**TDD for Tachometer and Doppler failure.**

Contents

[General: 1](#_Toc32320810)

[Requirements 1](#_Toc32320811)

[Summary of changes in DMI-ATP Interface spec 2](#_Toc32320812)

[ATP- DMI: 2](#_Toc32320813)

[DMI-ATP: 2](#_Toc32320814)

[IF ATP-DMI changes: 2](#_Toc32320815)

[ATP Changes 2](#_Toc32320816)

[Sequence of changes in the action: 3](#_Toc32320817)

[Log Events: 4](#_Toc32320818)

[Changes in the DMI Handler 4](#_Toc32320819)

[DMI changes 4](#_Toc32320820)

[DMI Indications: 4](#_Toc32320821)

[New DMI forms 4](#_Toc32320822)

# General:

In COD, On Tachometer and Doppler failure shall be

* Indicated on the DMI as long as the failure is reported
* Confirmed by the driver.
* Log the events to the DMI and R

# Requirements

AOS\_AS-394 S

The ATP shall supervise the Q\_CONTROL variable while, and only while, the variable Q\_ODOSAFE indicates Full Service.

If Q\_CONTROL indicates that an Odometer hardware unit has failed the ATP shall issue a log event and inform the driver about speed sensor failure, regardless which hardware unit that was reported as failed.

A **confirmation from the driver is required which shall be registered in the Recording Unit**.   
The **failure indication shall remain until the Odometer no longer reports the failur**e.

On Tachometer OR Doppler radar failure, Driver is indicated with on DMI.   
Driver shall select the DMIButton to confirm the unavailability.

According to GSP-2 Odometer IF, an indication will remain until next system restart.

# Summary of changes in DMI-ATP Interface spec

## ATP- DMI:

1. **Tachometer1IsFailed** Indication.
2. **Tachometer2IsFailed** Indication.
3. **DopplerRadarIsFailed** Indication.
4. For Driver to confirm Tachometer1 is Failed, ATP sends bit to show **FormConfirmTacho1Fail**.
5. For Driver to confirm Tachometer2 is Failed , ATP sends bit to show **FormConfirmTacho2Fail**.
6. For Driver to confirm Doppler is Failed , ATP sends bit to show **FormConfirmDopplerFail**.

## DMI-ATP:

1. Driver confirms Tachometer fail with additional bit **DMIButtonConfirmTacho1Fail.**
2. Driver confirms Tachometer fail with additional bit **DMIButtonConfirmTacho2Fail**.
3. Driver confirms Doppler Radar fail with additional bit **DMIButtonConfirmDopplerRadarFail**.

## IF ATP-DMI changes:

**ATP-DMI:**

New Field: Platform Status  
Data 24 bit 0 ($1) – Tachometer1 failure Data 24 bit 1 ($2) – Tachometer2 failure Data 24 bit 2 ($4) – Doppler Radar failure

New Field: Additional Status confirmation bits  
Data 23 bit 0($1) – Confirm «Tachometer1 is failure«.  
Data 23 bit 1($2) – Confirm «Tachometer2 is failure«.  
Data 23 bit 2(S4) – Confirm «Doppler Radar is failure«.

**DMI-ATP:**

Message Type 34 ,   
Data 0 bit 39 – DMIButton Confirm “Tachometer1 failure”  
Data 0 bit 40 – DMIButton Confirm “Tachometer2 failure”  
Data 0 bit 41 – DMIButton Confirm “Doppler Radar failure”

# ATP Changes

In abstract\_odometry, ATP supervises Q\_CONTROL in Odometry component. In method readMeasurementTelegram(), we read the measurement data from COD through data channel.

Currently, we check for Q\_CONTROL and odoMeasDataQControlSpeedSensorBitmask to log the sensor failure.

To check availability of Tachometer and Doppler Radar, primarily we check for **Q\_ODOSAFE** to be at full Service. Then the availability of Tachometer and doppler Radar are based on Q\_CONTROL bits received through the data telegram.

We check for the following reserved bits to check the availability of the Tachometer1  
Bit0 – Tachometer1 in error  
Bit8 – Connection lost to Tachometer1  
Bit12 – Tachometer 1 in permanent error.

We check for the following reserved bits to check the availability of the Tachometer2  
Bit1 – Tachometer2 in error  
Bit9 – Connection lost to Tachometer2  
Bit13 – Tachometer2 in permanent error.

We check for the following reserved bits to check the availability of the Doppler Radar

Bit 2 – Doppler Radar in error

Bit 3 – Doppler Radar in maintenance   
Bit 10 – Connection lost to Doppler

Bit 14 – Doppler Radar in permanent error

(Maintenance bit will be set when the Doppler radar has been invalid for more than 10% during one hour).

We shall check for these bits in Odometer to set the values on the availability of the Tachometer and Doppler Radar.

We got new Boolean flags Tachometer1Fail, Tachometer2Fail and dopplerRadarFail.

In abstract\_odometry ,we shall evaluate the availability of Tachometer and DopplerRadar by   
void AbstractOdometry::evaluateOdometerSensorFailure(uint16\_t qControl).

This sets the flags when Tachometer1 , Tachometer2 or Doppler Radar is failed.

These flags are accessed by getter functions ->

getTachometer1Failure()

getTachometer2Failure()

getDopplerRadarFailure() to get these values in DMIMessageOutATPModesAndStatus.

In evaluateOdometerSensorFailure(), we trigger the Log events if any of the components are failed.

We log events to RU and DMI on Error, Connection lost OR permanent error failure of Tachometer1, Tachometer2 and Doppler Radar.

In dmi\_message\_out\_atp\_modes\_and\_states, collectData(), we get the availability of Tachometer and DopplerRadar by access functions.

**For DMI Indication on failure:**

Indication bits are sent to DMI by checking the Availability values of the Tachometer and Doppler Radar by access functions getTachometer1Failure() , getTachometer1Failure() and getDopplerRadarFailure().

**For the Driver confirmation:**

From the access functions, we receive the status of the Tachometer.

We read the tachometer/doppler status and save the status flags in the DMI Handler.

The confirmed flag shall be cleared when failure changed from false to true.

The confirm form shall be shown as long as failure && !Confirmed.

The confirmed flag shall be cleared only when the driver confirms.

We check the change in Tachometer Status, If the tachostatus is changed, we show the **FormConfirmTachometer1Failure**

In DMI\_Handler, we set the flag if the tachometer is failed, save it to show **FormConfirmTachometer1Failure.** Clear it **by** DMIButtonConfirmTachometer1Failure.

**FormConfirmTachometer1 Fail =** Failure && !Confirmed

## Sequence of changes in the action:

There are two sequences happening in parallel.

1. A Confirmation from Driver shall be registered to the RU.
2. Constant Indication of failure until Odometer no longer reports failure.

For Confirmation:  
1. Tachometer fails.  
2. ATP sends the bit **FormConfirmTachometer1Failure** to DMI, for driver to confirm the Tachometer failure.  
3. Form to Confirm Tachometer failure is shown to driver , Form remains until driver selects.  
4. Driver selects **DMIButtonConfirmTachometer1Fail**  to confirm the Tachometer failure.

ATP clears the bit **FormConfirmTachometer1Fail** and still indicates **Tachometer1IsFailed** until the next restart OR tachometer no longer reports failure.

For Indication:1. Tachometer fails.  
2. ATP detects and sends the indication bit **Tachometer1IsFailed .  
3.** DMI indicates **DMIIndicationTachometer1Fail** indication. This indication is shown on DMI as long as the tachometer no longer reports failure or until next restart.

## Log Events:

RU:  
- Log event when any of component in Odometry

- Log Events of Tachometer 1, Tachometer2 and Doppler Radar failure.  
Each event for Error, Connection lost and permanent error failure of component.

DMI:

* Log DMIEvents of Error, Connection Lost and Permanent error failure of  
  Tachometer1, Tachometer2 and Doppler Radar failures.

## Changes in the DMI Handler

New DMIButton Types:

DMIButtonConfirmTachometer1Failure = 39U  
DMIButtonConfirmTachometer1Failure = 40U  
DMIButtonConfirmDopplerRadarFailure = 41U

# DMI changes

## DMI Indications:

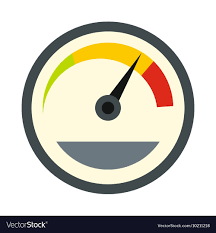
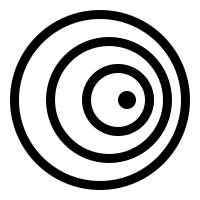
1. **DMIIndicationTachometerFail** Indication
2. **DMIIndicationDopplerRadarFail** Indication

## New DMI forms

1. FormConfirmTachometer1Failure
2. FormConfirmTachometer2Failure
3. FormConfirmDopplerRadarFailure

DMI Icons to indicate Doppler and Tacho:

These will be converted to 16bit before adding them to DMI resources.



DOPPLER TACHOMETER

Questions

1. Should the Driver confirm Tachometer failure OR Tachometer1 Failure?
2. Do we send Tacho1 and Tacho2 bits to DMI or Tachometer in General and log Events specific to each of Tcaho1 and Tacho2.