# Module -- LrnEOT

# High-Level Description

LrnEOT uses vehicle operational information to learn the appropriate end of travel positions for a given system.

# Figures

## Component Diagram



# Module Inputs and Outputs

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 (Global Variable Name) | Module Outputs (Global Variable Name) |
| MtrVelCRF\_MtrRadpS\_f32 | CWPosition\_HwDeg\_f32 |
| HandwheelPosition\_HwDeg\_f32 | CCWPosition\_HwDeg\_f32 |
| HandwheelAuthority\_Uls\_f32 | CWFound\_Cnt\_lgc |
| HwTorque\_HwNm\_f32 | CCWFound\_Cnt\_lgc |
| PostLimitTorque\_HwNm\_f32 |  |
|  |  |
| DiagStsHwPosDis\_Cnt\_lgc |  |
|  |  |
|  |  |

## 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 |
| CcwEOTTimer\_mS\_M\_u32 | 1 | 0 | Full | LRNEOT\_START\_SEC\_VAR\_32 |
| CwEOTTimer\_mS\_M\_u32 | 1 | 0 | Full | LRNEOT\_START\_SEC\_VAR\_32 |
| Position1Timer\_mS\_M\_u32 | 1 | FULL | FULL | LRNEOT\_START\_SEC\_VAR\_32 |
| Position2Timer\_mS\_M\_u32 | 1 | FULL | FULL | LRNEOT\_START\_SEC\_VAR\_32 |
|  |  |  |  |  |
| ResetLimitReq\_Cnt\_M\_lgc | N/A | N/A | N/A | LRNEOT\_START\_SEC\_VAR\_ BOOLEAN |

### User defined typedef definition/declaration

This section documents any user types uniquely used for the module.

|  |  |  |
| --- | --- | --- |
| Typedef Name | Element Name | Storage Type |
|  |  |  |

# 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\_MinRackTrvl\_HwDeg\_f32 |
| k\_MaxRackTrvl\_HwDeg\_f32 |
| k\_AuthorityStartLrn\_Uls\_f32 |
| k\_HwTrqEOTLrn\_HwNm\_f32 |
| k\_MtrVelEOTLrn\_MtrRadpS\_f32 |
| k\_EOTLrnTimer\_mS\_u16 |
| k\_MtrTrqEOTLrn\_MtrNm\_f32 |
|  |
| k\_MinResetAuthority\_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 | 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\_ZERO\_ULS\_F32 |
| D\_ZERO\_CNT\_U16 |

### Module specific Lookup Tables Constants

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

# Functions/Macros used by the Sub-Modules

## Library Functions / Macros

The library functions / Macros that are called by the various sub modules are identified below,

1. Abs\_f32\_m()
2. Max\_m()
3. Min\_m()

## Data Hiding Functions

The data hiding functions / macros used in this module are identified below,

1. Rte\_Call\_LearnedEOTData\_SetRamBlockStatus()
2. Rte\_Call\_LearnedEOTData\_WriteBlock()

## Local Functions/Macros Used by this MDD only

The local functions/macros in this module are identified below:

1. ResetEOT()

# Software Module Implementation

## Initialization Functions

### Init: LrnEOT\_Init()

#### Design Rationale

None

#### Initialize End-of-Travel



#### Module Outputs

None

#### Module Internal

Rte\_IWrite\_LrnEOT\_Init1\_CCWFound\_Cnt\_lgc(Rte\_Pim\_LearnedEOT()->CCWEOTFound\_Cnt\_lgc)

Rte\_IWrite\_LrnEOT\_Init1­\_CCWPosition\_HwDeg\_f32(Rte\_Pim\_LearnedEOT()->CCWEOTPosition\_HwDeg\_f32)

Rte\_IWrite\_LrnEOT\_Init1\_CWFound\_Cnt\_lgc(Rte\_Pim\_LearnedEOT()->CWEOTFound\_Cnt\_lgc)

Rte\_IWrite\_LrnEOT\_Init1\_CWPosition\_HwDeg\_f32(Rte\_Pim\_LearnedEOT()->CWEOTPosition\_HwDeg\_f32)

## Periodic Functions

### Per: LrnEOT\_Per1

#### Design Rationale

None

#### Program Flow Start

Rte\_Call\_LrnEOT\_Per1\_CP0\_CheckpointReached

#### Store Module Inputs to Local copies

Local Variables:

*DiagStsHwPosDis\_Cnt\_T\_lgc* = Rte\_IRead\_LrnEOT\_Per1\_DiagStsHwPosDis\_Cnt\_lgc()

*HandwheelAuthority\_Uls\_T\_f32* = Rte\_IRead\_LrnEOT\_Per1\_HandwheelAuthority\_Uls\_f32()

*HandwheelPosition\_HwDeg\_T\_f32* = Rte\_IRead\_LrnEOT\_Per1\_HandwheelPosition\_HwDeg\_f32()

*HwTorque\_MtrNm\_T\_f32* = Rte\_IRead\_LrnEOT\_Per1\_HwTorque\_HwNm\_f32()

*MtrVel\_MtrRadpS\_T\_f32* = Rte\_IRead\_LrnEOT\_Per1\_MtrVelCRF\_MtrRadpS\_f32()

*PostLimitTorque\_MtrNm\_T\_f32* = Rte\_IRead\_LrnEOT\_Per1\_PostLimitTorque\_MtrNm\_f32()

#### Reset EOT Limits



#### Learn End of Travel Limits



#### EOT Learn Complete Indication



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

PrevLrnPnCntrState\_Cnt\_M\_enum = LrnPnCntrState\_Cnt\_T\_enum

Rte\_IWrite\_LrnEOT\_Per1\_CCWFound\_Cnt\_lgc(Rte\_Pim\_LearnedEOT()->CCWEOTFound\_Cnt\_lgc)

Rte\_IWrite\_LrnEOT\_Per1\_CCWPosition\_HwDeg\_f32(Rte\_Pim\_LearnedEOT()->CCWEOTPosition\_HwDeg\_f32)

Rte\_IWrite\_LrnEOT\_Per1\_CWFound\_Cnt\_lgc(Rte\_Pim\_LearnedEOT()->CWEOTFound\_Cnt\_lgc)

Rte\_IWrite\_LrnEOT\_Per1\_CWPosition\_HwDeg\_f32(Rte\_Pim\_LearnedEOT**()->**CWEOTPosition\_HwDeg\_f32)

#### Program Flow End

Rte\_Call\_LrnEOT\_Per1\_CP1\_CheckpointReached

## Fault Recovery Functions

None

## Shutdown Functions

None

## Interrupt Functions

None

## Serial Communication Functions

### LrnEOT\_Scom\_ResetEOT

ResetLimitReq\_Cnt\_M\_lgc = True

## Local Function/Macro Definitions

### Reset End of Travel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | ResetEOT | Type | Min | Max |
| **Arguments Passed** |  |  |  |  |
| **Return Value** |  |  |  |  |

#### Description



# Execution Requirements

## 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 |
| LrnEOT\_Init () | On Event | On Init |
| LrnEOT\_Per1() | 10mS | ALL |

## Execution Requirements for Serial Communication Functions

|  |  |
| --- | --- |
| Function Name | Sub-Module called by (Serial Comm Function Name) |
| LrnEOT\_Scom\_ResetEOT |  |
|  |  |

# 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 |
| LrnEOT\_Per1 | RTE\_START\_SEC\_AP\_LRNEOT\_APPL\_CODE |
| LrnEOT\_Scom\_ResetEOT | RTE\_START\_SEC\_AP\_LRNEOT\_APPL\_CODE |
|  |  |

## Local Functions

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

|  |  |
| --- | --- |
| Name of Sub Module | Software Segment |
| ResetEOT |  |

# 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 | 1.0 | Initial release | 13Feb12 | M. Story |
| 2 | 2.0 | Updated ResetEOT function to use WriteBlock API, fixed anomaly 3206 | 26-Apr-12 | VK |
| 3 | 3 | Correction of anomaly 3259, QAC updates | 29-May-12 | LWW |
| 4 | 4 | Updated to SF-11 v002 | 20-Jun-12 | OT |
| 5 | 5 | UTP Fix – calibration name | 27-Jun-12 | OT |
| 6 | 6 | Moved RackCentering function to new Ap\_LnRkCr component – FDD SF-39 | 18-Aug-12 | BDO |
| 7 | 7 | Check points flow updated | 23- Sep -12 | Selva |
| 8 | 8 | Changed Per1 trigger rate from 4ms to 10ms. | 22-Oct-12 | BWL |