**Module Design Document**

**For**

**Motor Angle 1 Measurement**

**May 04, 2018**

**Prepared By:**

**Avinash James**

**Software Engineering**

**Nexteer Automotive,**

**Saginaw, MI, USA**

**Change History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Author** | **Version** | **Date** |
| Initial Version | Avinash James | 1.0 | 07-Jun-2016 |
| Removed local function CalcTurnCntr | Avinash James | 2.0 | 13-Jun-2016 |
| Added local function ProcTurnCntrReg. Added constants required for bitshifting and signal limiting. Added limiting block for output MotAg1TurnCntr. | Brendon Binder | 3.0 | 25-Aug-2017 |
| Update to FDD 4.1.0 added MotAg1WarnReg function | Mateusz Bartocha | 4.0 | 23-Oct-17 |
| Fix – Missed MotAg1TurnCntrRollgCntr output | Mateusz Bartocha | 5 | 21-Nov-17 |
| Updated Diagram & Unit Test Considerations | Matthew Leser | 6.0 | 14-Dec-2017 |
| Updated as per Design version 5.0.0 | Krzysztof Byrski | 7.0 | 25-Apr-2018 |
| Added ports for IO access | Avinash James | 8.0 | 27-Apr-2018 |
| Added updates for sensor offset learning | Avinash James | 9.0 | 04-May-2018 |

Table of Contents

[1 Introduction 5](#_Toc512444999)

[1.1 Purpose 5](#_Toc512445000)

[1.2 Scope 5](#_Toc512445001)

[2 MotAg1Meas High-Level Description 6](#_Toc512445002)

[3 Design details of software module 7](#_Toc512445003)

[3.1 Graphical representation of MotAg1Meas 7](#_Toc512445004)

[3.2 Data Flow Diagram 7](#_Toc512445005)

[3.2.1 Component level DFD 7](#_Toc512445006)

[3.2.2 Function level DFD 7](#_Toc512445007)

[4 Constant Data Dictionary 8](#_Toc512445008)

[4.1 Program (fixed) Constants 8](#_Toc512445009)

[4.1.1 Embedded Constants 8](#_Toc512445010)

[5 Software Component Implementation 9](#_Toc512445011)

[5.1 Sub-Module Functions 9](#_Toc512445012)

[5.1.1 Init: MotAg1MeasInit1 9](#_Toc512445013)

[5.1.1.1 Design Rationale 9](#_Toc512445014)

[5.1.1.2 Module Outputs 9](#_Toc512445015)

[5.1.1.3 Module Internal 9](#_Toc512445016)

[5.1.2 Per: MotAg1MeasPer1 9](#_Toc512445017)

[5.1.2.1 Design Rationale 9](#_Toc512445018)

[5.1.2.2 Store Module Inputs to Local copies 9](#_Toc512445019)

[5.1.2.3 (Processing of function)……… 9](#_Toc512445020)

[5.1.2.4 Store Local copy of outputs into Module Outputs 9](#_Toc512445021)

[5.1.3 Per: MotAg1MeasPer2 9](#_Toc512445022)

[5.1.3.1 Design Rationale 9](#_Toc512445023)

[5.1.3.2 Store Module Inputs to Local copies 10](#_Toc512445024)

[5.1.3.3 (Processing of function)……… 10](#_Toc512445025)

[5.1.3.4 Store Local copy of outputs into Module Outputs 10](#_Toc512445026)

[5.2 Server Runnables: MotAg1CoeffTblRead 10](#_Toc512445027)

[5.2.1.1 Design Rationale 10](#_Toc512445028)

[5.2.1.2 Store Module Inputs to Local copies 10](#_Toc512445029)

[5.2.1.3 (Processing of function)……… 10](#_Toc512445030)

[5.2.1.4 Store Local copy of outputs into Module Outputs 10](#_Toc512445031)

[5.3 Server Runnables: MotAg1CoeffTblWr 10](#_Toc512445032)

[5.3.1.1 Design Rationale 10](#_Toc512445033)

[5.3.1.2 Store Module Inputs to Local copies 10](#_Toc512445034)

[5.3.1.3 (Processing of function)……… 10](#_Toc512445035)

[5.3.1.4 Store Local copy of outputs into Module Outputs 10](#_Toc512445036)

[5.4 Server Runnables: MotAg1CfgLoPwrMod 10](#_Toc512445037)

[5.4.1.1 Design Rationale 10](#_Toc512445038)

[5.4.1.2 Store Module Inputs to Local copies 10](#_Toc512445039)

[5.4.1.3 (Processing of function)……… 11](#_Toc512445040)

[5.4.1.4 Store Local copy of outputs into Module Outputs 11](#_Toc512445041)

[5.5 Interrupt Functions 11](#_Toc512445042)

[5.6 Module Internal (Local) Functions 11](#_Toc512445043)

[5.6.1 ProcessErrorRegAndDieRevCtr 11](#_Toc512445140)

[5.6.2 SPI\_AnglePolarityAdjust 11](#_Toc512445141)

[5.6.3 SPIvsENCA 12](#_Toc512445142)

[5.6.4 CalcCorrnTbl 12](#_Toc512445143)

[5.6.5 MotAgFaultProcessing 12](#_Toc512445144)

[5.6.6 CalcNtcPrm 13](#_Toc512445145)

[5.6.7 SetMotAg1FltNtc 13](#_Toc512445146)

[5.6.8 OffsetCalculation 14](#_Toc512445147)

[5.6.9 CalculateMotAgTurnCntr 14](#_Toc512445148)

[5.6.10 SPI\_AngleRawProcess 15](#_Toc512445149)

[5.6.11 CompensateMechMtrPos 15](#_Toc512445150)

[5.7 GLOBAL Function/Macro Definitions 16](#_Toc512445151)

[6 Known Limitations with Design 17](#_Toc512445152)

[7 UNIT TEST CONSIDERATION 18](#_Toc512445153)

[Appendix A Abbreviations and Acronyms 19](#_Toc512445154)

[Appendix B Glossary 20](#_Toc512445155)

# Introduction

## Purpose

This document defines the module level design for the Sensor Offset and Correction Component. Major part of design has been captured in the FDD and any design rationale that has not been identified in the FDD and has been used to implement the component has been documented in the MDD

## Scope

The following definitions are used throughout this document:

* **Shall**: indicates a mandatory requirement without exception in compliance.
* **Should**: indicates a mandatory requirement; exceptions allowed only with documented justification.
* **May**: indicates an optional action.

# MotAg1MeasHigh-Level Description

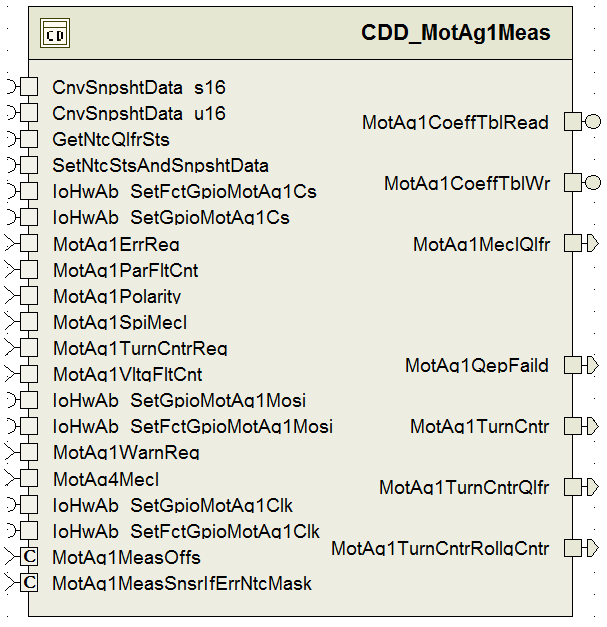
The CDD\_MotAg1Meas component is the complex driver for the motor angle 1 measurement subsystem. This function initializes the registers for CSIH3 SPI channel for communicating with the motor angle 1 measurement sensor board. The SPI transmission is triggered periodically by DMA component. This function receives the RAW sensor data at 62.5uS rate. The component contains two source files, both described in this MDD: CDD\_MotAg1Meas.c contains the RTE runnables and services; CDD\_MotAg1Meas\_MotCtrl.c contains the motor control runnable.

.

# Design details of software module

See FDD.

## Graphical representation of MotAg1Meas



## Data Flow Diagram

See FDD.

### Component level DFD

See FDD.

### Function level DFD

See FDD.

# Constant Data Dictionary

## Program (fixed) Constants

### Embedded Constants

#### Local Constants

|  |  |  |  |
| --- | --- | --- | --- |
| Constant Name | Resolution | Units | Value |
| MOTAG1TURNCNTRLOLIM\_CNT\_F32 | Single point | Cnt | -256.09375 |
| MOTAG1TURNCNTRHILIM\_CNT\_F32 | Single point | Cnt | 255.96875 |
| MOTCTRLMOTAG1WARNREGLOLIM\_CNT\_U32 | 1 | Cnt | 0 |
| MOTCTRLMOTAG1WARNREGHILIM\_CNT\_U32 | 1 | Cnt | 67108863 |
| MOTCTRLMOTAG1ERRREGLOLIM\_CNT\_U32 | 1 | Cnt | 0 |
| MOTCTRLMOTAG1ERRREGHILIM\_CNT\_U32 | 1 | Cnt | 67108863 |
| MOTCTRLMOTAG1TURNCNTRREGLOLIM\_CNT\_U32 | 1 | Cnt | 0 |
| MOTCTRLMOTAG1TURNCNTRREGHILIM\_CNT\_U32 | 1 | Cnt | 67108863 |

**\* Also see see FDD – CM640B\_MotAg1Meas\_DataDict.m file**

# Software Component Implementation

## Sub-Module Functions

The sub-module functions are grouped based on similar functionality that needs to be executed in a given “State” of the system (refer States and Modes). For a given module, the MDD will identify the type and number of sub-modules required. The sub-module types are described below.

### Init: MotAg1MeasInit1

## Design Rationale

All register initialization that is allowed at the register level (see Register/Field column of the CM640A\_MotAg1Meas\_RegisterConfiguration.xlsm spreadsheet in the FDD) is done at the register level to save execution time as compared to the read/modify/writes that would be needed to initialize at the field level. Field level initialization done only where required by the spreadsheet.

## Module Outputs

See FDD: MotAg1MeasInit1 model block.

## Module Internal

See FDD: MotAg1MeasInit1 model block for Per Instance Memory.

### Per: MotAg1MeasPer1

## Design Rationale

For run time efficiency in the motor control loop the **Compensate MechMtrPos** block is implemented in a optimized way in the code by letting a uint16 variable be overflown

## Store Module Inputs to Local copies

See FDD: MotAg1MeasPer1 model block

## (Processing of function)………

See FDD: MotAg1MeasPer1 model block.

## Store Local copy of outputs into Module Outputs

See FDD: MotAg1MeasPer1 model block.

### Per: MotAg1MeasPer2

## Design Rationale

Details of the implementation of block “Process MotAg1RawErr” differ from the model in order to meet design and coding standards.

## Store Module Inputs to Local copies

See FDD: MotAg1MeasPer2 model block

## (Processing of function)………

See FDD: MotAg1MeasPer2 model block.

## Store Local copy of outputs into Module Outputs

See FDD: MotAg1MeasPer2 model block.

## Server Runnables: MotAg1CoeffTblRead

## Design Rationale

None

## Store Module Inputs to Local copies

None

## (Processing of function)………

*See* MotAg1CoeffTblRead *block in the FDD*

## Store Local copy of outputs into Module Outputs

None

## Server Runnables: MotAg1CoeffTblWr

## Design Rationale

None

## Store Module Inputs to Local copies

None

## (Processing of function)………

*See* MotAg1CoeffTblWr *block in the FDD*

## Store Local copy of outputs into Module Outputs

*See* MotAg1MeasMotAg1CoeffTblWr*block in the FDD*

## Server Runnables: MotAg1CfgLoPwrMod

## Design Rationale

None

## Store Module Inputs to Local copies

None

## (Processing of function)………

*See* *MotAg1CfgLoPwrMod* *block in the FDD*

## Store Local copy of outputs into Module Outputs

*See MotAg1CfgLoPwrMo*d *block in the FDD*

## Interrupt Functions

None

## Module Internal (Local) Functions

### ProcessErrorRegAndDieRevCtr

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | ProcessErrorRegAndDieRevCtr | Type | Min | Max |
| **Arguments Passed** | MotAgErrReg\_Cnt\_T\_u32 | uint32 | 0 | 67108863 |
|  | MotAgTurnCntrReg\_Cnt\_T\_u32 | uint32 | 0 | 67108863 |
|  | MotAgWarnReg\_Cnt\_T\_u32 | uint32 | 0 | 67108863 |
| **Return Value** | MotAg1Err\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | MotAg1Warn\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | TurnCntrParFltCnt\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | TurnCntr\_Cnt\_T\_s16 | sint16 | -2048 | 2048 |
|  | MotAgTurnCntrRollgCntr\_Cnt\_T\_u08 | uint8 | 0 | 65535 |

#### Design Rationale

Implementation of "Process ErrorReg and DieRevCtr" Simulink block



### SPIvsENCA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | SPIvsENCA | Type | Min | Max |
| **Arguments Passed** | MotAgSpiMecl\_MotRev\_T\_u0p16 | u0p16 | 0 | 65535 |
|  | MotAgEncaMecl\_MotRev\_T\_u0p16 | u0p16 | 0 | 65535 |
| **Return Value** | MotAgSyncErr\_Cnt\_T\_u08 | uint8 | 0 | 1 |

#### Design Rationale

Implementation of "SPI vs ENCA" Simulink block

### CalcCorrnTbl

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | CalcCorrnTbl | Type | Min | Max |
| **Arguments Passed** | None | N/A | N/A | N/A |
| **Return Value** | N/A | N/A | N/A | N/A |

#### Design Rationale

See “Calculate Correction Table” block in the Simulink model of the design

### MotAgFaultProcessing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | MotAgFaultProcessing | Type | Min | Max |
| **Arguments Passed** | MotAgErr\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | MotAgWarn\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | TurnCntrParFltCnt\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | MotAgVltgFltCnt\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | MotAgParFltCnt\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | MotAgSyncErr\_Cnt\_T\_u16 | uint8 | 0 | 1 |
|  | MotAg4Mecl\_MotRev\_T\_u0p16 | u0p16 | 0 | 65535 |
|  | MotAgSpiMecl\_MotRev\_T\_u0p16 | u0p16 | 0 | 65535 |
|  | TurnCntr\_Cnt\_T\_s16 | sint16 | -2048 | 2048 |
| **Return Value** | MotAgQlfr\_Cnt\_T\_enum | SigQlfr1 | 0 | 2 |
|  | MotAgTurnCntrQlfr\_Cnt\_T\_enum | SigQlfr1 | 0 | 2 |
|  | MotAgQepFaild\_Cnt\_T\_logl | boolean | FALSE | TRUE |

#### Design Rationale

Implementation of "MotAg Fault Processing" Simulink block

### CalcNtcPrm

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | CalcNtcPrm | Type | Min | Max |
| **Arguments Passed** | MotAgErr\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | MotAgWarn\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | TurnCntrParFltCnt\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | MotAgVltgFltCnt\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
|  | MotAgParFltCnt\_Cnt\_T\_u16 | uint16 | 0 | 65535 |
| **Return Value** | MotAgNtcParm\_Cnt\_T\_u08 | uint8 | 0 | 127 |
|  | TurnCtrNtcParm\_Cnt\_T\_u08 | uint8 | 0 | 255 |
|  | TurnCtrVltgNtcParm\_Cnt\_T\_u08 | uint8 | 0 | 1 |

#### Design Rationale

See “Determine NTC Parameters” block in the Simulink model of the design

### SetMotAg1FltNtc

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | SetMotAg1FltNtc | Type | Min | Max |
| **Arguments Passed** | MotAgNtcParm\_Cnt\_T\_u08 | uint8 | 0 | 127 |
|  | TurnCntrNtcParm\_Cnt\_T\_u08 | uint8 | 0 | 255 |
|  | TurnCntrVltgNtcParm\_Cnt\_T\_u08 | uint8 | 0 | 1 |
|  | MotAgQepMecl\_MotRev\_T\_u0p16 | u0p16 | 0 | 65535 |
|  | MotAgSPIMecl\_\_MotRev\_T\_u0p16 | u0p16 | 0 | 65535 |
|  | TurnCntr\_Cnt\_T\_s16 | sint16 | -2048 | 2048 |
| **Return Value** | N/A | N/A | N/A | N/A |

#### Design Rationale

See “MotAg1PtrclFlt Processing, MotAg1TurnCntrFlt Processing & MotAg1TurnCntrVltgFlt Processing” block in the Simulink model of the design



### 

### CalculateMotAgTurnCntr

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | CalculateMotAgTurnCntr | Type | Min | Max |
| **Arguments Passed** | MotAgSpiMecl\_MotRev\_T\_u0p16 | u0p16 | 0 | 65535 |
|  | TurnCntr\_Cnt\_T\_s16 | sint16 | -2048 | 2048 |
|  | MotAgPolarity\_Cnt\_T\_s08 | sint8 | -1 | 1 |
| **Return Value** | MotAgTurnCntr\_Cnt\_T\_f32 | float32 | -256.09375 | 255.96875 |

#### Design Rationale

Implementation of "Calculate MotAgTurnCntr" Simulink block

### SPI\_AngleRawProcess

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | SPI\_AngleRawProcess | Type | Min | Max |
| **Arguments Passed** | RawAngReg\_Cnt\_T\_u32 | uint32 | 0 | 67108863 |
| **Return Value** | N/A | N/A | N/A | N/A |

#### Design Rationale

Implementation of "SPI\_AngleRawProcess" Simulink block

### CompensateMechMtrPos

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | CompensateMechMtrPos | Type | Min | Max |
| **Arguments Passed** | MotAgRawMecl\_Cnt\_T\_u0p16 | u0p16 | 0 | 65535 |
| **Return Value** | MotAgMecl\_MotRev\_T\_u0p16 | u0p16 | 0 | 65535 |

#### Design Rationale

Implementation of "Compensate MechMtrPos" Simulink block

## GLOBAL Function/Macro Definitions

None

# Known Limitations with Design

Analysis was done to ensure the datatype sm5p12 is sufficient to hold worst case values for the MotAg1CorrlnTbl.  This analysis will be added to the FDD.  See CR EA4#14937.  In addition, the correlation table values are verified in manufacturing after writing the coefficient table to NVM.

# UNIT TEST CONSIDERATION

Following variables are used as rolling counters, so the overflow of these variables is intentional.

Rte\_Pim\_MotAg1VltgFltCntPrev

Rte\_Pim\_MotAg1ParFltCntPrev

Rte\_Pim\_MotAg1MeclRollgCntrPrev

Rte\_Pim\_MotAg1TurnCntrParFltCntPrev

There is code in the SinCos\_f32() function which will not be covered in this component, because the input to this function is always positive. This is ok as the SinCos\_f32() is a library function.

For the config param MOTAG1MEAS\_DIQEPIF\_CNT\_LOGL please use the file CDD\_MotAg1Meas\_Cfg.h in the include folder.

Abbreviations and Acronyms

| **Abbreviation or Acronym** | **Description** |
| --- | --- |
|  |  |
|  |  |

Glossary

**Note**: Terms and definitions from the source “Nexteer Automotive” take precedence over all other definitions of the same term. Terms and definitions from the source “Nexteer Automotive” are formulated from multiple sources, including the following:

* ISO 9000
* ISO/IEC 12207
* ISO/IEC 15504
* Automotive SPICE® Process Reference Model (PRM)
* Automotive SPICE® Process Assessment Model (PAM)
* ISO/IEC 15288
* ISO 26262
* IEEE Standards
* SWEBOK
* PMBOK
* Existing Nexteer Automotive documentation

| **Term** | **Definition** | **Source** |
| --- | --- | --- |
| MDD | Module Design Document |  |
| DFD | Data Flow Diagram |  |

References

| **Ref. #** | **Title** | **Version** |
| --- | --- | --- |
| 1 | AUTOSAR Specification of Memory Mapping (Link:AUTOSAR\_SWS\_MemoryMapping.pdf) | v1.3.0 R4.0 Rev 2 |
| 2 | MDD Guideline | EA4 01.00.00 |
| 3 | EA4 Software Naming Conventions | 01.01.00 |
| 4 | Software Design and Coding Standards.doc | 2.1 |
| 5 | FDD – CM640B Motor Angle 1 Measure | See Synergy subproject version |