_16 |
| MtrPos\_Cos1Scaled\_Volts\_M\_u3p13 [D\_MTRPOSDBLBUFFSZ\_CNT\_U08] | 2-13 | 0 | 5 | MTRPOS\_START\_SEC\_VAR\_CLEARED\_16 |
| MtrPos\_Sin1Scaled\_Volts\_M\_u3p13 [D\_MTRPOSDBLBUFFSZ\_CNT\_U08] | 2-13 | 0 | 5 | MTRPOS\_START\_SEC\_VAR\_CLEARED\_16 |
| MtrPos\_MechMtrPos\_Rev\_M\_u0p16 [D\_MTRPOSDBLBUFFSZ\_CNT\_U08] | 2-16 | 0 | 0.99998 | MTRPOS\_START\_SEC\_VAR\_CLEARED\_16 |
| MtrPos\_CumMtrPosOff\_Deg\_M\_f32 |  | 0 | 360 | MTRPOS\_START\_SEC\_VAR\_NOINIT\_32 |
| MtrPos\_CumMtrPosInputBfr\_Cnt\_M\_u08 | 1 | 0 | 1 | MTRPOS\_START\_SEC\_VAR\_CLEARED\_8 |
| MtrPos\_CumMtrPosOffComplete\_Cnt\_M\_lgc | N/A | FALSE | TRUE | MTRPOS\_START\_SEC\_VAR\_CLEARED\_BOOLEAN |
| tempAlignedCumMtrPosMRF\_Rev\_M\_s15p16 | 2-16 | -32768 | 32767 | MTRPOS\_START\_SEC\_VAR\_NOINIT\_32 |

### User defined typedef definition/declaration

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Typedef Name | Element Name | User Defined Type | Legal Range  (min) | Legal Range  (max) |
| MtrPosCal\_DataType | BEMFCal\_Rev\_u0p16  R\_BEMFCal\_Rev\_u0p16  Sin1Offset\_Volts\_u3p13  Sin1AmpRec\_Uls\_u3p13  Cos1Offset\_Volts\_u3p13  Cos1AmpRec\_Uls\_u3p13  SinDelta1\_Uls\_s2p13  CosDelta1Rec\_Uls\_u3p13  Sin1OffCorr\_Volts\_s2p13  Sin1GainCorr\_Uls\_u1p15  Cos1OffCorr\_Volts\_s2p13  Cos1GainCorr\_Uls\_u1p15  SinHarTbl\_Cnt\_sm6p13[144]  CosHarTbl\_Cnt\_sm6p13[144] | u0p16\_T  u0p16\_T  u3p13\_T  u3p13\_T  u3p13\_T  u3p13\_T  s2p13\_T  u3p13\_T  s2p13\_T  u1p15\_T  s2p13\_T  u1p15\_T  sm6p13\_T  sm6p13\_T | 0  0  2.2  0.25  2.2  0.25  -0.0174524  0.99985  -0.5  0.8  -0.5  0.8  -0.0155  -0.0155 | 1  1  2.8  2.5  2.8  2.5  0.0174524  1  0.5  1.2  0.5  1.2  0.0155  0.0155 |

* \*In FDD 06B, Ver 6, range for SinHarTbl and CosHarTbl is given as -1 to 1 which is incorrect as it’s a 8bit signal. Informed to FDD Owner and will be updated accordingly.

# Constant Data Dictionary

## Calibration Constants

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

## Program(fixed) Constants

### Embedded Constants

#### Local

|  |  |  |  |
| --- | --- | --- | --- |
| Constant Name | Resolution | Units | Value |
| D\_ADCREF\_VOLTS\_F32 | floating point | Volts | 5 |
| D\_ADCFULLSCALE\_CNTS\_U16 | 1 | Counts | 4095 |
| D\_CNVRTP29TOP13\_CNT\_U16 | 1 | Counts | 16 |
| D\_SCALE\_VOLTSPERCOUNT\_U3P29 | 2-29 | Volts per Count | D\_ADCREF\_VOLTS\_F32/D\_ADCFULLSCALE\_CNTS\_U16 |
| D\_SIGCORRLOLMT\_ULS\_S5P26 | 2-26 | Unitless | -3 |
| D\_SIGCORRHILMT\_ULS\_S5P26 | 2-26 | Unitless | 3 |
| D\_SINCOSHILMT\_ULS\_S2P13 | 2-13 | Unitless | 2 |
| D\_SINCOSLOLMT\_ULS\_S2P13 | 2-13 | Unitless | -2 |
| D\_HARPOSROUNDFACTOR\_ULS\_U0P16 | 2-16 | Counts | 0.003472222222 |
| D\_HALFPREC13\_CNT\_S32 | 1 | Counts | 4096 |
| D\_SHIFT13\_CNT\_S16 | 1 | Counts | 13 |
| D\_SIGNBITSHIFT32\_CNT\_U16 | 1 | Counts | 31 |
| D\_HARTBLSIZE\_CNT\_U16 | 1 | Counts | 144 |
| D\_PIREV\_REV\_U1P15 | 2-15 | Rev | 0.5 |
| D\_MASK16BITS\_CNT\_U16 | 1 | Counts | 0xFFFF |

#### 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 |
| MtrPos\_EOLDataPtr\_Cnt\_M\_Str | N/A | Constant Pointer to Rte\_Pim\_MtrPosSnsr\_EOLData() | MTRPOS\_START\_SEC\_CONST\_UNSPECIFIED |

# 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. \_atan2\_asm\_
2. Limit\_m
3. Sign\_s16\_m

## Data Hiding Functions

1. Adc2\_GetSinTheta1\_Cnt\_u16\_m
2. Adc2\_GetCosTheta1\_Cnt\_u16\_m
3. Rte\_Pim\_MtrPosSnsr\_EOLData

## Global Functions/Macros Defined by this Module

1. MtrPos\_Per1

## Local Functions/Macros Used by this MDD only

### Signal Correction

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | SignalCorrect | Type | Min | Max |
| **Arguments Passed** | Signal\_Volts\_T\_u3p13 | u3p13\_T | 0 | 5 |
|  | Offset\_Volts\_T\_u3p13 | u3p13\_T | 1.7 | 3.8 |
|  | AmpRec\_Uls\_T\_u3p13 | u3p13\_T | 0.2 | 3.3 |
| **Return Value** | Signal\_Uls\_T\_s2p13 | s2p13\_T | -3 | 3 |

#### Design Rationale

This function is responsible for scaling and correcting the motor sense board sine and cosine signals based upon End Of Line calibrations. The correction uses the sensor calibrations to subtract the DC offset and normalize each signal to a nominal amplitude of +/- 1.

#### Description



### Quadrature Correction

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | QuadCorrect | Type | Min | Max |
| **Arguments Passed** | CorrSin\_Uls\_T\_s2p13 | s2p13\_T | -3 | 3 |
|  | CorrCos\_Uls\_T\_s2p13 | s2p13\_T | -3 | 3 |
|  | SinDelta\_Uls\_T\_s2p13 | s2p13\_T | -0.0174524 | 0.0174524 |
| **Return Value** | CorrCos\_Uls\_T\_s2p13 | s2p13\_T | -3 | 3 |

#### Design Rationale

This function is responsible for adjusting the cosine signal for any quadrature error between the sine and cosine signals. Ideally, the sine and cosine signals are exactly 90 degrees out of phase of eachother. This function will adjust the cosine signal based upon sensor calibrations to correct for any error from the ideal 90 degree phase difference.

The Quadrature correction should follow the following algorithm per requirements…



However, “δ” becomes a small enough number that the cos(δ) term becomes close enough to “1” to have a negligible effect on the signal. Therefore, in order to reduce throughput of this algorithm, the application of 1/cos(δ) is omitted.

#### Description



### Round and Shift by 13 Bits

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | RoundAndShift13\_s32 | Type | Min | Max |
| **Arguments Passed** | x | sint32 | -2^31 | (2^31)-(2^12)-1 |
| **Return Value** | See Function |  |  |  |

#### Design Rationale

This function will perform a rounding before applying a shift of 13 bits on the variable passed to the function. The function will use the sign bit of the input to help determine the value to be added to the signal before performing the shift operation. For a positive value, the value to be added is simply half of 2^[#bits to shift]… for this function, 13 bits are shifted off so 2^12 should be added to the signal before the shift is performed. However, for negative numbers, the value to be added is 2^[#bits to shift] -1. Therefore, the sign bit of the input is subtracted from 2^[#bits to shift], and this will result in the correct value for both positive and negative numbers.

Note that this function is designed to produce identical results to the **FPM\_FixWithRound\_m** that will cause a shift of 13 bits, but has a restricted input range as shown above.

#### Description

**return** (((x)+(D\_HALFPREC13\_CNT\_S32-((x)>>D\_SIGNBITSHIFT32\_CNT\_U16)))>>D\_SHIFT13\_CNT\_S16)

### Get Sine Voltage Macro Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | Adc2\_GetSinTheata1\_Cnt\_u16\_m | Type | Min | Max |
| **Arguments Passed** | None |  |  |  |
| **Return Value** | SinSignal\_Volts\_T\_u3p13 | U16 | 0 | 4095 |

#### Design Rationale

The actual implementation of this function is in macro form. When called, the function (macro) will return the ADC reading for the corresponding analog input.

### Get Cosine Voltage Macro Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | Adc2\_GetCosTheata1\_Cnt\_u16\_m | Type | Min | Max |
| **Arguments Passed** | None |  |  |  |
| **Return Value** | CosSignal\_Volts\_T\_u3p13 | U16 | 0 | 4095 |

#### Design Rationale

The actual implementation of this function is in macro form. When called, the function (macro) will return the ADC reading for the corresponding analog input.

# Software Module Implementation

## Runtime Environment (RTE) Initial Values

|  |  |
| --- | --- |
| Data | Value |
| Rte\_InitValue\_AlignedCumMtrPosMRF\_Deg\_f32 | 0 |
| Rte\_InitValue\_CumMtrPosMRF\_Deg\_f32 | 0 |
| Rte\_InitValue\_CumMtrPosCRF\_Deg\_f32 | 0 |
| Rte\_InitValue\_SysCCumMtrPosMRF\_Deg\_f32 | 0 |
| Rte\_InitValue\_SysCCumMtrPosCRF\_Deg\_f32 | 0 |

## Initialization Functions

### Init: \_Init1

#### Design Rationale

None

#### Module Internal and Outputs



## Periodic Functions

### Per: \_Per1

#### Design Rationale

The computation of AlignedCumMtrPosMRF\_Rev\_M\_s15p16 is optimized to take advantage of the fact that the relative position is sized to be a modulo 65536 value (i.e. rolls over at the16 bit value boundary). The operation specified to the compiler is to:

* Perform an unsigned subtraction of the z-1 and current position
* Explicitly mask the subtraction result to 16bits to ensure, that in the event that the compiler determines that a 32 bit subtraction is appropriate, the result is masked to only provide the lower 16 bits of the result.
* Cast the unsigned result of the mask as a signed 16 bit value so that rollovers occurring in the subtraction correctly indicate the integer amount of change.
* Finally cast the expression up to this point to a signed 32 bit number to promote the signed 16 integer and sign extend it to allow it to be added properly to the 32 bit accumulator, AlignedCumMtrPosMRF\_Rev\_M\_s15p16

The above method eliminates any need for a conditional branch expression to handle the rollover points, thus avoiding the branching penalty.

#### Program Flow Start

N/A

#### Store Module Inputs to Local copies



#### Signal Processing



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



#### Program Flow End

N/A

### Per: \_Per2

#### Design Rationale

This function accesses the AlignedCumMtrPosMRF\_Rev\_M\_s15p16 signal produced by Per1 which is running in the MtrCtrl ISR. Ther is no concern for data consistency in this case because the signal is 32 bits wide and it is assumed that the target processor supports atomic 32 data accesses (i.e as is the case for the TMS570).

This function should be run before MtrPos3\_Per1, as the comparisons (in both the main and diverse paths) on the generated signals assume that this order will be followed.

#### Program Flow Start

Rte\_Call\_MtrPos2\_Per1\_CP0\_CheckpointReached()

#### Store Module Inputs to Local copies

AssistAsmPolarity\_T\_f32 = Rte\_IRead\_MtrPos\_Per2\_AssistAssemblyPolarity\_Cnt\_s08()

#### Processing of function



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

Rte\_IWrite\_MtrPos\_Per2\_AlignedCumMtrPosMRF\_Deg\_f32(AlignedCumMtrPosMRF\_Deg\_T\_f32)

Rte\_IWrite\_MtrPos\_Per2\_CumMtrPosMRF\_Deg\_f32(CumMtrPosMRF\_Deg\_T\_f32)

Rte\_IWrite\_MtrPos\_Per2\_SysCCumMtrPosMRF\_Deg\_f32(CumMtrPosMRF\_Deg\_T\_f32)

Rte\_IWrite\_MtrPos\_Per2\_CumMtrPosCRF\_Deg\_f32(CumMtrPosMRF\_Deg\_T\_f32 \* AssistAsmPolarity\_T\_f32)

Rte\_IWrite\_MtrPos\_Per2\_SysCCumMtrPosCRF\_Deg\_f32(CumMtrPosMRF\_Deg\_T\_f32 \* AssistAsmPolarity\_T\_f32)

#### Program Flow End

N/A

## Fault Recovery Functions

None

## Shutdown Functions

None

## Interrupt Functions

None

## Serial Communication Functions

### SCom: \_SCom\_ReadEOLMtrCals

|  |  |
| --- | --- |
| **Arguments Passed** | Type |
| MtrCalDataPtr | MtrPosCal\_DataType \* |

#### Design Rationale

None

#### Program Flow Start

N/A

#### Store Module Inputs to Local copies

None

#### Processing of function



### SCom: \_SCom\_SetEOLMtrCals

|  |  |
| --- | --- |
| **Arguments Passed** | Type |
| MtrCalDataPtr | MtrPosCal\_DataType \* |

#### Design Rationale

#### Provide exclusive area around EOL Data update to ensure the data stays consistent. If the BEMF R/C check, for example, pre-empts this update while the BEMF and R\_BEMF values are inconsistent, a false failure will be detected. For simplicity, this implementation of the exclusive area protection does not provide a configurable type of protection, but rather always uses Suspend/Resume interrupts. If the need arises in the future this design can be updated.

#### Program Flow Start

N/A

#### Store Module Inputs to Local copies

None

#### Processing of function



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

N/A

#### Program Flow End

N/A

# Execution Requirements

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

|  |  |  |
| --- | --- | --- |
| Function Name | Calling Frequency | System State(s) in which the function is called |
| \_Init1 | Once | COLD\_INIT |
| \_Per1 | 62.5 us | ALL |
| \_Per2 | 1 ms | ALL |

## Execution Requirements for Serial Communication Functions

|  |  |
| --- | --- |
| Function Name | Sub-Module called by (Serial Comm Function Name) |
| MtrPos2\_SCom\_ReadEOLMtrCals |  |
| MtrPos2\_SCom\_SetEOLMtrCals |  |

# Memory Map Definition Requirements

## Sub Modules (Functions)

|  |  |
| --- | --- |
| Name of Sub Module | Software Segment |
| \_Init1 | RTE\_START\_SEC\_SA\_MTRPOS\_APPL\_CODE |
| \_Per1 |  |
| \_Per2 |  |

## Local Functions

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

|  |  |
| --- | --- |
| Name of Sub Module | Software Segment |
| SignalCorrect |  |
| QuadCorrect |  |
| RoundAndShift13\_s32 |  |

# Known Issues / Limitations With Design

1. INLINE functions in GlobalMacro.h are not unit tested

# Revision Control Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Rev #** | **Change Description** | **Date** | **Author Initials** |
| 1 | 1 | Initial MDD version. | 11 Jul 12 | JWJ |
| 2 | 2 | Added missing input/output/module internal variables | 13 Jul 12 | JWJ |
| 3 | 3 | Implemented FDD 06B v004 | 23-Oct-12 | OT |
| 4 | 4 | UTP Updates | 21-Nov-12 | OT |
| 5 | 5 | Updated Component design methodology to encapsulate component internal data flow outside of the Rte. | 20-Feb-13 | JJW |
| 6 | 6 | Changed cumulative position algorithm to eliminate the rollover synchronization issue between the atan and rollover detection methods. | 20-Mar-13 | JJW |
| 7 | 7 | MDD updates as per Src ver 7 | 14-june-13 | NRAR |