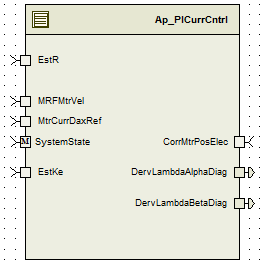
# Module – Torque Reasonable Diagnostics

# High-Level Description

The Torque Reasonableness Diagnostic compares the commanded electromagnetic motor torque (calculated from the commanded Iq and Id currents) to the measured electromagnetic torque (calculated from the measured Iq and Id currents) and sets a diagnostic flag when the error is outside of calibration boundaries for calibration time periods. This diagnostic is intended to trip due to a variety of possible errors within the closed loop control of the motor control, including but not exclusive to certain current measurement errors and output drive errors.

# 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 |
| DervLambdaAlphaDiag\_Volt\_f32 |  |
| DervLambdaBetaDiag\_Volt\_f32 |  |
| OutputRampMult\_Uls\_f32 |  |
| TrqLimitMin\_MtrNm\_f32 |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## 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 |
| TqRsDg\_AlpaCurrDiagPrimLPF\_M\_Str | LPF32KSV\_Str | See data dictionary | See data dictionary | TQRSDG\_START\_SEC\_VAR\_NOINIT\_UNSPECIFIED |
| TqRsDg\_BetaCurrDiagPrimLPF\_M\_Str | LPF32KSV\_Str | See data dictionary | See data dictionary | TQRSDG\_START\_SEC\_VAR\_NOINIT\_UNSPECIFIED |
| TqRsDg\_AlpaCurrDiagSecLPF\_M\_Str | LPF32KSV\_Str | See data dictionary | See data dictionary | TQRSDG\_START\_SEC\_VAR\_NOINIT\_UNSPECIFIED |
| TqRsDg\_BetaCurrDiagSecLPF\_M\_Str | LPF32KSV\_Str | See data dictionary | See data dictionary | TQRSDG\_START\_SEC\_VAR\_NOINIT\_UNSPECIFIED |
| TqRsDg\_CurrDiagPrimPNAccum\_Cnt\_M\_u16 | 1 | See data dictionary | See data dictionary | TQRSDG\_START\_SEC\_VAR\_CLEARED\_16 |
| TqRsDg\_CurrDiagSecPNAccum\_Cnt\_M\_u16 | 1 | See data dictionary | See data dictionary | TQRSDG\_START\_SEC\_VAR\_CLEARED\_16 |
| TqRsDg\_DervLambdaAlphaDiagPrimFilt\_Volt\_D\_f32 | Single precision float | See data dictionary | See data dictionary | TQRSDG\_START\_SEC\_VAR\_CLEARED\_32 |
| TqRsDg\_DervLambdaBetaDiagPrimFilt\_Volt\_D\_f32 | Single precision float | See data dictionary | See data dictionary | TQRSDG\_START\_SEC\_VAR\_CLEARED\_32 |
| TqRsDg\_DervLambdaAlphaDiagSecFilt\_Volt\_D\_f32 | Single precision float | See data dictionary | See data dictionary | TQRSDG\_START\_SEC\_VAR\_CLEARED\_32 |
| TqRsDg\_DervLambdaBetaDiagSecFilt\_Volt\_D\_f32 | Single precision float | See data dictionary | See data dictionary | TQRSDG\_START\_SEC\_VAR\_CLEARED\_32 |
| TqRsDg\_DervLambdaAlphaDiagLmtd\_Volt\_D\_f32 | Single precision float | See data dictionary | See data dictionary | TQRSDG\_START\_SEC\_VAR\_CLEARED\_32 |
| TqRsDg\_DervLambdaBetaDiagLmtd\_Volt\_D\_f32 | Single precision float | See data dictionary | See data dictionary | TQRSDG\_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) |
|  |  |  |  |  |

# 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\_CurrDiagPrimLPFKn\_Hz\_f32 |
| k\_CurrDiagSecLPFKn\_Hz\_f32 |
| k\_CurrDiagMtrEnvTblMax\_MtrNm\_f32 |
| k\_CurrDiagSecTrqLmtThresh\_Uls\_f32 |
| k\_CurrDiagPrimErrorThresh\_Volt\_f32 |
| k\_CurrDiagPrim\_Cnt\_Str |
| k\_CurrDiagSec\_Cnt\_Str |
| k\_CurrDiagSecErrorThresh\_Volt\_f32 |
| k\_CurrDiagLambdaMinMax\_Volt\_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\_BIT0\_ULS\_U8 | Uint8 | Cnt | 1 |
| D\_BIT1\_ULS\_U8 | Uint8 | Cnt | 2 |
|  |  |  |  |

#### 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\_ZERO\_CNT\_U8 |
| D\_2MS\_SEC\_F32 |
| D\_ONE\_ULS\_F32 |
| D\_ZERO\_CNT\_U16 |
| D\_ZERO\_ULS\_F32 |

### Module specific Lookup Tables Constants

|  |  |  |  |
| --- | --- | --- | --- |
| Constant Name | Resolution | Value | Software Segment |
|  |  |  |  |

# 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. Min\_m
2. Abs\_f32\_m
3. Limit\_m
4. LPF32KSV\_Str
5. DiagSettings\_Str
6. LPF\_OpUpdate\_f32\_m
7. LPF\_Init\_f32\_m
8. DiagNStep\_m
9. DiagPStep\_m
10. DiagFailed\_m

## Data Hiding Functions

1. Rte\_Call\_NxtrDiagMgr\_SetNTCStatus

## Global Functions/Macros Defined by this Module

### Global Function #1

None

## Local Functions/Macros Used by this MDD only

### Local Function #1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function Name** | (Exact name used) | Type | Min | Max | UTP Tol. |
| **Arguments Passed** | (if none, write None) |  |  |  |  |
|  | (Insert more rows for additional passed arguments) |  |  |  |  |
| **Return Value** | (if no value returned, write 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\_Init\_TqRsDg\_Per1\_DervLambdaAlphaDiag\_Volt\_f32 | 0.0f |
| Rte\_Init\_TqRsDg\_Per1\_DervLambdaBetaDiag\_Volt\_f32 | 0.0f |
| Rte\_Init\_TqRsDg\_Per1\_OutputRampMult\_Uls\_f32 | 0.0f |
| Rte\_Init\_TqRsDg\_Per1\_TrqLimitMin\_MtrNm\_f32 | 0.0f |

## Initialization Functions

### Init: TqRsDg\_Init1 \_Init

#### Design Rationale

None

#### Module Outputs

#### Module Internal

None

#### Processing of function



## Periodic Functions

### Per: TrRsDg\_Per1

#### Design Rationale

None

#### Program Flow Start

Rte\_Call\_TqRsDg\_Per1\_CP0\_CheckpointReached();

#### Store Module Inputs to Local copies

DervLambdaAlphaDiag\_Volt\_T\_f32 = Rte\_IRead\_TqRsDg\_Per1\_DervLambdaAlphaDiag\_Volt\_f32();

DervLambdaBetaDiag\_Volt\_T\_f32 = Rte\_IRead\_TqRsDg\_Per1\_DervLambdaBetaDiag\_Volt\_f32();

OutputRampMult\_Uls\_T\_f32 = Rte\_IRead\_TqRsDg\_Per1\_OutputRampMult\_Uls\_f32();

TrqLimitMin\_MtrNm\_T\_f32 = Rte\_IRead\_TqRsDg\_Per1\_TrqLimitMin\_MtrNm\_f32();

#### Processing of function



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

ModulationIndex\_Uls\_M\_f32 = ModulationIndex\_Uls\_T\_f32

#### Program Flow End

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

## Fault Recovery Functions

None

## Shutdown Functions

None

## Interrupt Functions

None

## Serial Communication Functions

### SCom: ModuleName\_SCom

#### Design Rationale

None

#### Program Flow Start

N/A

#### Store Module Inputs to Local copies

None

#### Processing of function

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

None

#### Program Flow End

N/A

# 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 |
| TqRsDg\_Init1 | On init | None |
| TqRsDg\_Per1 | 2ms | OPERATE |

## Execution Requirements for Serial Communication Functions

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

# 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 |
| TqRsDg\_Init1 | RTE\_START\_SEC\_AP\_TQRSDG\_APPL\_CODE |
| TqRsDg\_Per1 | RTE\_START\_SEC\_AP\_TQRSDG\_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 defined in GlobalMacro.h are not unit tested.

# Revision Control Log

|  |  |  |  |
| --- | --- | --- | --- |
| **Rev #** | **Change Description** | **Date** | **Author Initials** |
| 1.0 | Initial Version | 6-Nov-12 | Selva |
| 2.0 | Updated to version to SF-31 version 2 | 23-Feb-13 | Selva |
| 3.0 | Anomaly fix for A\_4644 | 25-Mar-13 | Srikanth |
| 4.0 | Fixes for Unit test findings | 15-Apr-13 | Srikanth |
| 5.0 | Anomaly fix for 4931 | 30-Apr-13 | OT |
| 6 | Updated for V3 SF31 (completely new) | 27-Nov-13 | Selva |
| 7 | Updated to SF-31 version 4 | 08-Jan-14 | VT |