# Module --

# High-Level Description

Key system inputs including handwheel torque, motor position, motor and handwheel velocity are widely distributed and used by command path and other safety critical system functions. The safety strategy for these “global” input signals is a dual channel diverse calculation with cross check to detect a systematic design fault that would propagate to the receiving functions. However, since the dual channel diversity does not continue along the command path out to the motor command output, this strategy cannot fully cover a potential systematic overwrite of the global signals after the cross checks are performed and downstream functions are then executed and use the primary path global signals. Therefore, an additional safety function has been defined to check for software over-write by co-existing software within the software memory partition where the primary global signals are calculated.

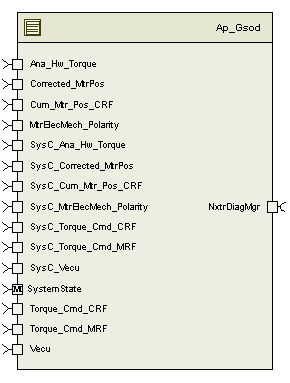
Each of the functions that generate a global input signal has been defined to store a copy of the output. The purpose of this function is to perform a comparison check of the primary signal (that was subsequently used by various system functions) and it’s redundantly stored copy. Any detected miscompare will generate an overwrite fault flag from this function to be evaluated by the diagnostics manager and result in performing an F1 shutdown response.

**Design rationale note:**

This module uses direct input reads instead of buffered reads to disable interrupts around data collection of the inputs to ensure that the data set is consistent. The limitation of this method is that it is only possible if inputs are being written from a task at the same or higher priority with the same or faster loop execution time; which is the case at the time of this implementation.

# 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 | |
| Corrected\_MtrPos\_Rev\_f32 | |  |
| Ana\_Hw\_Torque\_HwNm\_f32 | |  |
| Vecu\_Volt\_f32 | |  |
| Torque\_Cmd\_CRF\_MtrNm\_f32 | |  |
| Torque\_Cmd\_MRF\_MtrNm\_f32 | |  |
| Cum\_Mtr\_Pos\_CRF\_Deg\_f32 | |  |
| MtrElecMech\_Polarity\_Cnt\_s08 | |  |
| SysC\_Corrected\_MtrPos\_Rev\_f32 | |  |
| SysC\_Ana\_Hw\_Torque\_HwNm\_f32 | |  |
| SysC\_Vecu\_Volt\_f32 | |  |
| SysC\_Torque\_Cmd\_CRF\_MtrNm\_f32 | |  |
| SysC\_Torque\_Cmd\_MRF\_MtrNm\_f32 | |  |
| SysC\_Cum\_Mtr\_Pos\_CRF\_Deg\_f32 | |  |
| SysC\_MtrElecMech\_Polarity\_Cnt\_s32 | |  |

## 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 |
| MtrPos\_OverwriteFlt\_Cnt\_D\_lgc | Boolean | FALSE | TRUE | GSOD\_START\_SEC\_VAR\_CLEARED\_BOOLEAN |
| TorqCmd\_CRF\_OverwriteFlt\_Cnt\_D\_lgc | Boolean | FALSE | TRUE | GSOD\_START\_SEC\_VAR\_CLEARED\_BOOLEAN |
| HwTorque\_OverwriteFlt\_Cnt\_D\_lgc | Boolean | FALSE | TRUE | GSOD\_START\_SEC\_VAR\_CLEARED\_BOOLEAN |
| TorqCmd\_MRF\_OverwriteFlt\_Cnt\_D\_lgc | Boolean | FALSE | TRUE | GSOD\_START\_SEC\_VAR\_CLEARED\_BOOLEAN |
| Vecu\_OverwriteFlt\_Cnt\_D\_lgc | Boolean | FALSE | TRUE | GSOD\_START\_SEC\_VAR\_CLEARED\_BOOLEAN |
| CumMtrPos\_CRF\_OverwriteFlt\_Cnt\_D\_lgc | Boolean | FALSE | TRUE | GSOD\_START\_SEC\_VAR\_CLEARED\_BOOLEAN |
| MtrElecMech\_Polarity\_OverwriteFlt\_Cnt\_D\_lgc | Boolean | FALSE | TRUE | GSOD\_START\_SEC\_VAR\_CLEARED\_BOOLEAN |

### 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) |
| <none> |  |  |  |  |
|  |  |  |  |  |

# 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 |
| <None> |
|  |

## 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 |
| <None> |  |  |  |
|  |  |  |  |

#### 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\_TRUE\_CNT\_LGC |
| D\_FALSE\_CNT\_LGC |

### 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,

None

## Data Hiding Functions

None

## Global Functions/Macros Defined by this Module

None

## Local Functions/Macros Used by this MDD only

None

# 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 |
| <None> |  |

## Initialization Functions

## NonePeriodic Functions

### Per: Gsod\_Per1

#### Design Rationale

This module uses direct input reads instead of buffered reads to disable interrupts around data collection of the inputs to ensure that the data set is consistent. The limitation of this method is that it is only possible if inputs are being written from a task at the same or higher priority with the same or faster loop execution time; which is the case at the time of this implementation.

#### Program Flow Start

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

#### Store Module Inputs to Local copies

SuspendAllInterrupts()

Rte\_Read\_Corrected\_MtrPos\_Rev\_f32(&Corrected\_MtrPos\_Rev\_T\_f32)

Rte\_Read\_SysC\_Corrected\_MtrPos\_Rev\_f32(&SysC\_Corrected\_MtrPos\_Rev\_T\_f32)

Rte\_Read\_Ana\_Hw\_Torque\_HwNm\_f32(&Ana\_Hw\_Torque\_HwNm\_T\_f32)

Rte\_Read\_SysC\_Ana\_Hw\_Torque\_HwNm\_f32(&SysC\_Ana\_Hw\_Torque\_HwNm\_T\_f32)

Rte\_Read\_Vecu\_Volt\_f32(&Vecu\_Volt\_T\_f32)

Rte\_Read\_SysC\_Vecu\_Volt\_f32(&SysC\_Vecu\_Volt\_T\_f32)

Rte\_Read\_Torque\_Cmd\_CRF\_MtrNm\_f32(&Torque\_Cmd\_CRF\_MtrNm\_T\_f32)

Rte\_Read\_SysC\_Torque\_Cmd\_CRF\_MtrNm\_f32(&SysC\_Torque\_Cmd\_CRF\_MtrNm\_T\_f32)

Rte\_Read\_Torque\_Cmd\_MRF\_MtrNm\_f32(&Torque\_Cmd\_MRF\_MtrNm\_T\_f32)

Rte\_Read\_SysC\_Torque\_Cmd\_MRF\_MtrNm\_f32(&SysC\_Torque\_Cmd\_MRF\_MtrNm\_T\_f32)

Rte\_Read\_Cum\_Mtr\_Pos\_CRF\_Deg\_f32(&Cum\_Mtr\_Pos\_CRF\_Deg\_T\_f32)

Rte\_Read\_SysC\_Cum\_Mtr\_Pos\_CRF\_Deg\_f32(&SysC\_Cum\_Mtr\_Pos\_CRF\_Deg\_T\_f32)

Rte\_Read\_MtrElecMech\_Polarity\_Cnt\_s08(&MtrElecMech\_Polarity\_Cnt\_T\_s08)

Rte\_Read\_SysC\_MtrElecMech\_Polarity\_Cnt\_s32(&SysC\_MtrElecMech\_Polarity\_Cnt\_T\_s32)

ResumeAllInterrupts()

#### Processing





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

None

#### Program Flow End

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

## Fault Recovery Functions

None

## Shutdown Functions

None

## Interrupt Functions

None

## Serial Communication Functions

None

# Execution Requirements

## Execution Sequence of the Module

Gsod\_per1 must execute at the **end** of the 2ms loop after all other modules using global inputs are called. This is to verify the integrity of the global input signals after all cases that use them have completed.

## 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 |
| Gsod\_Per1 | 2 ms | WARM INIT, OPERATE, DISABLE |

## Execution Requirements for Serial Communication Functions

|  |  |
| --- | --- |
| Function Name | Sub-Module called by (Serial Comm Function Name) |
| <None> |  |

# 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 |
| Gsod\_Per1 | RTE\_START\_SEC\_AP\_GSOD\_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. None

# Revision Control Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Rev #** | **Change Description** | **Date** | **Author Initials** |
| 1 | 1 | Initial Component Creation | 22-Oct-12 | JWJ |
| 2 | 2 | Corrected incompatible input signal units | 14-Nov-12 | JWJ |
| 3 | 3 | Removed typecasts from flowcharts | 21-Mar-13 | LWW |
| 4 | 4 | Update per anomaly 5035 and FDD SF37 v002 | 21-May-13 | BDO |