# Module – Tuning Select Authority

# High-Level Description

This function broadcasts an authority to allow switching between calibration subsets while driving. It compares handwheel torque and vehicle speed to calibratable thresholds and outputs either a zero or a one.

# Figures

## Component Diagram

TunSel.tif

# 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 | |
| HwTorque\_HwNm\_f32 | | ActiveTunPers\_Cnt\_u16 |
| VehicleSpeed\_Kph\_f32 | | ActiveTunSet\_Cnt\_u16 |
| DesiredTunPers\_Cnt\_u16 | | TunPer\_Ptr\_Str \* |
| DesiredTunSet\_Cnt\_u16 | | TunSet\_Ptr\_Str \* |

*\* Note: For unit testing purposes these inputs are defined as pointers to uint16 (as opposed to pointers to tuning structures) for simplicity.*

## 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 |
| \*HwTrqLPFiltSV\_HwNm\_M\_str | Multiple | Multiple | Multiple | TUNINGSELAUTH\_START\_SEC\_VAR\_CLEARED\_UNSPECIFIED |
| * K\_Uls\_f32 | Single Precision Float | 0.0124877435 | 0.4665119090 |  |
| * SV\_HwNm\_f32 | Single Precision Float | -10 | 10 |  |
| PrevTunSet\_Cnt\_M\_u16 | 1 | 0 | 100 | TUNINGSELAUTH\_START\_SEC\_VAR\_CLEARED \_16 |
| PrevTunPers\_Cnt\_M\_u16 | 1 | 0 | 100 | TUNINGSELAUTH\_START\_SEC\_VAR\_CLEARED \_16 |

### 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 |
| k\_TunSelHwTrqThresh\_HwNm\_f32 |
| k\_TunSelVehSpdThresh\_Kph\_f32 |
| k\_TunSelHwTrqLPFKn\_Hz\_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\_10MS\_SEC\_F32 | Single Precision Float | Sec | 0.010 |

#### 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 |
| T\_TunSetSelectionTbl\_Ptr\_Str[] \* |
| T\_TunPersSelectionTbl\_Ptr\_Str[][] \* |

*\* Note: For unit testing purposes, these arrays of pointers are of size [3] and [3][5] respectively and are defined as pointers to uint16 (as opposed to pointers to tuning structures) for simplicity.*

### Module specific Lookup Tables Constants

|  |  |  |  |
| --- | --- | --- | --- |
| 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,

1. Abs\_f32\_m
2. LPF\_OpUpdate\_f32\_m
3. LPF\_KUpdate\_f32\_m

## Data Hiding Functions

1. <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 |
| HwTorque\_HwNm\_f32 | 0 |
| VehicleSpeed\_Kph\_f32 | 0 |
| DesiredTunPers\_Cnt\_u16 | 0 |
| DesiredTunSet\_Cnt\_u16 | 0 |
| ActiveTunPers\_Cnt\_u16 | 0 |
| ActiveTunSet\_Cnt\_u16 | 0 |

## Initialization Functions

### Init: TuningSelAuth\_Init1

#### Design Rationale

LPF\_KUpdate\_f32 is used to initialize the LPF filter instead of the full LPF\_Init\_f32 macro as an optimization since the required initial state of the filter is 0, which is the initialized value of the RAM, so there is no need to explicitly initialize the state variables in this init function.

#### Initialize Low Pass Filters

#### 

### Periodic Functions

#### Per: TuningSelAuth\_Per1

#### Design Rationale

#### Program Flow Start

#### Rte\_Call\_TuningSelAuth\_Per1\_CP0\_CheckpointReached() Store Module Inputs to Local copies

See below section

#### Description



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

See above section

#### Program Flow End

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

## Fault Recovery Functions

None

## Shutdown Functions

None

## Interrupt Functions

None

## Serial Communication Function

None

## Transition Functions

None

# Execution Requirements

## Execution Sequence of the Module

If something besides the defaults of “0” for desired tuning set and desired personality are required at poweron, Init1 must run after the init function that provides these initial values.

## 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 |
| TuningSelAuth\_Init1 | Once at Startup | COLDINIT |
| TuningSelAuth\_Per1 | 10 ms | ALL |

## 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 |
| TuningSelAuth \_Init1 | RTE\_START\_SEC\_AP\_TUNINGSELAUTH\_APPL\_CODE |
| TuningSelAuth\_Per1 | RTE\_START\_SEC\_AP\_TUNINGSELAUTH\_APPL\_CODE |

## Local Functions

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

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

# Known Issues / Limitations With Design

1. INLINE functions defined in “GlobalMacro.h” are not unit tested
2. The FDD indicates that this module will store the EEPROM value for tuning set, however, this implementation doesn’t provide this block. Instead, it is assumed that some other module will contain the tuning selection block. This was done to enable programs that don’t have multiple tuning sets to just use the default “0” without having to manage an EEPROM block.

# Revision Control Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Rev #** | **Change Description** | **Date** | **Author Initials** |
| 1 | 1.0 | Initial MDD implementing FDD SF-23 v001 | 03Jul12 | VK |
| 1 | 2.0 | Updates to provide the switching of tuning sets and personalities | 03/08/12 | LWW |
| 2 | 3.0 | Added checkpoints and memmap software segment is updated for static variables | 24-Sep-12 | Selva |
| 3 | 4.0 | Updated trigger rate for Per1 | 24-Oct-12 | BWL |