

Approaching Vehicle Audible System Module

(AVAS)

FUNCTIONAL SPECIFICATION

FS\_ML3T-14G113-AB

Version: 2.0 RELEASE

Date Revised: 07/30/2019

Author: Peter Sripinyo

**FORD MOTOR COMPANY CONFIDENTIAL AND PROPRIETARY**

This document contains Ford Motor Company confidential and proprietary information. Disclosure of the information contained in any portion of this document is not permitted without the expressed, written consent of a duly authorized representative of Ford Motor Company, Dearborn, Michigan, USA.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | Functional Specification Approaching Vehicle Audible System Module | | | | | | | | | | |
| PART NAME | | | | | | | | | | PART NUMBER | | | | | | |
| Functional Specification – Module Assembly – Approaching Vehicle Audible System | | | | | | | | | | FS-ML3T-14G113-AB | | | | | | |
|  | | LET | | FR | | REVISIONS | | | | | | DR | CK | REFERENCE | | |
| 08/08/2018 | | REL | | All | | FS-ML3T-14G113-AA released | | | | | |  |  |  | | |
| 07/30/2019 | | REL | | All | | FS-ML3T-14G113-AB Released | | | | | |  |  | PREPARED BY | | |
|  | |  | |  | |  | | | | | |  |  | Peter Sripinyo | | |
|  | |  | |  | |  | | | | | |  |  | APPROVED BY | | DETAILED BY |
|  | |  | |  | |  | | | | | |  |  | PS | | PS |
|  | |  | |  | |  | | | | | |  |  | CONCURRENCE/APPROVAL | | |
|  | |  | |  | |  | | | | | |  |  | SIGNATURES | | |
|  | |  | |  | |  | | | | | |  |  | Design Engineering Supervisor | | |
|  | |  | |  | |  | | | | | |  |  | John Ricks | | |
|  | |  | |  | |  | | | | | |  |  | Design Engineering Management | | |
|  | |  | |  | |  | | | | | |  |  | Jennifer Shaw | | |
|  | |  | |  | |  | | | | | |  |  | Manufacturing Engrg. | | |
|  | |  | |  | |  | | | | | |  |  |  | | |
|  | |  | |  | |  | | | | | |  |  | Quality Control | | |
|  | |  | |  | |  | | | | | |  |  |  | | |
|  | |  | |  | |  | | | | | |  |  | Purchasing | | |
|  | |  | |  | |  | | | | | |  |  |  | | |
|  | |  | |  | |  | | | | | |  |  | Supplier Quality Assistance | | |
|  | |  | |  | |  | | | | | |  |  |  | | |
|  | |  | |  | |  | | | | | |  |  |  | | |
|  | |  | |  | |  | | | | | |  |  |  | | |
|  | |  | |  | |  | | | | | |  |  |  | | |
|  | |  | |  | |  | | | | | |  |  |  | | |
| FRAME | 1 | | | | OF | | 131 | | REV REL | |  | | | | | |
| PD  May 1988 | | | **3947a1e** | | | | | (Previous editions may **NOT** be used) | | | | | | |

# Content

[**FORD MOTOR COMPANY CONFIDENTIAL AND PROPRIETARY** 0](#_Toc16151726)

[Content 2](#_Toc16151727)

[1. DOCUMENT OVERVIEW 5](#_Toc16151728)

[1.1 DOCUMENT OVERVIEW 5](#_Toc16151729)

[1.2 Purpose and Scope 5](#_Toc16151730)

[1.2.1 Conflict of Documentation 5](#_Toc16151731)

[1.3 DOCUMENT ROAD MAP 5](#_Toc16151732)

[1.4 DOCUMENT CONVENTIONS 6](#_Toc16151733)

[1.4.1 Methodology & Data Flow Diagrams 6](#_Toc16151734)

[1.4.2 Requirements Representations 6](#_Toc16151735)

[1.4.3 State Transition Table/Diagram Notation 7](#_Toc16151736)

[1.4.3.1 Feature Functional Requirements 10](#_Toc16151737)

[1.4.4 Glossary of Terms 11](#_Toc16151738)

[1.4.5 Assumed Interface to NVRAM Manager 13](#_Toc16151739)

[1.4.6 Timer Mark Event / Time Since Event 13](#_Toc16151740)

[1.5 RELATED DOCUMENTS 16](#_Toc16151741)

[1.5.1 PARSED 18](#_Toc16151742)

[1.5.2 Cybersecurity 18](#_Toc16151743)

[1.6 General Requirements 19](#_Toc16151744)

[1.6.1 Module - Memory / Power Up / Microcontroller Reset 19](#_Toc16151745)

[1.6.1.1 Microcontroller memory Storage Classification Requirements: 19](#_Toc16151746)

[1.6.1.2 Non-volatile Memory Requirements: 19](#_Toc16151747)

[1.6.1.3 Module Power Up / Microcontroller Reset Requirements: 20](#_Toc16151748)

[1.6.2 Software Classification Level 20](#_Toc16151749)

[1.6.3 Timing Requirements 21](#_Toc16151750)

[1.6.3.1 Timing / Response Requirements 21](#_Toc16151751)

[1.6.4 Order of Execution 21](#_Toc16151752)

[1.6.4.1 Order of Execution Requirements 21](#_Toc16151753)

[2 APPROACHING VEHICLE AUDIBLE SYSTEM (AVAS) 23](#_Toc16151754)

[2.1 Feature Behavior Summary 23](#_Toc16151755)

[2.2 Subsystem Overview 24](#_Toc16151756)

[2.3 Feature Behavior Detail 25](#_Toc16151757)

[2.4 Determine Transmission Status 26](#_Toc16151758)

[2.5 Determine Vehicle Mode Status 31](#_Toc16151759)

[2.6 Determine Vehicle Speed Status 35](#_Toc16151760)

[2.7 Determine AVAS State 37](#_Toc16151761)

[2.7.1 AVAS Fault State Signal 38](#_Toc16151762)

[2.8 Generate AVAS Output 40](#_Toc16151763)

[2.8.1 General AVAS Output Requirements 40](#_Toc16151764)

[2.8.2 AVAS Required Signal Flow 40](#_Toc16151765)

[2.8.3 Signal Generation Block Requirements 41](#_Toc16151766)

[2.8.4 Filter Block Requirements 43](#_Toc16151767)

[2.8.5 One Speaker Systems versus Multi-Speaker Systems 43](#_Toc16151768)

[2.8.6 AVAS Output Timing Requirements 44](#_Toc16151769)

[2.9 Voltage Range Monitor 45](#_Toc16151770)

[2.9.1 Voltage Range Monitor Feature Behavior Summary 45](#_Toc16151771)

[2.9.2 Basic Voltage Range Requirements 48](#_Toc16151772)

[2.9.3 Voltage Range State Machine 49](#_Toc16151773)

[2.9.4 Voltage Range Monitor Defined Voltage Ranges 53](#_Toc16151774)

[2.9.4.1 Voltage Range Configuration (VR\_080\_160\_VB) 53](#_Toc16151775)

[2.9.4.2 Diagnostic Trouble Code (DTC) Logging (VR\_100\_155\_VB) 54](#_Toc16151776)

[2.9.4.3 MS/HS CAN Interface (VR\_080\_160\_CAN) 55](#_Toc16151777)

[2.9.5 Shutdown Detection Voltage Range (VRange\_SDown) 56](#_Toc16151778)

[2.9.6 Summary of Voltage Range Monitors 57](#_Toc16151779)

[2.10 Diagnostics 58](#_Toc16151780)

[2.10.1 Overview 58](#_Toc16151781)

[2.10.2 Supported Diagnostic Identifiers 58](#_Toc16151782)

[2.10.3 Supported Security Levels 58](#_Toc16151783)

[2.10.3.1 Security Level (0x01) Specific Information 58](#_Toc16151784)

[2.10.3.2 Security Level (0x03) Specific Information 59](#_Toc16151785)

[2.10.4 AVAS Supported Diagnostic Services 59](#_Toc16151786)

[2.10.5 AVAS Diagnostic Session Control (0x10) Service 60](#_Toc16151787)

[2.10.5.1 Supported Diagnostic Session Control 60](#_Toc16151788)

[2.10.5.2 Preconditions for a Diagnostic Session Change (Service 0x10) 61](#_Toc16151789)

[2.10.5.3 Extended Session Exit Conditions 61](#_Toc16151790)

[2.10.5.4 Diagnostic Session Control State Transition Diagram 63](#_Toc16151791)

[2.10.5.5 Diagnostic Timer Monitor 68](#_Toc16151792)

[2.10.6 AVAS Routine Control (0x31) Service 68](#_Toc16151793)

[2.10.6.1 Supported Functional Execution Routines 68](#_Toc16151794)

[2.10.6.2 Type 1 Routines 69](#_Toc16151795)

[2.10.6.3 Type 2 Routines 70](#_Toc16151796)

[2.10.6.4 Type 3 Routines 72](#_Toc16151797)

[2.10.7 AVAS Data Identifiers (DIDs) 72](#_Toc16151798)

[2.10.7.1 ReadDataByIdentifer (0x22) Service 72](#_Toc16151799)

[2.10.7.2 Writeable DIDs (service $2E) 77](#_Toc16151800)

[2.10.7.3 Supplier Range DIDs 78](#_Toc16151801)

[2.10.8 AVAS Diagnostic Trouble Codes (DTCs) 79](#_Toc16151802)

[2.10.8.1 Diagnostic Trouble Codes General Requirements 79](#_Toc16151803)

[2.10.8.2 ControlDTCSetting (0x85) Service 79](#_Toc16151804)

[2.10.8.3 DTCs for Hardware Outputs 80](#_Toc16151805)

[2.10.8.4 Asynchronous Fault Detection 80](#_Toc16151806)

[2.10.8.5 Control SettingAging of DTCs Finite State Machine 81](#_Toc16151807)

[2.10.8.6 DTC Definitions 83](#_Toc16151808)

[2.10.8.7 DTC Definitions and Requirements Table 84](#_Toc16151809)

[2.10.8.8 DTCMaxAction(index) Function 92](#_Toc16151810)

[2.10.8.9 DTCMinAction(index) Function 92](#_Toc16151811)

[2.10.8.10 DTC Suppression 92](#_Toc16151812)

[2.10.8.11 Definitions for DTC Status Bits 95](#_Toc16151813)

[2.10.8.12 Diagnostic Cycle Monitor 95](#_Toc16151814)

[2.10.8.13 ResetDTCCounters() Function 96](#_Toc16151815)

[2.10.8.14 AgeAppropriateDTCs() Function 96](#_Toc16151816)

[2.10.8.15 AgeDTC(index) 96](#_Toc16151817)

[2.10.9 AVAS Configuration Data (Method 2 & 3) 97](#_Toc16151818)

[2.10.9.1 A note about Method 2 data versus Method 3 data 97](#_Toc16151819)

[2.10.9.2 Method 2 Data 97](#_Toc16151820)

[2.10.9.3 Method 2 Configuration Parameters 98](#_Toc16151821)

[2.10.9.4 AVAS Configuration Data (Supplier Range FD00-FEFF) 98](#_Toc16151822)

[2.10.10 Fault conditions for Speaker 101](#_Toc16151823)

[2.10.11 Fault conditions for Lost Communication 101](#_Toc16151824)

[2.10.12 Fault conditions for Internal Failures with AVAS ECU: 102](#_Toc16151825)

[2.10.13 CAN Based Data Identifiers 103](#_Toc16151826)

[2.10.13.1 DID 0x1505 Vehicle Speed - High Resolution 103](#_Toc16151827)

[2.10.13.2 DID 0x7218 Reverse Gear Position 103](#_Toc16151828)

[2.10.13.3 DID 0x0599 PowerPack State 104](#_Toc16151829)

[2.10.13.4 DID 0x40B5 Ignition Position Final Status 104](#_Toc16151830)

[2.10.13.5 DID 0x0130 Transmission Shift Lever Position 105](#_Toc16151831)

[2.10.14 Bootloader 105](#_Toc16151832)

[2.10.14.1 Single Key Storage 108](#_Toc16151833)

[2.10.14.2 Two Key Storage 108](#_Toc16151834)

[2.11 Factory Mode and Transport Mode 109](#_Toc16151835)

[2.12 Data Dictionary 109](#_Toc16151836)

[2.13 Revision History 139](#_Toc16151837)

[2.13.1 Version 2.0 (Author: Peter Sripinyo 7/30/2019) FS-ML3T-14G113-AB 139](#_Toc16151838)

# DOCUMENT OVERVIEW

## DOCUMENT OVERVIEW

Approaching Vehicle Audible System Module Author Team

|  |  |
| --- | --- |
| SW D&R | Core D&R |
| Peter Sripinyo | Dynora Martinez Carranza |
|  |  |

## Purpose and Scope

This document provides detailed requirement descriptions of Approaching Vehicle Audible System (AVAS) Module subsystem.

Although the Hatley-Pirbhai System Specification method was used to generate the contents herein, it has been tailored substantially for conformance to the generic systems engineering process and specification formats. This format is intended to provide the reader a well-organized structure to ease the understanding of the functionality allocated to this subsystem and at the same time provide a modular set of specification elements for reuse and/or re-allocation.

The requirements detailed in this functional specification meet the applicable MPLELC SDS requirements as stated in the SOW. See section 7 for compliance traceability for the feature description and MPLELC SDS.

### Conflict of Documentation

This specification shall follow all government regulations. In case of conflict between specifications the government requirements should take precedence.

If any conflict of documentation, the released Part Drawing shall take precedence over the Component Specification, which shall take precedence over this Functional Specification, which shall take precedence over the System Specification.

## DOCUMENT ROAD MAP

|  |  |
| --- | --- |
| **Section 1.0:** | This section provides the scope and purpose of the features in the AVAS control module. It also contains general requirements for the features in the AVAS Module and references to related documents. |
| **Section 2.0** | This section provides the detailed specification of all the features for the AVAS control module. The subsections in Section 2.0 detail an overview of the AVAS module as well as the specific requirements on how AVAS determines whether or not to play a sound and whether or not the AVAS module is an operational or faulted state. This section also covers VRM and diagnostics. |

## DOCUMENT CONVENTIONS

### Methodology & Data Flow Diagrams

The requirements in this specification are partitioned into processes with data flowing between them. This partitioning is represented in the data flow diagrams. A bubble represents each process. Arrows represent data flows with the direction indicating the direction of the flow of information.

The context diagram shown below is presented here to illustrate the methodology and conventions used in the requirements modeling.

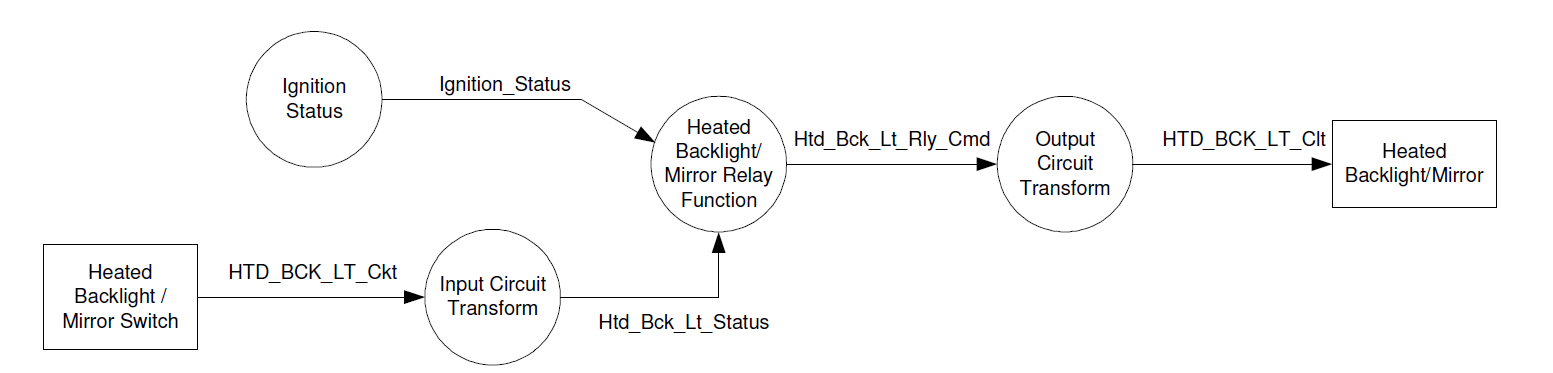


Figure ‑ Context Diagram Illustration

The Hatley-Pirbhai method is used exclusively. Control flows, CFDs, and CSPECS are NOT used. All modeling is done via data flows and DFDs. *unless otherwise specified, it is assumed that all processes are activated when the module awakes and are deactivated when the module goes to sleep*.

When the use of an event would be highly desirable, the ”Ev” suffix is appended to a data flow.

All context diagrams in***2 APPROACHING VEHICLE AUDIBLE SYSTEM (AVAS)***include terminators for all data flows. Terminators can be processes or physical devices connected to the module. If the terminator is a process, such as *IGNITION STATUS,* this means that the reader must go to *the subsection for the process* to actually see the requirements for the Ignition Status Process.

### Requirements Representations

Within each process, the required functionality is described in the form of text, decision tables, state transition diagrams, and/or state transition tables.

Decision tables show combinational logic where Inputs are on the left side of the table and Outputs are on the right side of the table. A double vertical line separates inputs and Outputs. Each row has a unique number to allow reference to that particular requirement.

State transition diagrams and state transition tables contain four key elements: states, transitions, events, and actions. States represent a known condition with the model. Transitions represent the interaction of the states. Events represent the conditions, which must be true for a transition to be taken. Actions represent the operations that must be accomplished when a transition is taken.

State transition diagrams use the following conventions: States are represented by rectangles. Arrows represent transitions.

The symbols\_, –>, and => mean "transitions to" and represent an event occurring at a specific point in time. For example, Ignition\_Status \_ RUN means that the ignition switch has transitioned to the RUN position. This is different than Ignition\_Status = RUN, which means that the ignition switch is in the RUN position. The events and actions for a transition are in text with the events listed before a "/" and the actions following the "/". Timers in one State Transition Diagram are independent of timers in other State Transition Diagrams.

State transition tables have the Current State, Events, Actions, Next State column format with one transition per row. Each row has a unique number to allow reference to that particular requirement. Processing order shall be such that all inputs have been processed prior to evaluation of a given p-spec.

Requirements that are in purple bolded italics and highlighted in blue are "protect for" requirements. Requirements marked as such identify features that are 1) implemented in the FS, but not coded, 2) are not testable or 3) mark summary type information that needs to be searched easily. An example of such markings is ***R:2.4.2.12.2***

### State Transition Table/Diagram Notation

In state machines, some special symbols are used. All symbols used in state machines are listed below. The equality symbol (=) is explained because it is used both for comparisons and assignments.

|  |  |  |
| --- | --- | --- |
| **Symbol** | **Event or  Action** | **Definition** |
| <n> | Event | Requirement number <n>: uniquely identifies requirement #1 when transitioning between state A and state B. is assigned a unique requirement number. Example: <1> |
| = | Event | Equality: |
| <> | Event | Inequality: |
| >= | Event | Greater than or equal: |
| <= | Event | Less than or equal: |
| > | Event | Greater than: |
| < | Event | Less than: |
| -> 🡪 | Event | Transitions to: activates only on the transition from one value to the target value. Unless specified otherwise in the Finite State Machine, the state machine must look for the data transition to occur while it is in the state (or super state) that has the -> as an exit condition. In modeling terms, this means that the transition flag is cleared upon entry to the state (or super state). Special care must be taken when the -> event must be evaluated as part of a logical AND operation. . |
| & | Event | Boolean “AND”: |
| | | Event | Boolean “OR”: |
| = | Action | Assignment: |
| no event | Event | No event trigger |
| no action | Action | No action taken: |
| Mark event xyzzy | Action | Event in time: conceptual timing requirement – this action marks the event “xyzzy” on an imaginary timeline. Later referenced by Time since event |
| Time since event xyzzy | Event | Elapsed time: determines the amount of time that has elapsed since the last occurrence of the Mark event xyzzy |

Table ‑ Special Symbols used in Finite State Machines

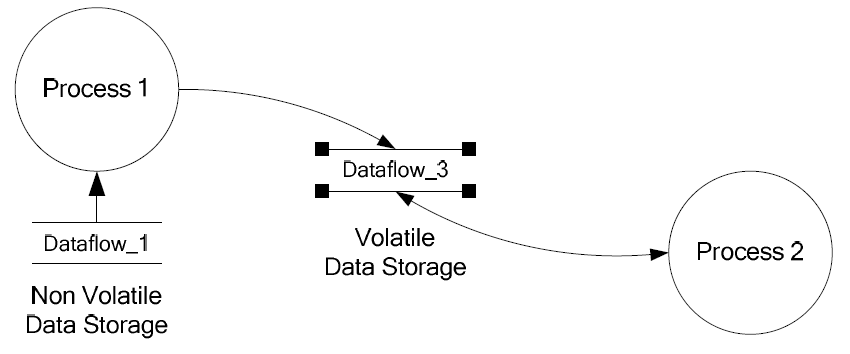


Figure ‑ Example of Detecting a Transition

**Feature Behavior Summary**

To show by representation the difference between volatile memory and non-volatile memory data storage symbols. The non-volatile memory data storage symbol representation is 2 horizontal lines, one line above and one line below the non-volatile dataflow name. The volatile memory data storage is represented by 2 horizontal bars with solid / filled boxes at the end of each line, one line above and one line below the volatile dataflow name.

**Feature Behavior Detail**



**Figure 1.4‑3 Convention Context Diagram**

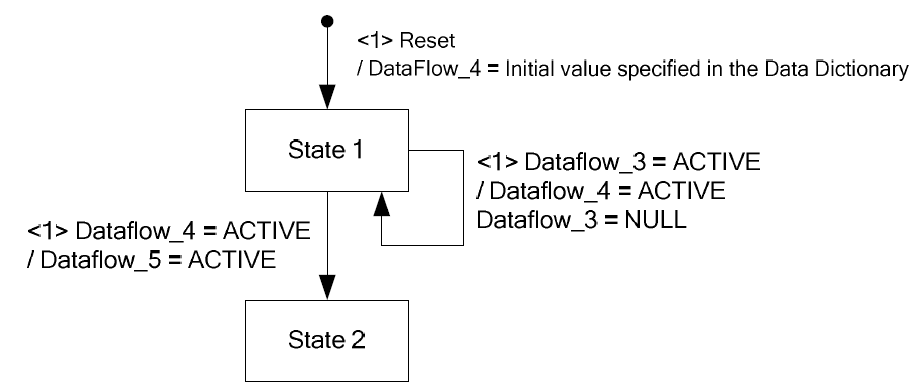
#### Feature Functional Requirements

The following state transition diagram defines the core processing for the feature.

|  |  |
| --- | --- |
| R: . | At Reset, Dataflow\_3 must be set to the initial value specified in the Data Dictionary. |

| **Rqmt. No.** | **Dataflow\_1** | **Dataflow\_2** |  | **Dataflow\_3** |
| --- | --- | --- | --- | --- |
| **R: 1.4.3.1.2** | INACTIVE | INACTIVE |  | No Change |
| **R: 1.4.3.1.3** | INACTIVE | ACTIVE |  | No Change |
| **R: 1.4.3.1.4** | ACTIVE | INACTIVE |  | No Change |
| **R: 1.4.3.1.5** | ACTIVE | ACTIVE |  | ACTIVE |

**Table 1.4‑2 Process 1 Determine Dataflow Status**



**Figure 1.4‑4 Process 2 Convention State Machine**

### Glossary of Terms

| Acronyms and terms used in this document that may not be commonplace in the engineering world. | |
| --- | --- |
| Table ‑ Glossary of Terms | |
| **Term** | **Definition as used and applied in this functional specification** |
| Active | ON or Enable |
| A/D | Analog to Digital convertor |
| ANI | Agree not to implement  Functions with this flag are not included in the program. No planning/status entry is expected in any column further to the right of it in the FIP template. |
| Awake | All individual state machines are evaluating their operating conditions. (Reference sections 1.4.1 – 1.4.10) |
| AVAS | Approaching Vehicle Audible System |
| BCM | Body Control Module |
| Cfg | Configure, Configuration, Configurable |
| Cmd | Command |
| Constant | In Program Memory, Only Suppler can change this value (Flash/Re-Flash/ROM or EEPROM) |
| Disable | OFF |
| DTC | Diagnostic Trouble Code |
| EEPROM | Electrically Erasable Programmable Read Only Memory |
| EESE | Electrical / Electronic Systems Engineering |
| EESS | Electrical / Electronic System Specification |
| Enable | ON |
| Ev | Event |
| ICE | Internal Combustion Engine |
| Inactive | OFF or Disable |
| INDEF | Indefinitely |
| Initial State | Upon reset this is the value that the data flow is to take until a new value can be obtained. |
| Initial Value | Upon reset this is the value that the data flow is to take until a new value can be obtained. |
| NM | Network Manager |
| Network Receive Default | This is the timed out value that the data flow is to take until a new value is received over the network. |
| Non-Volatile Customer SET | Customer uses feature to change this value. Diagnostics can change this value. (EEPROM) |
| Non-Volatile Factory SET | Diagnostics can change this value. (EEPROM) |
| N/A | Not Applicable |
| OFF | Off / Disable |
| ON | On / Enable |
| Out | Output |
| PNI | Plan not to implement  Functions with this flag are included in the program but are not implemented at the relevant integration point. This flag is only applicable to "Planned" column. No entry is expected in the "Actual" column in the FIP template. |
| Received Default | This is the timed out value that the data flow is to take until a new value is received over the network. |
| Req or Rqst | Request |
| Rqmt. No. | Requirement Number <n>, **R: 1.4.1** |
| **R: 1.4.1** | For example, **R: 1.4.1** is requirement number 1 in Section 1.4 of this document. Requirements are text denoted as "Caption, the font is Times New Roman 8pt, **BOLD"** to identify a requirement. All Requirements are Level 4 Captions.  Requirements are sequenced based upon the "Heading 2" numbering sequence |
| Reset | To force the state machine to a known condition. Each state machine may have an independent reset condition not defined in the software requirements. This independent condition shall not conflict with the body module software requirements. Reference the software requirements document for conditions that may or may not cause a reset. |
| Sleep | All of the sleep criteria for the module have been met. All individual state machines must retain their current states prior to sleep. Retained states to be used as defined in the Wake Up definition. (Reference software requirement 62, and sections 1.4.1) |
| Toggle | To change state for example from On to OFF, or Disable to Enable |
| Volatile | Changes during run-time, program execution (RAM) |
| Wake Up | An input event that results in a transition from the module SLEEP state to the AWAKE state. All individual state machines resume at their previously retained states as defined in the Sleep definition. (Reference sections 1.4.1) |

### Assumed Interface to NVRAM Manager

**Note:** This document assumes the following interface with an EEPROM manager. The supplier does not need to support this exact interface.

**NVRAM\_Rqst** **NULL** = do nothing

**UPDATE** = post all changes to NVRAM (going to reboot)

**STOP** = Let any current write finish – don't start another

**NVRAM\_Status** **BUSY** = NVRAM is busy, write is occurring

**NULL** = NVRAM is not busy

**DONE** = request for UPDATE or STOP has been completed

This interface is used in the Sleep/Awake feature and in the Diagnostics feature.

### Timer Mark Event / Time Since Event

|  |  |  |
| --- | --- | --- |
| Mark event xyzzy | Action | Event in time. This action marks the event “xyzzy” on an imaginary timeline. Later referenced by Time since event” |
| Time since event xyzzy > = TimeValue\_Cfg | Condition | Elapsed time: determines the amount of time that has elapsed since the last occurrence of the "Mark event xyzzy" |

**Timers in the Behavior and Implementation models**

a) The Behavior models shall implement the following timer resolutions (selection based on Data Dictionary criteria):

b) Note: all timers are 32-bit unsigned integers (U32).

c) Following API shall be used to mark and check timers:

**Mark\_Timer\_msec(TIMER\_NAME)**

**Check\_Timer\_msec(TIMER\_NAME)**

This timer runs only while the module is awake. While the resolution is in millisecond, it may be actually incremented by 5 every 5 msec, for example. This timer will retain its value across module sleep status, and be reset to 0 if the CPU is reset.

**Mark\_Timer\_sec(TIMER\_NAME)**

**Check\_Timer\_sec(TIMER\_NAME)**

This timer runs only while the module is awake. This timer will retain its value across module sleep status, and be reset to 0 if the CPU is reset.

**Mark\_GRTimer(TIMER\_NAME)**

**Check\_GRTimer(TIMER\_NAME)**

This global real timer runs while the module is supplied with power, even while the module is asleep. Its resolution is 100 milliseconds. The value is committed to EEPROM on a schedule, so that a "recent" value will be retained across a power loss. The value is also (normally) retained across short CPU resets.

Note that this means that across a power loss, the timer could "go backwards". Practically, this means that after a power loss, you must Mark a GRTimer before checking it.

Note also that this clock is not necessarily synchronized with the other 2 timers. i.e., it may run at a slightly different rate, because it must be very accurate over long time periods to be used for time of day clock on the radio.

Note that even though GRT timer resolution is 100milliseconds **Check\_GRTimer()** reports in seconds. This is to have consistent usage of timers in milliseconds or seconds.

d) The supplier may develop a timer behavior library which simulates the timer implementation.   
Test harness shall use this library to test the timers that are used in behavior library model.

All Data Dictionary time entries should use milliseconds or seconds as their units.

Note: The GRT is not allowed to use 0xFFFFFFFF (which indicates "unknown" in the vehicle CAN network). So every 13.6 years, we will have a 100msec error in GRT (which we will ignore). The GRT timer actually uses 100msec resolution. However, all uses of GRT use seconds as the unit. This means the calculation performed by the subroutine must be (ActualGRTime - BaseGRTime)/10 to convert to seconds.

**Example Timer Implementation**

Timer is 32-bit unsigned integer (U32).

|  |  |  |  |
| --- | --- | --- | --- |
| **Mark Event** | **Timer Base Unit** | **Time Since Event** | **Eng. Unit** |
| Mark\_Timer\_msec | Milliseconds | Actual - Base >= Target | milliseconds |
| Mark\_Timer\_sec | second | Actual - Base >= Target | second |
| Mark\_GRTTimer | 100 milliseconds | Actual – Base >= Target  10 | second |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | **Time Value  Hexadecimal (U32)** | **Decimal** |
|  |  |  | 0000 0000 | 0 |
| Mark event | Base |  | 0000 0001 | 1 |
|  |  |  | 0000 0002 | 2 |
|  |  |  | 0000 0003 | 3 |
|  |  |  | 0000 0004 | 4 |
|  |  |  | 0000 0005 | 5 |
| Time since event | Actual |  | 0000 0006 | 6 |
|  |  |  | 0000 0007 | 7 |
|  |  |  | 0000 0008 | 8 |

Example computer computation:

Actual - Base

Actual + Two's Compliment of (Base)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Actual | 6 |  | 0000 0006 |  | 0000 0006 | Time Since Event |
| - Base | -1 |  | 0000 0001 |  | + FFFF FFFF | Mark Event 2's Compliment of Base |
|  | 5 |  | 0000 0005 |  | 0000 0005 |  |

Compare Result of **Actual + Two's Compliment of (Base)** to **Target** (Note: Target is typically a method 3 **TimeValue\_Cfg**)

Now compare **0000 0005** to the **Target** (TimeValue\_Cfg)

## RELATED DOCUMENTS

Related documents, referenced elsewhere in this document, are listed below for quick reference.

|  | | |
| --- | --- | --- |
| **Table 1.5‑1 – Related Documents** | | |
| **Item** | **Title** | **Control Number** |
| 1 | Engineering Specification | ES-HG9T-14G113-AA |
| 2 | Minimum Noise Target Summary | V2 |
| **SDS Requirements** | | |
| 3 | SDS, ELCOMP Generic Body Module SDS | Rev.: 21 |
| 4 | SDS, MPLELC Generic Body E/E Feature Function SDS | Rev.: 29 |
| **Requirements for PARSED** | | |
| 4a | [DVM-0055-EY EconoCentralConfigTest Spec](https://pd1.extspt.ford.com/sites/AVAS/Core%20Documents/PARSED%20Additional%20Documents/DVM-0055-EY%20EconoCentralConfigTest%20SpecV1.0.pdf) | Latest Version at time of Work Agreement |
| 4b | [Economized Central Configuration Specification](https://pd1.extspt.ford.com/sites/AVAS/Core%20Documents/PARSED%20Additional%20Documents/Economized%20Central%20Configuration%20Specification%20003.doc) | Latest Version at time of Work Agreement |
| 4c | [On Vehicle Telematics Protocol Specification](https://pd1.extspt.ford.com/sites/AVAS/Core%20Documents/PARSED%20Additional%20Documents/On%20Vehicle%20Telematics%20Protocol%20Specification_v003.pdf) | Latest Version at time of Work Agreement |
| 4d | Central Software PARSED Minimum Requirements | Latest Version at time of Work Agreement |
| **Body Modules Software Requirements** | | |
| 5 | ECU SW Design Rules - Body | EESE-SMD-CSE-PG-033 Version 2017.1 |
| 6 | ECU SW Testing Requirements | EESE-SMD-CSE-PG-034 Version 2017.1 |
| 7 | Global eSOW Software Attachment | EESE-SMD-CSE-PG-070 Version 2018.0 |
| 8 | Global eSOW SW Attachment – Read me first | N/A |
| 9 | Non-Volatile Memory Requirements | EESE-SMD-CSE-PG-070 v2017.1 |
| 10 | Outputs Fault Management | EESE-SMD-CSE-PG-032 Version 2017.1 |
| 11 | Software Release Notes | EESE-SMD-CSE-PG-044 ver 2009.0 |
| 12 | SWQA\_Common TDR Checklist | Version: 2018\_0 |
| 13 | Voltage Range Monitor | EESE-SMD-CSE-PG-035 Version 2018.0 |

|  |  |  |
| --- | --- | --- |
| **Multiplex Communications & Diagnostic Specifications** | | |
| 14 | EESE Network Communications Diagnostic Statement of Work | Version 2018.\_1 |
| 15 | EESE Network Communications Diagnostic Statement of Work Appendix A | Version 2018\_1 |
| 16 | EESE Network Communications Multiplex Technology Statement of Work (NetCom MUX SOW GPDS) | 2018.1 |
| 17 | EESE Network Communications Multiplex Technology Statement of Work Appendix A (SOW\_AppendixA) | 2018.1 |
|  | Note: This is not an exhaustive list of Netcom requirements documents. A full list including correct version number should be obtainable from the program’s Netcom Application Engineer. All versioning information should come from the aforementioned source. | |
|  |
|  |
|  |
| **Diagnostic Specifications** | | |
| 29 | Generic Global Diagnostics Specification (Part I) – GGDS-00.06.15.001-004.pdf | Issue Index 004, Volume No 01 (2013-05-02) |
| 30 | GGDS-004\_Errata\_15AUG17.pdf | 15 AUG 2017 |
| 31 | Software Download Specification | 00.06.15.002-006 |
| 32 | SWDL Errata (SWDL-006\_Errata\_06DEC17.pdf) | 06 DEC 2017 |
| 33 | Software Download Functional Test Specification 006 | 00.06.15.223-003 |
| 34 | Versatile Binary Format Specification 3.1 | 00.06.15.004-008 |
| 35 | VBF 3.1 Test Specification | 00.06.15.245\_01 |
| 36 | FS-ML3T-14G113-AA Data Dictionary V 1.0 | 21 AUG 2018 |
|  | Note: This is not an exhaustive list of Netcom requirements documents. A full list including correct version number should be obtainable from the program’s Netcom Application Engineer. All versioning information should come from the aforementioned source. | |
| **Cybersecurity Specifications** | | |
| 37 | [ReqSTD-2018-02-23-09-16](https://pd1.extspt.ford.com/sites/AVAS/AVAS%20SW/Security/ReqSTD-2018-02-23-09-16.pdf) (Cybersecurity general requirements) | Latest revision in FEDE at time of signing statement of work, at time of this release 2018-02-23-09-16 is current |
| 38 | [B.10 Ford\_CyberAssurance-SOW\_Release](https://pd1.extspt.ford.com/sites/AVAS/AVAS%20SW/Security/B.10%20Ford_CyberAssurance-SOW_ReleaseV1.1.doc) | Latest Revision at time of signing statement of work. |
| 39 | App Signing Requirements | Latest Revision at time of signing statement of work. |
| 40 | Cybersecurity DV Test Plan (DV test plan requirements from CyberSecurity Team) | Latest Revision at time of signing statement of work. |
|  |  |  |

### PARSED

PARSED functionality and requirements are defined by FORD CENTRAL SOFTWARE and the PARSED development team. Requirements cascaded for implementation of PARSED is captured by reference items [1] through [4] and were current at the writing of this functional spec. The most up-to-date version of these requirements should be requested from the PARSED team and included in the Statement of Work. Additionally, the PARSED team provides a test environment that is available to suppliers on request.

Note that these are minimum requirements. AVAS/PACM suppliers should work with the AVAS/PACM team to support any additional information that should be passed via PARSED. These items should be tracked in this FS or documented and made freely available in the AVAS Sharepoint with a reference to the document added here.

### Cybersecurity

Cybersecurity requirements and statement of work are written and maintained by the Ford Security Team. For the ease of tracking, we have listed basic Cybersecurity documents in the Cybersecurity Specifications section but these are not meant to include all required documentation for Cybersecurity. The full list of documents can be found in the CyberAssurance Statement of work [38]. Actual requirements for Cybersecurity are listed in FEDE. The latest revision of these documents at the time of signing the statement of work should be used.

## General Requirements

### Module - Memory / Power Up / Microcontroller Reset

#### Microcontroller memory Storage Classification Requirements:

The data dictionary specifies a "Storage Class" for every data flow used in this functional specification. The following five (5) requirements further specify / define the memory storage classes:

Table ‑ Memory Storage Classification Requirements

|  |  |  |
| --- | --- | --- |
| **Rqmt. No.** | **Memory Storage Classification** | **Definition** |
| **R: 1.6.1.1.1** | Constant | Re-Program the program memory to change it.  (FLASH / ROM or EEPROM, Named Complier Constants) |
| **R: 1.6.1.1.2** | Non-Volatile – Customer Set | Customer uses feature to change it.  Diagnostics can change it. (EEPROM) |
| **R: 1.6.1.1.3** | Non-Volatile -- Factory Set Method 2 | Diagnostics can change it. (EEPROM) |
| **R: 1.6.1.1.4** | Non-Volatile -- Factory Set Method 3 | Diagnostics can change it. (EEPROM) |
| **R: 1.6.1.1.5** | Non-Volatile – Functional Requirement | Changes during program run – time. (EEPROM) |
| **R: 1.6.1.1.6** | Volatile | Changes during program run – time. (RAM) |

#### Non-volatile Memory Requirements:

**Table 1.6‑2 Non-Volatile Memory Generic Requirements**

|  |  |
| --- | --- |
| **Rqmt. No.** | **Requirement** |
| **R: 1.6.1.2.1** | NVRAM Management According to the requirements & expectations for development (RED)  nonvolatile memory (NVM) document #EESE-SMD-CSE-PG-032 v2016.1 |
| **R: 1.6.1.2.2** | ECU SW Design Rules – Body version 2016.2 |

#### Module Power Up / Microcontroller Reset Requirements:

**Table 1.6‑3 Power Up / Reset Requirements**

|  |  |  |
| --- | --- | --- |
| **Rqmt. No.** | **Memory storage Class** | **Description** |
| **R: 1.6.1.3.1** | Volatile | Upon module power-up and / or upon module reset the body feature initial values specified in the Data dictionary shall be used. |
| **R: 1.6.1.3.2** | ALL Non-Volatile | Shall be set to the initial values specified in the Data Dictionary prior to delivery to FORD. |
| **R: 1.6.1.3.3** | Constant | Shall be set to the initial value specified in the Data dictionary |
| **R: 1.6.1.3.4** | Upon module power-up and / or upon module reset all of the decision tables defined within this functional specification must have all of the output data flows set to the initial value as specified in the Data Dictionary | |

### Software Classification Level

The functional importance classifications are defined in RQT-191001-009906/18.

***FUNCTIONAL IMPORTANCE CLASS C:*** Any function that is essential to the operation or control of the vehicle (e.g., braking, engine management) or is essential for safe vehicle operation.

***FUNCTIONAL IMPORTANCE CLASS B:*** Any function that enhances the operation or control of the vehicle, but that is not essential to the operation or control of the vehicle.

***FUNCTIONAL IMPORTANCE CLASS A:*** All other functions (e.g., comfort, entertainment).

Functional classifications are used when considering design failure modes and associated design mitigations both at the vehicle/subsystem level and at component level. Functional classifications are also used to describe acceptance criteria associated with particular tests (e.g. EMC Radiated Immunity).

Table ‑ AVAS Software Classification Level Requirement

|  |  |
| --- | --- |
| **Requirement Number** | **Description** |
| R: . | AVAS is defined as a FUNCTIONAL IMPORTANCE CLASS C system |
| NOTE: AVAS has been reviewed by the EESE Functional Safety team. A determination was made that the AVAS module had an overall Automotive Safety Integrity Level (ASIL) of QM. | |

### Timing Requirements

#### Timing / Response Requirements

|  |  |
| --- | --- |
| **R: 1.6.3.1.1** | T-1: Unless stated otherwise in the individual feature specification, activation / deactivation of an output response shall occur within the maximum delay time of the corresponding input(s) change as defined in  *Table* 1.6‑4 below. The maximum delay time is pin-to-pin, including debounce time and multiplex messaging. |
| **R: 1.6.3.1.2** | T1.1 Since ignition input debounce is longer than 100 milliseconds and is an exception to requirement T1.0, 55 milliseconds is the maximum delay time for output device activation after an ignition change is debounced. |
| **R: 1.6.3.1.3** | T-2: The time tolerances of all timing requirements are +/- 10% unless otherwise stated. |

### Order of Execution

Order of execution is important to prevent momentary output glitches and to ensure consistent sets of related outputs.

Software implementation shall comply with the requirements in Tables 1.6.5-1 and 1.6.5-2.

#### Order of Execution Requirements

Table ‑ Overall Order of Execution

|  |  |
| --- | --- |
| **Rqmt. No.** | **Requirement** |
| **R: 1.6.5.1.1** | The flow of data within this FS is generally organized from process to process in this order:  input,  feature,  arbitrator,  output. Within each feature category (e.g. Exterior Lighting) all inputs shall be executed before a feature is executed. |
| **R: 1.6.5.1.2** | All features that feed an arbitrator shall be executed together as a group. This will ensure that the features operate on a single set of input values and provide a single consistent set of output values to downstream processes. |
| **R: 1.6.5.1.3** | The arbitrator(s) that feed an output shall be executed before the output is executed. |

Table 1.6.5‑‑ Internal Process Order of Execution

|  |  |
| --- | --- |
| **Rqmt. No.** | **Requirement** |
| **R: 1.6.5.1.5** | Each (input, feature, arbitrator, and output) process shall be executed completely in a single time-slice. |
| **R: 1.6.5.1.6** | Each (input, feature, arbitrator, and output) process shall be executed atomically. |
| **R: 1.6.5.1.7** | Within every (input, feature, arbitrator, and output) process, data generally flows in the order in which decision tables and state transition diagrams are presented in this FS. For example, a decision table may feed a state transition diagram which may feed another decision table. The elements within a process shall be executed in the order of this internal data flow. |

# APPROACHING VEHICLE AUDIBLE SYSTEM (AVAS)

## Feature Behavior Summary

This functional specification defines system functionality for the Approaching Vehicle Audible System intended for electric vehicles.

Due to quiet operation on Electric Vehicles (EV), Hybrid Electric Vehicles (HEV) and Plug-in Hybrid Electric Vehicle (PHEV) at low vehicle speeds, there exists a higher risk of vehicle/pedestrians collisions. The purpose of the AVAS is to have the function of a ‘vehicle sounder’ to create pleasing noise to alert pedestrians of the presence of vehicles when they are nearby. AVAS should provide alerting information at least equivalent to the cues provided by internal combustion engine (ICE).

The sounder will operate as the applicable government regulation states.

The system should do FADE in/out for some pressure level and frequencies when vehicle is transitioning among all normal operation scenarios listed below.

Normal operations are:

**Stationary:**

In the case of a vehicle with an automatic transmission, the vehicle’s gear selector is in Neutral or any gear position other than Park that provides forward vehicle propulsion sound must be enabled.

**Park:**

When the vehicle is in park AVAS should be INACTIVE and sound must be disabled.

**Reverse:**

Whenever the vehicle’s gear selector is in the Reverse position sound must be enabled.

**Drive:**

Whenever the vehicle’s gear selector is in Drive/Low/Sport position sound must be enabled.

**Acceleration and Deceleration:**

Pitch Shifting is a characteristic sound where pedestrian associate a vehicle with the acceleration/decelerations based on the sound emitted by ICE vehicles. The sound produced on this scenario should be the same as the sound produced by the ICE vehicles when speed increases or decreases.

**Constant Speed:**

In this scenario the vehicle should produce a sound at certain speed, especially at low speed levels for example: 10 km/hr, 20 km/hr or 30 km/hr. On this scenario the vehicle should produce a minimum sound level in case vehicle is in this constant speed.

The sound levels of this module should follow any regional regulations. Sound levels or tones should be defined by NVH team.

## Subsystem Overview

The context diagram for the AVAS module is illustrated in **Figure 2.2‑1 AVAS Subsystem Overview**.



Figure ‑ AVAS Subsystem Overview

## Feature Behavior Detail

The AVAS module is a component of electric vehicles EV, PHEV or HEV. When the vehicle’s powertrain pack (see **Table 2.5‑2 Translation of PwPckTq\_D\_Stat Status Signal** PwPckOn\_TqAvailable) is in a state where it is enabled and capable of producing motive torque on demand by the driver while stationary or moving in Reverse or Forward direction (see **Table 2.4‑1 Translation of GearRverse\_D\_Actl** signal, Active\_Confirmed), and below the top speed defined for the system (e.g. 30KPH in first release of FMVSS 141), the module should generate a sound that alerts pedestrians of vehicle presence. The module should not generate a sound in park. The module also does not make sound when the ignition status is off. (**Table 2.5‑4 Determining Vehicle Mode** and **Table 2.7‑1 Determine AVAS state**)



Figure ‑ Context Diagram

## Determine Transmission Status

To determine vehicle direction, the module needs the message 0x202 EngVehicleSpThrottle2. Signal GearRvrse\_D\_Actl determines if reverse gear is in use or not, ACTIVE state is set when vehicle is going in reverse direction, and INACTIVE is set when direction is going forward.



Figure ‑ Vehicle Gear Reverse Status Data Flow

Table ‑ Translation of GearRverse\_D\_Actl signal

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rqmt. No.** | **GearRverse\_Status\_Available** | **GearRvrse\_D\_Actl** | **Values** | **GearRverse\_Status** |
| R: 2.4.0 | WAITING | Don’t Care | N/A | INACTIVE |
| R: 2.4.1 | LOST | Don’t Care | N/A | LOST |
| R: 2.4.2 | AVAILABLE | Active\_Confirmed | 0x3 | ACTIVE |
| R: 2.4.3 | All Other Valid Combination of Values | | | INACTIVE |

**Figure 2.4‑2 Gear Reverse Signal Status Transition Diagram** shows the logic to transition between the states of the GearRvese\_Signal\_Status including timing. When status equals LOST and GearRevrse\_D\_Actl\_Signal\_Received\_Flag = NULL, the module should set a DTC, please see Diagnostics section below.



Figure ‑ Gear Reverse Signal Status Transition Diagram

In order to prevent an audible delay in sound output when the driver moves the gear lever position through the Reverse gear position, AVAS shall delay for TransitionDelayCfg before transitioning GearRverse\_Status from an Active to Inactive state and vice versa. **Figure 2.4‑3 Determine GearRverse\_Status Active-Inactive State** runs while GearRverse\_Status\_Available = Available and ensures that there is no audible pause in the sound when the driver shifts through but does not remain in park.



Figure ‑ Determine GearRverse\_Status Active-Inactive State

When the gear lever position is in park the AVAS system should be INACTIVE. To determine if the vehicle gear position is park, the module receives the signal GearLvrPos\_D\_Actl (in HEV vehicles before Gen IV). If the gear is in a park position, GearPark\_Status is set to ACTIVE. If the vehicle’s gear position is not park, GearPark\_Status is set as INACTIVE. If the signal GearLvrPos\_D\_Actl is lost, determined by the value GearLrvPosAvailable, then GearPark\_Status is set to LOST. **Figure 2.4‑4 Vehicle Gear Park Status Data Flow** details how to determine if the GearPark\_Available state is Lost or Available.



Figure ‑ Vehicle Gear Park Status Data Flow

Table ‑ Translation of GearLvrPos\_D\_Actl signal and GearPark\_Status

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rqmt. No.** | **GearLvrPos\_Available** | **GearLvrPos\_D\_Actl** | **Values** | **GearPark\_Status** |
| R: 2.4.3a | WAITING | Don’t Care | Don’t Care | INACTIVE |
| R: . | LOST | Don’t Care | Don’t Care | LOST |
| R: . | AVAILABLE | Park | 0x0 | ACTIVE |
| R: . | All Other Valid Combination of Values | | | INACTIVE |

**Figure 2.4‑5 Determine Park Status Transition Diagram** shows the logic to transition to the different states of GearPark\_Status. When GearLvrPos\_Available equals LOST and GearLvrPos\_D\_Actl\_Signal\_Received\_Flag = NULL, the module should set a DTC, please see Diagnostics section below.



Figure ‑ Determine Park Status Transition Diagram

In order to determine the AVAS state, the GearPark\_Status and the GearRverse\_Status\_Available are combined into the Trans\_Signal\_Status whose value is used to determine the AVAS state in **Table 2.7‑1 Determine AVAS state**.

Table ‑ Determination of Trans\_Signal\_Status

|  |  |  |  |
| --- | --- | --- | --- |
| **Rqmt. No.** | **GearRverse\_Status** | **GearPark\_Status** | **Trans\_Signal\_Status** |
| R: 2.4.7 | LOST | Don’t Care | LOST | |
| **R: 2.4.8** | Don’t Care | LOST | LOST | |
| **R: 2.4.9** | ACTIVE | Don’t Care | ACTIVE | |
| **R: 2.4.10** | All Other Valid Combination of Values | | INACTIVE | |

## Determine Vehicle Mode Status

To determine Vehicle mode, the module should receive messages 0x167 VehicleOperatingModes and 0x3B3 BodyInfo\_3. Signals PwPackTq\_D\_Stat and Ignition Status helps to determine mode status of the vehicle. The results of the evaluation represent valid modes of the vehicle.



Figure ‑ Determine Vehicle Mode Status Data Flow 1

Before using the values of PwPckTq\_D\_Stat and Ignition\_Status, the AVAS module must check to see if the signals are available or lost. If the signal is lost, we have no valid values to use. **Figure 2.5‑2 Determine PwPckTq Status Transition Diagram** shows the state machine that determines whether the PwPckTq\_D\_Stat signal is available or lost.



Figure ‑ Determine PwPckTq Status Transition Diagram

When the status of PwPckTq\_Status\_Available = LOST and PwPckTq\_D\_Stat\_Signal\_Received\_Flag = NULL, an appropriate DTC should be set. Please see the Diagnostics section below.

**Figure 2.5‑3 Determine Ignition\_Status Signal Status Transition Diagram** shows the state machine that determines whether the Ignition\_Status signal is available or lost.



Figure ‑ Determine Ignition\_Status Signal Status Transition Diagram

When the Ignition\_Status\_Available = LOST and the Ignition\_Status\_Received\_Flag = NULL, a LOST DTC must be set. Please see the Diagnostics section below.

The values of Ignition\_Status\_Available and PwPckTq\_Status\_Available will be combined into one value, PwPck&IgnStat\_Signal\_Status as shown in the table below.

Table ‑ Determination of PwPck&IgnStat\_Signal\_Status

|  |  |  |  |
| --- | --- | --- | --- |
| **Rqmt. No.** | **PwPckTq\_Status\_Available** | **Ignition\_Status\_Available** | **PwPck&IgnStat\_Signal\_Status** |
| R: 2.5.1 | | Lost | Don’t Care | Lost | |
| R: 2.5.2 | | Don’t Care | Lost | Lost | |
| R: 2.5.3 | | Available | Available | Available | |

Provided the signals are available and not lost, the value of the signals are translated to internal values used by AVAS as specified in **Table 2.5‑2 Translation of PwPckTq\_D\_Stat Status Signal** and **Table 2.5‑3 Translation of Ignitions Status** below.

Table ‑ Translation of PwPckTq\_D\_Stat Status Signal

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CAN ID** | **CAN Message** | **Signal** | **Detailed**  **Meaning** | **State**  **Encoded** | **AVAS\_PwPckTq\_D\_Stat** |
| 0x167 | VehicleOperatingModes | PwPckTq\_D\_Stat | PwPckOff\_TqNotAvailable | 0x0 | OFF\_NO\_TQ |
| PwPckOn\_TqNotAvailable | 0x1 | ON\_NO\_TQ |
| PwPckStrtInProg\_TqNotAvail | 0x2 | START\_IN\_PROGRESS |
| PwPckOn\_TqAvailable | 0x3 | ON\_TQ\_AVAILABLE |

Table ‑ Translation of Ignitions Status

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CAN ID** | **CAN Message** | **Signal** | **Detailed**  **Meaning** | **State**  **Encoded** | **AVAS\_Ignition\_Status** |
| 0x3B3 | BodyInfo\_3 | Ignition\_Status | Accessory | 0x2 | ACCESORY |
| Run | 0x4 | RUN |
| Start | 0x8 | START |
| Invalid | 0xF | INVALID |
| Off | 0x1 | OFF |
| Unknown | 0x0 | UNKNOWN |

The availability of the PwPckTq\_D\_Stat and Ignition\_Status signal, the GearPark\_Status and the internal signal values AVAS\_Ignition\_Status and AVAS\_PwPckTq\_D\_Stat are used to evaluate the current vehicle mode. This value is stored as Vehicle\_Mode.

Table ‑ Determining Vehicle Mode

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Rqmt. No.** | **PwPck&**  **IgnStat\_Signal\_Status** | **GearPark\_Status** | **AVAS\_Ignition\_Status** | **AVAS\_PwPckTq\_D\_Stat** | **Vehicle\_Mode** |
| **R: 2.5.4** | AVAILABLE | Don’t Care | Accessory | Don´t Care | ACCESORY |
| **R: 2.5.5** | AVAILABLE | ACTIVE | RUN | Don’t Care | GEAR\_PARK |
| **R: 2.5.6** | AVAILABLE | < > ACTIVE | RUN | ON\_NO\_TQ | OFF\_NO\_TQ | GEAR\_PARK |
| **R: 2.5.7** | AVAILABLE | Don’t Care | Off | OFF\_NO\_TQ | IGNITION\_OFF |
| **R: 2.5.8** | AVAILABLE | Don’t Care | Don´t Care | START\_IN\_PROGRESS | CRANKING || DIAGNOSTIC\_MODE |
| **R: 2.5.9** | AVAILABLE | < > ACTIVE | RUN | ON\_TQ\_AVAILABLE | POWERPACK\_ON |
| **R: 2.5.10** | LOST | Don’t Care | Don’t Care | Don’t Care | LOST |
| **R: 2.5.11** | AVAILABLE | Don’t Care | START | Don’t Care | CRANKING |

## Determine Vehicle Speed Status

To determine vehicle speed, the module needs to receive the message 0x202 EngVehicleSpThrottle2. Signal Veh\_V\_ActlEng determine the current speed for electric vehicles. The signal VehVActlEng\_D\_Qf reports the quality of the value in Veh\_V\_ActlEng. This value may be faulty and invalid.



Figure ‑ Determine Vehicle Speed and Vehicle Speed Status

**Figure 2.6‑2 Determine Vehicle Speed Status Transition Diagram** shows how Vehicle\_Speed\_Available is determined. When the Vehicle\_Speed\_Available equals LOST and Veh\_V\_Actl\_Eng\_Signal\_Received\_Flag = NULL, the module should set a DTC as per the Diagnostics section below.



Figure ‑ Determine Vehicle Speed Status Transition Diagram

## 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Rqmt. No.** | **Vehicle\_Mode** | **Trans\_Signal\_Status** | **Vehicle\_Speed\_Available** | **Vehicle\_Speed** | **AVAS\_State** |
| R: 2.7.1 | LOST | Don´t Care | Don’t Care | Don´t Care | FAULT |
| R: 2.7.2 | Don´t Care | LOST | Don’t Care | Don´t Care | FAULT |
| R: 2.7.3 | Don´t Care | Don´t Care | LOST | Don’t Care | FAULT |
| R: 2.7.3a | < > LOST | < > LOST | WAITING | Don’t Care | INACTIVE |
| R: 2.7.4 | Don´t Care | Don´t Care | AVAILABLE | > Max\_Speed\_Sound\_Cfg | INACTIVE |
| R: 2.7.5 | ACCESORY | Don´t Care | AVAILABLE | <= Max\_Speed\_Sound\_Cfg | INACTIVE |
| R: 2.7.6 | CRANKING | Don´t Care | AVAILABLE | <= Max\_Speed\_Sound\_Cfg | INACTIVE |
| R: 2.7.7 | IGNITION\_OFF | Don´t Care | AVAILABLE | <= Max\_Speed\_Sound\_Cfg | INACTIVE |
| R: 2.7.8 | POWERPACK\_ON | Don´t Care | AVAILABLE | <= Max\_Speed\_Sound\_Cfg | ACTIVE |
| R: 2.7.9 | GEAR\_PARK | Don’t Care | Don’t Care | Don’t Care | INACTIVE |
| R: 2.7.10 | Check Diagnostics Section for Fault conditions | | | | FAULT |

## Determine AVAS State

The state of the AVAS ECU provides information when the module is ACTIVE, INACTIVE or FAULT. The module should be capable of informing when the module is producing sound, meaning; the state should be set to ACTIVE when the module is producing sound; INACTIVE when the module is not producing sound, and if there is something wrong with some of the input signals or due to some internal failures, then the module should set the state to FAULT.

The module should produce sound if all the elements such as the inputs necessary to determine the state of the system, or the internal variables and components of the AVAS system are correct. The module should check for a specific speed limit defined internally in the CONFIGURATION file and after exceeding it, it should not produce any sound. This maximum speed depends on the value of Max\_Speed\_Sound\_Cfg.

The module should be capable to detect failures that compromise correct operation, these failures are:

* + - Speaker#1 Failure (Generally the Front Speaker or only speaker in a one speaker system)
    - Speaker#2 Failure (Generally the rear speaker if it exists on the hardware)
    - Amplifier failure
    - Unknown state
    - Lost communication or signal reported as faulted
    - Internal failures

For more detailed information on specific failures refer to **2.10 Diagnostics**. When the failures above are present, the module should set AVAS\_State to FAULT.

The next table represents the internal status of the AVAS module represented with the dataflow AVAS\_State.

Table ‑ Determine AVAS state

### AVAS Fault State Signal

The module should transmit a CAN message PedestrianAlert\_Data with ID 0x40C, signal name is PdstrnAlrt\_B\_Falt. This signal should indicate current condition of the module. This signal should be transmitted when vehicle mode is in ACCESORY, DIAGNOSTICS\_MODE, CRANKING, GEAR\_NUETRAL, and POWERPACK\_ON as defined in **Table 2.5‑4 Determining Vehicle Mode.**

* YES state indicates module is not working and a failure is present.
* NO state indicates module is working OK.



Figure ‑ AVAS Fault State Flow Diagram

Table ‑ PdstrnAlrt\_B\_Falt signal status

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CAN ID** | **CAN Message** | **Signal** | **Detailed**  **Meaning** | **State**  **Encoded** |
| 0x40C | PedestrianAlert\_Data | PdstrnAlrt\_B\_Falt | PACM\_Fault\_NO | 0x0 |
| PACM\_Fault\_YES | 0x1 |

The module should set PdstrnAlrt\_B\_Falt according to AVAS\_State:

Table ‑ PdsTrnAlrt\_B\_Falt set conditions

|  |  |  |
| --- | --- | --- |
| Requirement Number | AVAS\_State | PdstrnAlrt\_B\_Falt |
| R: 2.7.11 | ACTIVE || INACTIVE | PACM\_Fault\_NO |
| R: 2.7.12 | FAULT | PACM\_Fault\_YES |

## Generate AVAS Output

### General AVAS Output Requirements

The following requirements apply to generating the AVAS output:

**Table 2.8.1.1 General AVAS Output Requirements**

|  |  |
| --- | --- |
| **Rqmt. No.** | **Description** |
| R: 2.8.1 | AVAS system must be able to produce Broadband sound(s); adjustable for all 1/3 octave bands between 315 Hz and 5 KHz. |
| R: 2.8.2 | The software sound strategy must include a time domain signal generation and frequency domain filter blocks. |
| R: 2.8.3 | The signal generation block must use either a wave or grain based signal generation. For wave designs, the input files should be a minimum of 5 seconds at 16 kHz and 16 bit. The design must accommodate a minimum of two WAV files. For grain designs, the design must accommodate five grains of one second each. Design must be approved by NVH prior to implementation. |
| R: 2.8.4 | The supplier must deliver a tuning tool to allow Ford to modify both the signal generation and the filter blocks. |
| R: 2.8.5 | The system must also be capable of adjusting an overall gain vs speed within a 100 dB range. |
| R: 2.8.6 | AVAS must have the capability to statically equalize the sound with respect to the vehicle transfer function using either a 1025 tap FIR filter or a bank of 8 IIR filters which include parametric peaking/notch, high pass, low pass, parametric bass shelf and parametric treble shelf. |
| R: 2.8.7 | The calibration files must separate the time history components from the tuning parameters to enable end of line flash parameters. |
| R: 2.8.8 | NVH will determine whether the supplier sound generation capability and GUI is capable of producing the desired sound characteristics per vehicle line and per requirement. |
| Note: The way of communicating this information should be agreed between the Ford vehicle NVH engineer, the Ford AVAS application D&R engineer and the supplier. | |

### AVAS Required Signal Flow

The figure below details the AVAS Digital Signal Processing flow and the required processing of sounds. Configuration for these processes should be through a GUI tool that allows a technician to set configuration values through a graphical user interface (GUI) and creates a configuration file for use with the AVAS ECU.



Figure ‑ AVAS Digital Signal Processing Flow

### Signal Generation Block Requirements

Ford Motor Company prefers for suppliers to implement an optimized signal generation strategy. A signal generation strategy should be designed by the supplier and approved by Ford Motor Company before final implementation. The concepts for Signal Generation presented in this section are example implementations but may be used as the basis for a supplier’s signal generation design.



Figure ‑ AVAS Signal Generation Block Concepts

Table ‑ Requirements for the AVAS Signal Generation Block

|  |  |
| --- | --- |
| Requirement Number | Description |
| R: 2.8.9 | The supplier shall present and implement a strategy for signal generation that is approved by Ford Motor Company. |
| R: 2.8.10 | The Signal Generation Block shall create sound using time domain techniques such as wave or grain synthesis. |

### Filter Block Requirements

Ford Motor Company prefers for suppliers to implement an optimized filtering strategy. A filtering strategy should be designed by the supplier and approved by Ford Motor Company before final implementation. The concepts for filtering presented in this section are example implementations but may be used as the basis for a supplier’s signal generation design.



Figure ‑ AVAS Filter Block Concepts

Table ‑ Requirements for the AVAS Filter Block

|  |  |
| --- | --- |
| Requirement Number | Description |
| R: 2.8.11 | The supplier shall present and implement a strategy for signal filtering that is approved by Ford Motor Company. |
| R: 2.8.12 | A static frequency filter block shall be used to enable faster tuning time and reduced vehicle usage. |

### One Speaker Systems versus Multi-Speaker Systems

The following requirements apply to configuration and functionality of a one-speaker system versus a multi-speaker system.

Table ‑ Requirements for One Speaker and Multi-Speaker Systems

|  |  |
| --- | --- |
| Requirement Number | Description |
| R: 2.8.13 | AVAS\_Spkr\_Cfg shall be used to configure AVAS to be a one-speaker system or a multi-speaker system. |
| **R: 2.8.14** | In a one speaker system, all sound will be played from the single speaker (ACTIVE\_SPKR1) |
| **R: 2.8.15** | In a two speaker system, the reverse sound will be played from the rear speaker (ACTIVE\_SPKR2) and any other sounds will be played through the front speaker (ACTIVE\_SPKR1) |

**Figure 2.8‑4 Determination of Which Speaker is Playing Audio: Two Speaker System** demonstrates how to determine which speaker is being used for playback.



Figure ‑ Determination of Which Speaker is Playing Audio: Two Speaker System

### AVAS Output Timing Requirements

Table ‑ AVAS Specific Timing Requirements

| **Rqm't Num.** | **Input Action** | **Output Response** | **Maximum Delay Time With Module  In Awake State** | **Maximum Delay Time With Module  In Sleep State** |
| --- | --- | --- | --- | --- |
| **R: 2.8.16** | Voltage:  CAN:  Vehicle Speed or  Gear Position  Ignition Status | Sound emitted by AVAS | 500 ms (see note 10) | No Sleep state |
| **R: 2.8.17** | During a transition from forward sound/audio to reverse sound/audio or vice versa, the current sound shall continue to play until the new sound is loaded and ready to play unless AVAS enters into a state where no sound should be played. | | | |

| **Notes** | Maximum Delay Time With Module In Awake State | Maximum Delay Time With Module In Sleep State |
| --- | --- | --- |
| 1 | 1a – Calculation 50 ms to debounce the input. 20 ms to process the input.  3 ms to turn on the output. | 1b – Calculation 50 ms to detect the switch has changed state  1 ms to wake up. 40 ms to debounce the input. 20 ms to process the input.  3 ms to turn on the output. |
| 2 | 2a – Calculation 90 ms to debounce the input. 20 ms to process the input.  3 ms to turn on the output. |  |
| 8 | All Maximum Delay Times are in milliseconds | |
| 9 | The input / output task schedule is every 20 milliseconds | |
| 10 | From the reception of the first CAN frames of Vehicle Speed, Reverse Status, Power Pack Torque, Gear Lever Position Actual and when voltage is in the nominal operating range.  In the case where time is needed to determine a valid signal (debounce) such as with reverse status then a debounce time may be added to the maximum delay. Maximum delay time with the module in the awake state with debounce time added cannot exceed 750 ms | |

## Voltage Range Monitor

### Voltage Range Monitor Feature Behavior Summary

This feature provides monitoring of battery and relay power voltages for proper operation of the features and outputs of the AVAS module. Different loads and features require different voltages to operate correctly. This feature maintains the state of the battery for each load/feature viz. NORMAL, LOW, HIGH, UNDER, OVER.

The voltage range monitor (VRM) will monitor the battery current according to the configuration constants defined below to determine whether or not the power to the AVAS and CAN transceiver are within normal operating range and not in danger of behaving abnormally. The feature behavior is detailed below.



Figure ‑ Battery Voltage Monitor Data Flow Diagram

A voltage range is defined to be in one of five different states. **Table 2.9‑1 VBattState[i]** Definitions lists all of these states along with the associated voltage definition and what is expected of the typical feature. Each voltage range is mapped to an entry in the VBattState[i] array.

Table ‑ VBattState[i] Definitions

|  |  |  |
| --- | --- | --- |
| **VBattState[i]** | **Voltage** | **Typical Feature Behavior** |
| NORM\_V | Normal | Normal feature behavior |
| LO\_V | Temporary Low | Most features should operate normally (there may be exceptions) |
| HI\_V | Temporary High | Most features should operate normally (there may be exceptions) |
| UNDER\_V | Too low too long, Under Voltage | Most features should shed loads (there may be exceptions) |
| OVER\_V | Too high too long, Over Voltage | Most features should shed loads (there may be exceptions) |

The following dataflows are defined for each voltage range. Note that whenever VBatt is outside the normal range (NORM\_V) VBattState[i] immediately changes to LO\_V or HIGH\_V.

Table ‑ Voltage Range Dataflows

| **Dataflow Name** | **Description** | |
| --- | --- | --- |
| VMin\_Cfg[i] | Defines minimum voltage for a *Voltage Range* |
| VMax\_Cfg[i] | Defines maximum voltage for a *Voltage Range* |
| VMinRcv\_Cfg[i] | Voltage hysteresis for recovering from an UNDER\_V condition |
| VMaxRcv\_Cfg[i] | Voltage hysteresis for recovering from an OVER\_V condition |
|  | Note: The data dictionary contains the default values of VMin\_Cfg, VMax\_Cfg, VMinRcv\_Cfg  and VMaxRcv\_Cfg. The supplier will perform a worst case analysis of the circuits. The supplier will supply the updated values for the data dictionary to be included in subsequent builds and verification and validation testing. |
|  |  | |
| LoTime\_Cfg[i] | After VBatt dips low, this is the minimum amount of time normal voltage must be present before allowing inputs to be read again (return to NORM\_V). This value must be set based on worst-case analysis of the slowest circuit that uses this *Voltage Range*. |
| HiTime\_Cfg[i] | After VBatt pops high, this is the minimum amount of time normal voltage must be present before allowing inputs to be read again (return to NORM\_V). This value must be set based on worst-case analysis of the slowest circuit that uses this *Voltage Range*. |
| LoOvrTime\_Cfg[i] | Defines the amount of time to wait in a low voltage (LO\_V) condition before declaring an UNDER\_V condition. |
| HiOvrTime\_Cfg[i] | Defines the amount of time to wait in a high voltage (HI\_V) condition before declaring an OVER\_V condition. |
| LoRecovTime\_Cfg[i] | Defines the amount of time that the voltage must be above VMin\_Cfg[i] before recovering from an UNDER\_V condition and allowing input sampling again. |
| HiRecovTime\_Cfg[i] | Defines the amount of time that the voltage must be below VMax\_Cfg[i] before recovering from an OVER\_V condition and allowing input sampling again. |
|  |  | |
| VBattState[VR\_xxx] | Current State of a specific *Voltage Range*. |

### Basic Voltage Range Requirements

The following voltage range requirements apply to each voltage range supported by the AVAS Voltage RangeMonitor:

Table ‑ Basic Voltage Range Requirements

|  |  |
| --- | --- |
| **Rqmt. No.** | **Requirement** |
| R: 2.9.1 | Each feature is mapped to a *Voltage Range* (each *Voltage Range* can support multiple features and input circuits) |
| R: 2.9.2 | VBattRaw shall always be readable (whenever the micro is running). |
| R: 2.9.3 | If a feature is implemented without using a microprocessor, then the design must guarantee the fundamental concept in this document of not responding to input values when VBattRaw voltage makes the input circuit unreliable. The design must be approved in a Design Review. |
| R: 2.9.4 | VBattRaw must be sampled every 5 milliseconds or faster. |

### Voltage Range State Machine



Figure ‑ Voltage Range Monitoring Finite State Machine

|  |  |
| --- | --- |
| R: 2.9.5 | Each of the defined *Voltage Ranges* in **Table 2.9‑3 Basic Voltage Range Requirements** must be evaluated using an instance of this state machine. |

**Table 2.9‑4 Voltage Range Monitoring State Transition Diagram Requirements** describes the transitional requirements of the Voltage Range Monitoring Finite State Machine.

Table ‑ Voltage Range Monitoring State Transition Diagram Requirements

| **Rqmt. No.** | **Source State -> Destination State. Rqmt. No** | **Description** |
| --- | --- | --- |
| R: 2.9.6 | 0 -> 4.1 | Wait normal from low on reset |
| R: 2.9.7 | 0 ->9.1 | Wait normal from over on reset to detect jump start from reset |
|  |  |  |
| R: 2.9.8 | 1->3.1 | Voltage rises to HIGH from Normal |
| R: 2.9.9 | 1->2.1 | Voltage drops to LOW from Normal |
|  |  |  |
| R: 2.9.10 | 3->5.1 | Voltage drops back to Normal range |
| R: 2.9.11 | 3->7.1 | Voltage remains HIGH for more than HiOvrTime |
|  |  |  |
| R: 2.9.12 | 5->3.1 | Voltage goes back to HIGH within HiTimeCfg |
| R: 2.9.13 | 5->1.1 | Voltage stays in NORMAL range for HiTimeCfg or more |
|  |  |  |
| R: 2.9.14 | 2->4.1 | Voltage rises back to Normal range |
| R: 2.9.15 | 2->6.1 | Voltage remains LOW for more than LoOvrTime |
|  |  |  |
| R: 2.9.16 | 4->2.1 | Voltage goes back to LOW within LoTimeCfg |
| R: 2.9.17 | 4->1.1 | Voltage stays in NORMAL range for LoTimeCfg or more |
|  |  |  |
| R: 2.9.18 | 7->9.1 | Voltage returns to NORMAL |
|  |  |  |
| R: 2.9.19 | 9->7.1 | Voltage goes back above threshold |
| R: 2.9.20 | 9->1.1 | Voltage stays in NORMAL for more than HiRecovTime\_Cfg |
|  |  |  |
| R: 2.9.21 | 6->8.1 | Voltage returns to NORMAL |
|  |  |  |
| R: 2.9.22 | 8->6.1 | Voltage goes back below threshold |
| R: 2.9.23 | 8->1.1 | Voltage stays in NORMAL for more than LoRecovTime\_Cfg |
|  |  |  |
| R: 2.9.24 | The conditions to transition from state 4 shall be evaluated in the following order (from highest to lowest priority).  <1> 4->2  <2> 4->1 | |
| R: 2.9.25 | The conditions to transition from state 5 shall be evaluated in the following order (from highest to lowest priority).  <1> 5->3  <2> 5->1 | |
| R: 2.9.26 | The conditions to transition from state 8 shall be evaluated in the following order (from highest to lowest priority).  <1> 8->6  <2> 8->1 | |
| R: 2.9.27 | The conditions to transition from state 9 shall be evaluated in the following order (from highest to lowest priority).  <1> 9->7  <2> 9->1 | |

### Voltage Range Monitor Defined Voltage Ranges

#### Voltage Range Configuration (VR\_080\_160\_VB)

Table ‑ Voltage Range Configuration for VB 8.0 -16V

|  |  |
| --- | --- |
| R: 2.9.28 | * Shown in the table below are the operating voltages that have been specified as critical for the AVAS module in this range. These are to be monitored in dataflow VBattState[VR\_080\_160\_VB]. * Reference **Figure 2.9‑2 Voltage Range Monitoring Finite State Machine** and * This voltage range may be merged into another, wider voltage range given the condition that hardware supports this |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Rqmt. No.** | **Dataflow/Other** | | **Value** | | **Description** | **Index 'i'** |
| R: 2.9.29 | VMin\_Cfg[VR\_080\_160\_VB], VMinRcv\_Cfg[VR\_080\_160\_VB] | | 8.0 volts (typical but may be less) | | Minimum voltage to operate the AVAS ECU below which the module should report as a faulted because of under voltage. | | I:0 |
| R: 2.9.30 | VMax\_Cfg[VR\_080\_160\_VB], VMaxRcv\_Cfg[VR\_080\_160\_VB] | | 16.0 volts (typical but may be more) | |
| R: 2.9.31 | LoTime\_Cfg[VR\_080\_160\_VB] | | 15 *m*sec | | Recovery time to NORM\_V from LO\_V | |
| R: 2.9.32 | HiTime\_Cfg[VR\_080\_160\_VB] | | 15 *m*sec | | Recovery time to NORM\_V from HI\_V | |
| R: 2.9.33 | LoOvrTime\_Cfg[VR\_080\_160\_VB] | | 20 *m*sec | | LO\_V too long, enter UNDER\_V | |
| R: 2.9.34 | HiOvrTime\_Cfg[VR\_080\_160\_VB] | | 160 *msec* | | HI\_V too long, enter OVER\_V | |
| R: 2.9.35 | LoRecovTime\_Cfg[VR\_080\_160\_VB] | | 200 *m*sec | | Recovery time to NORM\_V from UNDER\_V (Ref.: EC-0043) | |
| R: 2.9.36 | HiRecovTime\_Cfg[VR\_080\_160\_VB] | | 200 *m*sec | | Recovery time to NORM\_V from OVER\_V | |

#### Diagnostic Trouble Code (DTC) Logging (VR\_100\_155\_VB)

|  |  |
| --- | --- |
| R: 2.9.37 | * Shown in the table below are the operating voltages that have been specified as critical for proper logging of Diagnostic\_Trouble\_Codes\_Logging [VR\_100\_155\_VB] and are to be monitored in dataflow VBattState[VR\_100\_155\_VB]. * Reference **Figure 2.9‑2 Voltage Range Monitoring Finite State Machine** and * **Table 2.9‑6 Voltage Range Configuration for DTC Logging** * This voltage range **shall not** be merged into another, wider voltage range |

Table ‑ Voltage Range Configuration for DTC Logging

| **Rqmt. No.** | **Dataflow/Other** | **Value** | **Description** | **Index i** |
| --- | --- | --- | --- | --- |
| R: 2.9.38 | VMin\_Cfg[VR\_100\_155\_VB], VMinRcv\_Cfg[VR\_100\_155\_VB] | 10.0 volts | Minimum/Maximum voltage for DTC reporting | I:1 |
| R: 2.9.39 | VMax\_Cfg[VR\_100\_155\_VB], VMaxRcv\_Cfg[VR\_100\_155\_VB] | 15.5 volts |
| R: 2.9.40 | LoTime\_Cfg[VR\_100\_155\_VB] | 15 *m*sec | Recovery time to NORM\_V from LO\_V |
| R: 2.9.41 | HiTime\_Cfg[VR\_100\_155\_VB] | 15 *m*sec | Recovery time to NORM\_V from HI\_V |
| R: 2.9.42 | LoOvrTime\_Cfg[VR\_100\_155\_VB] | 500  *m*sec | LO\_V too long, enter UNDER\_V |
| R: 2.9.43 | HiOvrTime\_Cfg[VR\_100\_155\_VB] | 500  *m*sec | HI\_V too long, enter OVER\_V |
| R: 2.9.44 | LoRecovTime\_Cfg[VR\_100\_155\_VB] | 2,000 *m*sec | Recovery time to NORM\_V from UNDER\_V |
| R: 2.9.45 | HiRecovTime\_Cfg[VR\_100\_155\_VB] | 2,000 *m*sec | Recovery time to NORM\_V from OVER\_V |
| R: 2.9.46 | Use VBattState[VR\_100\_155\_VB] for compliance to Body Software Requirements #0021, #0064 and #0066 | | |

#### MS/HS CAN Interface (VR\_080\_160\_CAN)

|  |  |
| --- | --- |
| R: 2.9.47 | * Shown in the table below are the operating voltages that have been specified as critical for proper operation of MS and HS CAN interfaces and are to be monitored in dataflow VBattState[VR\_080\_160\_CAN]. * Reference **Figure 2.9‑2 Voltage Range Monitoring Finite State Machine** and * **Table 2.9‑7 Voltage Range Configuration for MS/HS CAN** * This voltage range **shall not** be merged into another, wider voltage range |

Table ‑ Voltage Range Configuration for MS/HS CAN

| **Rqmt. No.** | **Dataflow/Other** | **Value** | **Description** | **Index i** |
| --- | --- | --- | --- | --- |
| R: 2.9.48 | VMin\_Cfg[VR\_080\_160\_CAN], VMinRcv\_Cfg[VR\_080\_160\_CAN] | 8.0 volts | Inherits:   * worst-case feature voltage range in module * worst-case feature voltage range of shared, network inputs (not used by any class C features in your module, but is critical to another class C feature in another module and your module transmits this input on network).   (For Body features, voltage ranges are specified in ELCOMP SDS, Rqmt EC-0058 details.) | I:2 |
| R: 2.9.49 | VMax\_Cfg[VR\_080\_160\_CAN], VMaxRcv\_Cfg[VR\_080\_160\_CAN] | 16.0 volts |
| R: 2.9.50 | LoTime\_Cfg[VR\_080\_160\_CAN] | 15 *m*sec | Recovery time to NORM\_V from LO\_V |
| R: 2.9.51 | HiTime\_Cfg[VR\_080\_160\_CAN] | 15 *m*sec | Recovery time to NORM\_V from HI\_V |
| R: 2.9.52 | LoOvrTime\_Cfg[VR\_080\_160\_CAN] | 20  *m*sec | LO\_V too long, enter UNDER\_V |
| R: 2.9.53 | HiOvrTime\_Cfg[VR\_080\_160\_CAN] | 160  *m*sec | HI\_V too long, enter OVER\_V |
| R: 2.9.54 | LoRecovTime\_Cfg[VR\_080\_160\_CAN] | 200 *m*sec | Recovery time to NORM\_V from UNDER\_V |
| R: 2.9.55 | HiRecovTime\_Cfg[VR\_080\_160\_CAN] | 200 *m*sec | Recovery time to NORM\_V from OVER\_V |
| R: 2.9.56 | Use VBattState[VR\_080\_160\_CAN] for compliance to Body Software Requirements #0021, #0064 and #0066 | | |

### Shutdown Detection Voltage Range (VRange\_SDown)

|  |  |
| --- | --- |
| R: 2.9.57 | * Shown below in **Table 2.9‑8 Voltage Range Configuration for Low Voltage Shutdown Detection** are the Shutdown Detection and Recovery voltages and are to be monitored in dataflow VBattState[VRange\_SDown]. * Reference **Figure 2.9‑2 Voltage Range Monitoring Finite State Machine** * This voltage range **shall not** bemerged into another, wider voltage range. * Note: for faster shutdown detection, battery voltage may be sampled using a sampling period less than 5 *m*sec, but the number of samples and the timing must be approved by Ford. This value should be faster than the feature VRM sampling rate. |

Table ‑ Voltage Range Configuration for Low Voltage Shutdown Detection

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rqmt No.** | **Dataflow/Other** | **Value** | **Description** | |
| R: 2.9.58 | VMin\_Cfg[VRange\_SDown] | 6.0 volts | Low voltage shutdown for power dropout requirement in EMC spec. |
| R: 2.9.59 | VMinRcv\_Cfg[VRange\_SDown] | 8.0 volts | Low voltage shutdown voltage hysteresis for recovery from UNDER\_V |
| R: 2.9.60 | VMax\_Cfg[VRange\_SDown], VMaxRcv\_Cfg[VRange\_SDown] | 25.0 volts | No need for high voltage shutdown |
| R: 2.9.61 | LoTime\_Cfg[VRange\_SDown] | 0 *m*sec | Recovery time to NORM\_V from LO\_V (1 sample) |
| R: 2.9.62 | HiTime\_Cfg[VRange\_SDown] | 0 *m*sec | Recovery time to NORM\_V from HI\_V (1 sample) |
| R: 2.9.63 | LoOvrTime\_Cfg[VRange\_SDown] | 10 *m*sec | LO\_V too long, enter UNDER\_V (3 samples min) |
| R: 2.9.64 | HiOvrTime\_Cfg[VRange\_SDown] | 160 *m*sec | HI\_V too long, enter OVER\_V |
| R: 2.9.65 | LoRecovTime\_Cfg[VRange\_SDown] | 10  *m*sec | Recovery time to NORM\_V from UNDER\_V (3 samples min) |
| R: 2.9.66 | HiRecovTime\_Cfg[VRange\_SDown] | 2,000 *m*sec | Recovery time to NORM\_V from OVER\_V |

### Summary of Voltage Range Monitors

The following table defines each voltage range supported by the AVAS module VRM.

Table ‑ Index “I” for VBattState

|  |  |  |  |
| --- | --- | --- | --- |
| **Voltage Range** | **Index 'i' for VBattState in** Error! Reference source not found. | | **Feature** |
| VR\_080\_160\_VB | I:0 | | Voltage Range to help determine AVAS state | |
| VR\_100\_155\_VB | I:1 | | Voltage Range to determine capability of setting DTCs | |
| VR\_080\_160\_CAN | I:2 | | Voltage Range to determine capability of MS/HS CAN Interface | |
| VRange\_SDown | I:3 | | System Shutdown Voltage Range | |

## Diagnostics

### Overview

References to ISO 14229 and Ford Generic Global Diagnostic Specification (GGDS) are used throughout this section. These documents are to be referenced for definitions of diagnostic services, sub-functions, data-parameters, general parameter definitions, response codes, etc.

### Supported Diagnostic Identifiers

Table ‑ General Diagnostic Requirements

|  |  |
| --- | --- |
| **ECU Information** | |
| ECU Name: | Pedestrian Alert Control Module |
| ECU Acronym: | PACM |
| Diagnostic Communication Link: | HS-CAN |
| ECU Diagnostic Reception ID: | 750 |
| ECU Diagnostic Transmission ID: | 758 |
| Functional Diagnostic ID: | 7DF |
| Network Initialization ID: |  |
| **Specification Versions** | |
| GGDS Specification Version: | 003 |
| SWDL Specification Version: | 005 |
| ECU Configuration Version: | 003 |
|  |  |
| **Miscellaneous** | |
| Bootloader Supported | Yes |

### Supported Security Levels

#### Security Level (0x01) Specific Information

Table ‑ Security Level (0x01) Specific Information

|  |  |
| --- | --- |
| Detailed Information for Security Level 0x01 | |
| Security Level Number | 0x01 |
| Security Level Name | Programming Security Level |
| Security Level Usage Description | Used to unlock capability of flash programming |
| Security Level Entry Criteria | Valid seed/key sequence for security level 0x01 |
| Security Level Exit Criteria | Transition to same diagnostic session (via service 0x10), transition to a different diagnostic session, or transition to a different security level via a valid seed / key sequence |
| Sessions Supported In | 0x02 |
| Number of False Access Attempts | 0 |
| False Access Delay Time | 0ms |

#### Security Level (0x03) Specific Information

Table ‑ Security Level (0x03) Specific Information

|  |  |
| --- | --- |
| Detailed Information for Security Level 0x03 | |
| Security Level Number (request seed SF) | 0x03 |
| Security Level Name | WriteDataByIdentifier Security Level |
| Security Level Usage Description | Used to unlock capability to write data by identifier |
| Security Level Entry Criteria | Valid seed/key sequence for security level 0x03 |
| Security Level Exit Criteria | Transition to same diagnostic session (via service 0x10), transition to a different diagnostic session, or transition to a different security level via a valid seed / key sequence |
| Sessions Supported In | 0x03 |
| Number of False Access Attempts | 0 |
| False Access Delay Time | 0ms |
| Session 0x03 security protected services | 0x2E |

### AVAS Supported Diagnostic Services

Table ‑ Table of Supported Diagnostic Services

| **Rqmt. No.** | **Services** | **Functionality** | **Default ($01)** | **Program ($02)** | **Extend ($03)** |
| --- | --- | --- | --- | --- | --- |
| R: 2.10.1 | $10 | DiagnosticSessionControl  ($01- Default, $02- Programming, $03- Extended) | Y | Y | Y |
| R: 2.10.2 | $11 | ECUReset | Y | Y | Y |
| R: 2.10.3 | $14 | ClearDiagnosticInformation | Y | - | Y |
| R: 2.10.4 | $19 | Read DTCInformation | Y | - | Y |
| R: 2.10.5 | $22 | ReadDataByIdentifier | Y | Y | Y |
| R: 2.10.6 | $27 | SecurityAccess | - | Y | Y |
| **R: 2.10.7** | $2E | WriteDataByIdentifier (DID/Method 2) | Y | - | Y |
| R: 2.10.8 | $31 | RoutineControl | - | Y | Y |
| R: 2.10.9 | $34 | PBL: RequestDownload (to RAM) SBL: RequestDownload (to Flash) | - | Y | - |
| R: 2.10.10 | $36 | TransferData | - | Y | - |
| R: 2.10.11 | $37 | RequestTransferExit | - | Y | - |
| R: 2.10.12 | $3E | TesterPresent PBL: TesterPresent | Y | Y | Y |
| R: 2.10.13 | $85 | ControlDTCSetting (turn off missing msg DTCs) | - | - | Y |

### AVAS Diagnostic Session Control (0x10) Service

#### Supported Diagnostic Session Control



Figure ‑ Diagnostic Session Control Context Diagram

#### Preconditions for a Diagnostic Session Change (Service 0x10)

The system starts in the DEFAULT session. Sessions can be changed but only when the following conditions are satisfied.

Table ‑ Diagnostic Session Change Preconditions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Rqmt. No.** | **VBattState [VR\_Diagnostics]** | **Vehicle\_Speed** | **SelfTest** | **Ignition\_ Status** | **ChangeSessionPreCond** |
| R: 2.10.14 | UNDER\_V | Don't care | Don't care | Don't care | NOT\_OK |
| R: 2.10.15 | <> UNDER\_V | > DiagOutputCtrlSpeedLimit\_Cfg | Don't care | Don’t care | NOT\_OK |
| R: 2.10.16 | <> UNDER\_V | <= DiagOutputCtrlSpeedLimit\_Cfg | Don't care | <> RUN | NOT\_OK |
| R: 2.10.17 | <> UNDER\_V | <= DiagOutputCtrlSpeedLimit\_Cfg | TEST | RUN | NOT\_OK |
| R: 2.10.18 | <> UNDER\_V | <= DiagOutputCtrlSpeedLimit\_Cfg | NULL | RUN | OK |

Note: Due to issues with service charging a dead battery at 16+ volts and then not getting the ECU(CPDB) to enter EXTEND Session, we are modifying this preconditions to allow OVER\_V to enter EXTEND Session, but we will not allow SelfTest or I/O control (read/writeDIDs and other execution routines are ok).

**Table 2.10‑5a OTA Diagnostic Session Change Preconditions**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Rqmt. No.** | **VBattState [VR\_Diagnostics]** | **Vehicle\_Speed** | **SelfTest** | **Ignition\_ Status** | **Veh\_RunStart\_Source** | **OTAChangeSessionPreCond** |
| R: 2.10.18a | UNDER\_V | Don't care | Don't care | Don't care | Don’t Care | NOT\_OK |
| R: 2.10.18b | <> UNDER\_V | > DiagOutputCtrlSpeedLimit\_Cfg | Don't care | Don’t care | Don’t Care | NOT\_OK |
| R: 2.10.18c | <> UNDER\_V | <= DiagOutputCtrlSpeedLimit\_Cfg | Don't care | <> OFF | Don’t Care | NOT\_OK |
| R: 2.10.18d | <> UNDER\_V | <= DiagOutputCtrlSpeedLimit\_Cfg | TEST | OFF | Don’t Care | NOT\_OK |
| R: 2.10.18e | <> UNDER\_V | <= DiagOutputCtrlSpeedLimit\_Cfg | NULL | OFF | < > OTA | NOT\_OK |
| R: 2.10.18f | <> UNDER\_V | <= DiagOutputCtrlSpeedLimit\_Cfg | NULL | OFF | OTA | OK |

#### Extended Session Exit Conditions

The following conditions will force an exit from an extended session (indicated by the value of exit in the exitSessCond.)

Table ‑ Diagnostic Session Change Preconditions (leaving 0x01 to 0x02 or 0x03)

|  |  |  |  |
| --- | --- | --- | --- |
| **Rqmt. No.** | **Ignition\_Status** | **Vehicle\_Speed** | **ExitSessCond** |
| **R: 2.10.19** | RUN | > DiagOutputCtrlSpeedLimit\_Cfg | EXIT |
| **R: 2.10.20** | <> RUN | Don’t care | EXIT |
| **R: 2.10.21** | RUN | <= DiagOutputCtrlSpeedLimit\_Cfg | NULL |

#### Diagnostic Session Control State Transition Diagram



Figure ‑ Diagnostic Session Control Sate Transition Diagram

Note: It is left to the supplier if transitions 1->2.1 and 1->2.2 are necessary or need to be modified.

**Figure 2.10‑2 Diagnostic Session Control Sate Transition Diagram** is used only for reference and it is not necessary to be synchronized with **Table 2.10‑7 Diagnostic Session Control Requirements**. In case of any discrepancy consider **Table 2.10‑7 Diagnostic Session Control Requirements** as the final requirement.

Table ‑ Diagnostic Session Control Requirements

| **Rqmt. No.** | **Current Diag (psuedo)Session** | **Condition Additional Conditions** | **Other Actions** | **Next Diag (pseudo)Session** |
| --- | --- | --- | --- | --- |
| R: 2.10.22 | Boot - PBL | NoValidApp & delay elapses  & GotoPRGM\_Flag = NULL | BloaderPowerBusOff()  DiagSession =DEFAULT | PseudoNORMAL – PBL |
| **R: 2.10.23** | DiagTool request for PROGRAM sessions | GotoPRGM\_Flag = NULL  BloaderPowerBusOn()  *Turn Off all non-PowerBus outputs*  DiagSession = PROGRAM | PROGRAM – PBL |
| **R: 2.10.24** | NoValidApp & delay elapses  & GotoPRGM\_Flag = PROGRAM | GotoPRGM\_Flag = NULL  BloaderPowerBusOn()  *Turn Off all non-PowerBus outputs*  DiagSession = PROGRAM | PROGRAM – PBL |
| **R: 2.10.25** | ValidApp & delay elapses  & GotoPRGM\_Flag = PROGRAM | GotoPRGM\_Flag = NULL  BloaderPowerBusOn()  *Turn Off all non-PowerBus outputs*  DiagSession = PROGRAM | PROGRAM – PBL |
| **R: 2.10.26** | ValidApp & delay elapses  & GotoPRGM\_Flag = NULL | *This shall happen at the beginning of application*  UpdateEEPROMmap()  UpdateResetCauseDID()  DiagSession = DEFAULT | DEFAULT - Strategy |
| **R: 2.10.27** | Does not respond to DiagTool reset request | | |
| **R: 2.10.28** | No other transitions are supported | | |
|  | | | | |
| **R: 2.10.29** | PseudoNORMAL – PBL | DiagTool request for PROGRAM sessions | BloaderPowerBusOn()  DiagSession = PROGRAM | PROGRAM – PBL |
| **R: 2.10.30** | DiagTool Reset request | GotoPRGM\_Flag = NULL  ForceReset() | Boot – PBL |
| **R: 2.10.31** | No other exit transitions are supported | | |
|  |  |  | | |
| **R: 2.10.32** | PROGRAM - PBL | DiagTool Reset request | GotoPRGM\_Flag = NULL  ForceReset() | Boot – PBL |
| **R: 2.10.33** | Servicing of Change Session requests only occurs after Flash writes and EEPROM updates have completed | | |
| **R: 2.10.34** | No other exit transitions are supported | | |
|  |  |  | | |
| **Note:** AVAS does not act as a power bus for any other ECUs and directives for BloaderPowerBusOn() and BloaderPowerBusOff() maybe be ignored. | | | | |
| **R: 2.10.35** | DEFAULT - Strategy | DiagTool request for PROGRAM session  & (ChangeSessionPreCond = OK | (Veh\_Start\_Inhibit = TRUE & DID $D04F preconditions are set to 0 & otaChangedSessionPreCond = OK)) | WaitEEPROMUpdate()  BloaderPowerBusOn()  DiagSession = PROGRAM  *Turn Off all non-PowerBus outputs*  *Stop all application processing* | PROGRAM – PBL |
| **R: 2.10.36** | DiagTool request for EXTEND session  & (ChangeSessionPreCond = OK |  OTAChangeSessionPreCond = OK) | ST\_Results = NONE  SelfTest = NULL  DiagSession = EXTENDED | EXTEND - Strategy |
| **R: 2.10.37** | DiagTool Reset request  & ExitSessCond <> EXIT | GotoPRGM\_Flag = NULL  WaitEEPROMUpdate()  *Stop all application processing*  ForceReset() | Boot – PBL |
| **R: 2.10.38** | DiagTool Reset request  & ExitSessCond = EXIT | Reply ConditionsNotCorrect | DEFAULT - Strategy |
| **R: 2.10.39** | DiagTool request for EXTEND session & ChangeSessionPreCond = NOT\_OK | Reply ConditionsNotCorrect | DEFAULT - Strategy |
| **R: 2.10.40** | Periodically calls Clear\_PIDCtrlSignals() every PIDClear\_Time\_Cfg (5 seconds or so) | | |
| **R: 2.10.41** | No other exit transitions are supported | | |
|  |  |  | | |
| **R: 2.10.42** | EXTEND - Strategy | DiagTool request for DEFAULT session  & ChangeSessionPreCond = OK | GotoPRGM\_Flag = NULL  WaitEEPROMUpdate()  ForceReset() | Boot – PBL (DEFAULT) |
| **R: 2.10.43** | DiagTool request for PROGRAM session  & (ChangeSessionPreCond = OK | (Veh\_Start\_Inhibit = TRUE  & DID $D04F preconditions are set to 0 & OTAChangeSessionPreCond = OK)) | GotoPRGM\_Flag = PROGRAM  WaitEEPROMUpdate()  ForceReset() | Boot – PBL (PROGRAM) |
| **R: 2.10.44** | DiagTool Reset request | GotoPRGM\_Flag = NULL  WaitEEPROMUpdate()  ForceReset() | Boot – PBL |
| **R: 2.10.45** | ExitSessCond = EXIT | GotoPRGM\_Flag = NULL  WaitEEPROMUpdate()  ForceReset() | Boot – PBL (DEFAULT) |
| **R: 2.10.46** | No other exit transitions are supported | | |
|  |  |  | | |
| **R: 2.10.47** | The Session Manager in the Strategy (not bootloader) shall publish these dataflows (at a minimum) for use by any process in the Strategy:   * SecurityLevel * DiagSession | | | |
| **Note 1:** AVAS does not act as a power bus for any other ECUs and directives for BloaderPowerBusOn() and BloaderPowerBusOff() maybe be ignored. | | | | |
| **Note** 2: DID $D04F preconditions byte 1, bit 7 = 0 indicates preconditions supported. The AVAS module checks vehicle speed, ignition status, and self -test status in **Table 2.10‑5a OTA Diagnostic Session Change Preconditions** | | | | |

##### WaitEEPROMUpdate() Function

Table ‑ WaitEEPROMUpdate() Function Requirements

|  |  |
| --- | --- |
| **Rqmt. No.** | **Requirements** |
| R: 2.10.48 | This function shall:   * Set NVRAM\_Rqst = UPDATE   + - * + Wait for EEPROM to finish updating before returning         + Or – give up after a short time and return (due to the voltage being too low for too long) * Set NVRAM\_Status = DONE |

##### ForceReset() Function

Table ‑ ForceReset() Function Requirements

|  |  |
| --- | --- |
| **Rqmt. No.** | **Requirements** |
| R: 2.10.49 | This function shall:   * Wait a configurable (DiagResetDelay\_Cfg) amount of time to allow other ECUs to complete EEPROM writes * Set the “Diag Forced Reset” flag * Force the microprocess or reset line to assert   **Note:** It is better to start from a known hardware and software point. |

##### UpdateEEPROMmap() Function

Table ‑ Update EEPROMMap() Function Requirements

|  |  |
| --- | --- |
| **Rqmt. No.** | **Requirements** |
| R: 2.10.50 | This function shall:   * Compare *Existing EEPROM map* version with the *Required EEPROM map* version * If the *Required EEPROM map* version is incompatible with the *Existing EEPROM map* version then this function shall create any missing elements.   **Note:** The allowed strategies for this are defined in the Software SOW package (Non-Volatile Memory.docx file) |

##### UpdateResetCauseDID() Function

Table ‑ UpdateResetCauseDID() Function Requirements

| **Rqmt. No.** | **Requirements** | |
| --- | --- | --- |
| R: 2.10.51 | This function shall:   * Update the reset reason flags/counts in DID D700 and D701 * “Forced software reset” (e.g. Diag tool reset, extending from extended session to programming session) shall not be counted as watch dog rest, power on reset. * “Diag Forced Reset” flag is used to determine force rest * Clear these flags so they don’t get counted again on the next reset |

#### Diagnostic Timer Monitor

Now as part of FNOS, all diag services that require the tester to be present, are now run from within Extended Session.

### AVAS Routine Control (0x31) Service

#### Supported Functional Execution Routines

Control routines are executable routines resident within an ECU that may be invoked by a tester to evaluate the ECU and its associated components or to execute specific functionality. Refer to section 4.3 Control routine in Ford's Generic Global Diagnostic Specification for generic Routine implementation requirements.

CGEA 1.3 AVAS supports the following Execution Routines in Service #31.

Table ‑ Supported Functional Execution Routine Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Rqmt. No.** | **Routine Identifier** | **Routine Type** | **Functionality** | **Default**  **($01)** | **Program**  **(0x02)** | **Extended**  **(0x03)** |
| R: 2.10.52 | $0301 | 1 | Activate Secondary Boot-loader | - | Y | - |
| R: 2.10.53 | $0304 | 1 | Check Valid Application | - | Y | - |
| R: 2.10.54 | $FF00 | 1 | Flash Erase | - | Y | - |
| R: 2.10.55 | $FF01 | 1 | Check Programming Dependencies | - | Y | - |
| R: 2.10.56 | $0202 | 2 | On Demand Self-test | - | - | Y |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| R: 2.10.56a | TBD | 1 | Control Routine | - | - | Y |

|  |  |
| --- | --- |
| **Rqm’t Num.** | **Requirement** |
| **R: 2.10.56b** | The ECU may allow required Control Routines (Service 31) to be executed when ignition is OFF and VehOnSrc\_D\_Stat = OverTheAir. This may include any routines that are needed to run after an OTA SW re-flash. |

#### Type 1 Routines

Requirements for supported type 1 routines are defined in the NETCOM Statement of Work (SOW).

#### Type 2 Routines

##### On-Demand Self-Test ($0202)

###### Overview

On Demand Self-Test is a diagnostic execution routine in service $31 with Routine Identified $0202. This routine shall detect and identify the ECU's I/O circuit faults. Upon completion of the test, the outputs shall return to normal ECU control. The routines return a special case of routine results with a data type of DTC. The DTC data type reports a list of repeating 3 byte values referred to as On-Demand DTCs. Refer to section 4.3.3 Fault Detection Control Routines in Ford's Generic Global Diagnostic Specification.

The AVAS ECU should complete its on-demand self-test within maximum of 5000 msec after receiving the request to perform the test.



Figure ‑ Self-Test Control Context Diagram

Table ‑ Voltage Requirements and Ignition Status Requirements for Self-Test

|  |  |  |  |
| --- | --- | --- | --- |
| Rqmt. No. | VBattState[VR\_100\_155\_VB] | Ignition\_Status | Okay to run self-test? |
| **R: 2.10.57** | UNDER\_V | OVER\_V | Don’t Care | No | |
| **R: 2.10.58** | NORM\_V | LO\_V | HI\_V | <> RUN | No | |
| **R: 2.10.59** | NORM\_V | LO\_V | HI\_V | RUN | Yes | |

NOTE: Vehicle Speed is not considered as a pre-condition for Self-Test because it has already been considered as a pre-condition for being in Extended Session.

**Table 2.10‑14 Criteria for Entering Self-Test**

|  |  |
| --- | --- |
| **Rqmt. No.** | **Description** |
| **R: 2.10.60** | Okay to run self-test is “Yes” as in **Table 2.10‑13 Voltage Requirements and Ignition Status Requirements for Self-Test** |
| **R: 2.10.61** | Extended Session |
| **R: 2.10.62** | On-Demand is requested by Tester |

Table ‑ Criteria for Existing Self-Test

|  |  |
| --- | --- |
| **Rqmt. No.** | **Description** |
| **R: 2.10.63** | Okay to run self-test is “No” as in **Table 2.10‑13 Voltage Requirements and Ignition Status Requirements for Self-Test** |
| **R: 2.10.64** | A stop routine command is issued. |
| **R: 2.10.65** | Tester does NOT communicate for more than five (5) seconds. |
| **R: 2.10.66** | Test is complete. |

###### Self-Test Output Activation

**Note 1:** It is left to the supplier to define what is tested when and which outputs are activated in what order. Ford requires the supplier to provide a description of the self-test in the Part II Diagnostic Spec.

Table ‑ Self-Test Output Activation

| **Rqmt No.** | **Requirement** |
| --- | --- |
| R: 2.10.67 | Output activation must be staggered by at least 50 *m*sec between sets of 10 amp loads. |
| R: 2.10.68 | It is the supplier’s responsibility to avoid causing DTCs due to interactions between inputs and outputs assuming there are no power closure products (power lift gate, power sliding door…) on the vehicle. This means that a normal lift gate might open when the power lift gate release is activated. |
| R: 2.10.69 | General SelfTest Sequence:   * First turn all outputs off and wait long enough to detect short circuits * Then turn on all outputs and wait long enough to ensure that any short is detected. |
| R: 2.10.70 | Use PIDCtrl dataflows to control the outputs – even though these may interfere with I/O control currently being used. |
| R: 2.10.71 | After completion of Self-test set all PIDCtrl dataflows to NULL |

###### Self-Test Input Testing

**Note 1**: It is left to the supplier when any input validation testing is performed during the self-test sequence. Ford requires the supplier to provide a description of the self-test in the Part II Diagnostic Spec.

| **Rqmt No.** | **Requirement** |
| --- | --- |
| R: 2.10.72 | Any supplemental diagnostic circuit to test the validity of input hardware shall be used either before or after all output testing. |
| R: 2.10.73 | General SelfTest Sequence:   * Activate input diagnostic circuit, * wait long enough for the circuit to stabilize * Then sample input circuit to test for valid range |

#### Type 3 Routines

No Type 3 control routines are supported by this ECU.

### AVAS Data Identifiers (DIDs)

#### ReadDataByIdentifer (0x22) Service

**Note:** Supplier DIDs are not in the Ford GMRDB. Each supplier range DID was assigned by picking a unique ID number within the range of $FD00 - $FEFF.

##### Miscellaneous Service $22 Requirements

Table ‑ Miscellaneous Data Identifier Requirements

|  |  |
| --- | --- |
| **Rqmt No.** | **Requirement** |
| **R: 2.10.74** | To all DID requests for DIDs that have been configured OFF/DISABLED reply RequestOutOfRange | |
| **R: 2.10.75** | For bit-mapped or packeted DIDs, all fields shall be set to 0 for any datavalue that is disabled via configuration. | |

##### Supported DID List

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table 2.10‑18 Supported DID List** | | | | | | | | | | **Diag Session** | | | |
| **Rqmt. No.** | **DID** | | **DID Name/Description** | | | | **Config Reqts** | **Dataflow** | **01** | | **02** | **03** |
| **R: 2.10.76** | 0x0202 | | Number of Trouble Codes Set due to Diagnostic Test | | | | n/a | Supplier Defined | R | | - | R |
| **R: 2.10.77** | 0xD100 | | Active Diagnostic Session | | | | n/a | DiagSession | R | | R | R |
| **R: 2.10.78** | 0xF110 | | On-Line Diagnostic Database Reference Number | | | | n/a | Supplier Defined | R | | - | R |
| **R: 2.10.79** | 0xF111 | | ECU Core Assembly P/N | | | | n/a | Supplier Defined | R | | R | R |
| **R: 2.10.80** | 0xF113 | | ECU Delivery Assembly P/N | | | | n/a | Supplier Defined | R | | R | R |
| **R: 2.10.81** | 0xF17C | | NOS Bootloader Generation Tool Version Number | | | | n/a | Supplier Defined | R | | R | R |
| **R: 2.10.82** | 0xF15F | | NOS Generation Tool Version Number | | | | n/a | Supplier Defined | R | | - | R |
| **R: 2.10.83** | 0xF160 | | NOS Diagnostic Version Number | | | | n/a | Supplier Defined | R | | - | R |
| **R: 2.10.84** | 0xF161 | | NOS CAN Communication Layer Version Number | | | | n/a | Supplier Defined | R | | - | R |
| **R: 2.10.85** | 0xF162 | | Software Download Specification Version | | | | n/a | Supplier Defined | - | | R | - |
| **R: 2.10.86** | 0xF163 | | Diagnostic Specification Version | | | | n/a | Supplier Defined | R | | - | R |
| **R: 2.10.87** | 0xF166 | | NOS Message Database #1 Version Number | | | | n/a | Supplier Defined | R | | - | R |
| **R: 2.10.88** | 0xF170 | | NOS Bootloader Package Version Number | | | | n/a | Supplier Defined | - | | R | - |
| **R: 2.10.89** | 0xF171 | | NOS Bootloader Main Version Number | | | | n/a | Supplier Defined | - | | R | - |
| **R: 2.10.90** | 0xF172 | | NOS Bootloader Diagnostic Version Number | | | | n/a | Supplier Defined | - | | R | - |
| **R: 2.10.91** | 0xF173 | | NOS Bootloader Network/Transport Layer Version Number | | | | n/a | Supplier Defined | - | | R | - |
| **R: 2.10.92** | 0xF174 | | NOS Bootloader Flash Routines Version Number | | | | n/a | Supplier Defined | - | | R | - |
| **R: 2.10.93** | 0xF175 | | NOS Bootloader Hardware File Version Number | | | | n/a | Supplier Defined | - | | R | - |
| **R: 2.10.94** | 0xF176 | | NOS Bootloader API Version Number | | | | n/a | Supplier Defined | - | | R | - |
| **R: 2.10.95** | 0xF177 | | NOS Bootloader Security Algorithm Version Number | | | | n/a | Supplier Defined | - | | R | - |
| **R: 2.10.96** | 0xF178 | | NOS Bootloader Flash I/O Version Number | | | | n/a | Supplier Defined | - | | R | - |
| **R: 2.10.97** | 0xF17B | | NOS Bootloader Memory I/O Version Number | | | | n/a | Supplier Defined | - | | R | - |
| **R: 2.10.98** | 0xF188 | | Vehicle Manufacturer ECU Software Number | | | | n/a | Supplier Defined | R | | - | R |
| **R: 2.10.99** | 0xD111 | | ECU Power Supply Voltage | | | | n/a | ECU Power Supply Voltage | R | | - | R |
| **R: 2.10.100** | 0xF180 | | Boot Software Identification | | | | n/a | Supplier Defined | R | | R | R |
| **R: 2.10.101** | 0xF10A | | ECU Cal-Config Part Number | | | | n/a | Supplier Defined | R | | - | R |
| **R: 2.10.102** | 0xF18C | | ECU Serial Number | | | | n/a | Supplier Defined | R | | R | R |
| **R: 2.10.103** | 0x0599 | | PowerPack State | | | | n/a | PowerPack State | R | | - | R |
| **R: 2.10.104** | 0x40B5 | | Ignition Position Final Status | | | | n/a | Ignition Position Final Status | R | | - | R |
| **R: 2.10.105** | 0x7218 | | Reverse Gear Position | | | | n/a | Reverse Gear Position | R | | - | R |
| **R: 2.10.106** | 0xD115 | | Vehicle Speed | | | | n/a | Vehicle Speed | R | | - | R |
| **R: 2.10.107** | 0x0130 | | Transmission Shift Lever Position | | | | n/a | Transmission Shift Lever Position | R | | - | R |
| **R: 2.10.108** | 0xF124 | | ECU Calibration Data #1 Number | | | | n/a | Supplier Defined | - | | - | R |
| **R: 2.10.109** | 0xF125 | | ECU Calibration Data #2 Number | | | | n/a | Supplier Defined | - | | - | R |
| **R: 2.10.110** | 0xD700 | | Critical Software Parameter Monitoring #1 | | | | n/a | Supplier Defined | R | | - | RW |
| **R: 2.10.111** | 0xD701 | | Critical Software Parameter Monitoring #2 | | | | n/a | Supplier Defined | R | | - | RW |
| **R: 2.10.111a** | 0xD028 | | Application Signature Validation Status | | | | n/a | Supplier Defined | R | | - | R |
| **R:2.10.111b** | 0xD03F | | In-Use Application Signing Public Key Hash | | | | n/a | Supplier Defined | - | | - | R |
| **NOTE 1**: DIDs that support diagnosing SW signing need to be available when the ECU is in Bootloader and Application  **NOTE 2**: If an ECU consists of multiple VBFs, report information on the first VBF that fails SW signing. Otherwise if SW signing passes, report information on the last VBF downloaded or the main application VBF. | | | | | | | | | | | | |
| **R:2.10.111c** | | 0xD04F | | OTA Programming Session Entry and A/B Swap Precondition Status | n/a | Byte 1, bit 0 - ESCL Lock Pending  Byte 2, bit 0 – Park Lamps On  Byte 1, bit 1 - Ignition Off / Acc Functionality Active  Byte 2, bit 1 - Liftgate Ajar  Byte 1, bit 2 - Hazards On  Byte 2, bit 2 - Ignition Status Out of Range  Byte 1, bit 3 - PRNDL Out of Range  Byte 2, bit 3 -Charging Fault  Byte 1, bit 4 - Charging in Progress  Byte 3, bit 2 - Park Brake Out of Range or Activation in Progress  Byte 2, bit 4 - Engine RPM Too High (or Torque Available)  Byte 1, bit 5 - Voltage Out of Range  Byte 3, bit 3 - Brake Pedal Pressed  Byte 2, bit 5 - Diagnostic Self-Test Active  Byte 1, bit 6 - Vehicle Speed Too High  Byte 3, bit 4 – Motor Movement Active  Byte 2, bit 6 - Steering Pinsion Torque Out of Range  Byte 1, bit 7 - No Preconditions Supported  Byte 3, bit 5 - Door Ajar  Byte 3, bit 6 - Illuminated Exit Active  Byte 2, bit 7 - Alarm Actively Sounding  Byte 3, bit 7 - Limp Home Active | | | R | | - | R |
| **Note on 0xD04F**: Any bits that are not applicable to the AVAS module may be set to 0. If any bits are applicable, then Byte 1, bit 7 must be set to 0 to indicate that Preconditions are supported by AVAS. | | | | | | | | | | | | |

**Note:** R – Supports Diagnostic Service $22 ReadDataByIdentifer; W – Supports Diagnostic Service $2E WriteDataByIdentifier

#### Writeable DIDs (service $2E)

##### General Writeable DID Requirements

|  |  |
| --- | --- |
| **Table 2.10‑19 General Writeable DID Requirements Table** | |
| Rqmt. No. | **Description** |
| R: 2.10.112 | All Values written to DIDs shall come into effect immediately. |

##### DID $D700 – Critical Parameters #1

Table ‑ – DID $D700 Writing Requirements

| Rqmt. No. | **DID $D700 Write Value** |  | **Result** | |
| --- | --- | --- | --- | --- |
| R: 2.10.113 | Zero (0) |  | DID $D700 Power on reset count = 0  DID $D700 Illegal op-code count = 0  DID $D700 Watchdog timer reset count = 0  DID $D700 Data range failure count = 0  Reply OK |
| R: 2.10.114 | Non-Zero |  | Reply RequestOutOfRange |
| R: 2.10.115 | When AVAS is configured and before delivery to Ford, clear the counts used by $D700 | | |

##### DID $D701 – Critical Parameters #2

Table ‑ – DID $D701 Writing Requirements

| Rqmt. No. | **DID $D701 Write Value** |  | **Result** | |
| --- | --- | --- | --- | --- |
| R: 2.10.116 | Zero (0) |  | DID $D701 Stack overflow count = 0  DID $D701 Loss of idle time count = 0  DID $D701Minimum idle time = 255  DID $D701 Recover from power dropout count= 0  Reply OK |
| R: 2.10.117 | Non-Zero |  | Reply RequestOutOfRange |
| R: 2.10.118 | When AVAS is configured and before delivery to Ford, clear the counts used by $D701 (**note:** minimum idle time set to 255)) | | |

#### Supplier Range DIDs

AVAS supports configuration of the module by the supplier through DIDs in the FD00-FEFF range. Configuration parameters in this range are not read or modified by Ford Motor Company in the factory at end-of-line nor during service. Ford relies on the supplier to correctly configure these parameters as described in **2.10.9.4AVAS Configuration Data (Supplier Range FD00-FEFF).**

### AVAS Diagnostic Trouble Codes (DTCs)

#### Diagnostic Trouble Codes General Requirements

A basic aim of diagnostics is to detect and report faults in an ECU, its peripherals, and the subsystem/vehicle. A Diagnostic Trouble Codes (DTC) is a 3-byte numerical identifier for a specific fault condition that can be identified by the ECU's on-board diagnostic system. The first two most significant bytes of a DTC are referred to as the root DTC. The least significant byte of a DTC is referred to as the Failure Type Byte. (e.g., Failure Type Byte $11 = "Circuit Short to Ground"). In general, the root DTC will not contain the failure type but will rather consist of the root component. However, when the root DTC description does already contain the failure type information, then the actual Failure Type Byte value shall be set to $00 indicating no additional failure type information.

Every DTC has certain status information associated with it which indicates information such as whether the fault detection test has completed during the current DTC operation cycle, whether the result of the test is a pass or fail, and whether or not a warning indicator is actively being illuminated for the DTC.

#### ControlDTCSetting (0x85) Service

The Test tool can request the ECU to suppress detection of all DTCs. This state machine responds to these requests and publishes the dataflow *DTC\_Ctrl* for the DTC detection routine to use.

This state machine also detects when the DTC $F00049 (ECU Fault) is set and responds by setting DTC\_Ctrl to IGNORE in order to minimize the number of preconditions that the DTCs need to evaluate before evaluating the status of a DTC.



Figure ‑ Control DTC Setting FSM

Table ‑ Control Setting DTC FSM Requirements

| **Rqmt. No.** | **Source State -> Destination State Rqmt. No.** |
| --- | --- |
| R: 2.10.119 | 0->1.1 |
| R: 2.10.120 | 0->2.1 |
| **R: 2.10.121** | 1->1.1 |
| **R: 2.10.122** | 1->1.2 |
| **R: 2.10.123** | 1->1.7 |
| **R: 2.10.124** | 1->1.6 |
| **R: 2.10.125** | 1->2.1 |
| **R: 2.10.126** | 2->2.1 |
| **R: 2.10.127** | 2->2.7 |
| **R: 2.10.128** | 2->2.6 |
| **R: 2.10.129** | 2->1.1 |

**Note:** You can only use ControlDTCSetting in the EXTEND state. The EXTEND state always exits with a reset so there is no recovery from **Ignore** to **Watch** – we leave it to the reset. Also, we don't check to see if we are in EXTEND when we are in the **Ignore** state since the only way you can be in **Ignore** is to be in EXTEND.

#### DTCs for Hardware Outputs

Table ‑ Unique DTCs for Hardware Outputs Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| **Rqmt Num.** | **Driver Type** | **Unique DTC Requirement** | |
| R: 2.10.130 | High-Side (low or high current) | Open-circuit or Short-to-Battery |
| R: 2.10.131 | High-Side (low or high current) | Short-to-ground |
| R: 2.10.132 | Low-Side | Open-circuit / Short-to-ground |
| R: 2.10.133 | Low-Side | Short-to-VBatt |

#### Asynchronous Fault Detection

In order to avoid synchronization issues with DTC monitoring, detection of high-current faults (short circuits) and low-current faults (open circuit) will use a latched dataflow indicating the status of the circuit the last time the circuit was in a state that allowed fault detection by the ECU.

Table ‑ Asynchronous DTC Sampling Requirements

| **Rqmt. No.** | **Requirements** | |
| --- | --- | --- |
| **R: 2.10.134** | There shall be a high-current fault status dataflow indicating if a high current fault exists  (based on the last time the circuit was in a state  where the ECU can detect high-current faults) |
| **R: 2.10.135** | There shall be a low-current fault status dataflow indicating if a low current fault exists  (based on the last time the circuit was in a state  where the ECU can detect low-current faults) |

**Note:** when both the high-current & low-current fault dataflows indicate no problem, the circuit is operating normally.

#### Control SettingAging of DTCs Finite State Machine

The following state machine controls when each DTC can be sampled, whether it can be evaluated or not at this time and its detection criteria using the requirements listed in Section **2.10.8.7 DTC Definitions and Requirements Table**. If the detection criteria indicate there may be a DTC, a unique 8-bit signed counter is incremented by a specified value. If the criteria indicate there isn't a DTC, the counter is decremented by a specified value. When the MAX value is reached, a DTC will be set. When the MIN value is reached, the DTC will be aged at the end of the monitoring cycle (as long as DTC isn't later detected).

General Information:

* All hardware-based DTCs require multiple (at least 4 – preferably more) samples indicating a fault exits before a DTC is set.
* “Functional” DTC (DTCs requested due to functional processes of the feature) will usually jump right to the MAX value in one step. Functional DTCs include: Missing Vehicle Speed Message, RKE Transmitter out-of-sync…

Incrementing the counter always sets a negative counter to 0 before incrementing it, while decrementing a counter never zeros the counter first.



Figure ‑ Control Setting/Aging of DTCs State Transition Diagram Requirements

Table ‑ Control Setting/Aging of DTCs State Transition Diagram Requirements

| **Rqmt. No.** | **Source State -> Destination State** |
| --- | --- |
| R: 2.10.136 | 0->1.1 |
| R: 2.10.137 | 1->2.1 |
| R: 2.10.138 | 2->1.1 |
| R: 2.10.139 | 2->3.1 |
| R: 2.10.140 | 3->1.1 |
| R: 2.10.141 | 3->4.1 |
| R: 2.10.142 | 3->4.2 |
| R: 2.10.143 | 4->1.1 |
| R: 2.10.144 | 4->1.2 |
| R: 2.10.145 | 4->1.3 |

**Functional DTCs**

“Functional” DTCs (DTCs set by feature processing) normally do not have a method for clearing them – they just age until gone (or service 14). To avoid EEPROM wear-out, DTC aging occurs at the end of the monitoring cycle (start of a new cycle). A new cycle is defined to start on a RUN to OFF ignition transition.

The standard interface for “Functional” DTCs is a dataflow that is SET by the requesting feature and cleared by the Diagnostic DTC routines. If the dataflow is unset at the end of a monitoring cycle, the associated Functional DTC will be aged.

#### DTC Definitions

These notes apply to the *DTC Definition Tables* below.

|  |  |
| --- | --- |
| **General Note** | SelfTest DTCs must use/modify the structure DTCSTest[n]. Continuous DTCs must use/modify the structure DTCRun[n]. The definitions below generally modify the .count and the .monitorMin/ .monitorMax values in both these structure. If a DTC is self-test only, then it doesn’t need to report on a Continuous DTC.     It is nice to have a Continuous DTC set when the matching SelfTest fails – that is what we are attempting to do for this ECU. This is where we call out specifically a call to DTCMaxAction() which allocates NVM for the Continuous DTC.   SelfTest DTCs are only stored in RAM  Continuous DTCs are stored in EEPROM  Refer ECU Software Req #0043 for Ignition switch position change and micro reset. |
| **Note 1** | VBattState[VR\_100\_155\_VB] = NORM\_V (according to ECU Software Rqmt #0064) |
| **Note 2** | Fault detection must use the appropriate value of FETOpenThreshold\_Cfg[FET\_Ctrl\_Index] or FETShortThreshold\_Cfg[FET\_Ctrl\_Index] determine the presence of a fault. Also, only increment or decrement the counter if the specific fault is detectable. |
| **Note 3** | Output high-current short DTCs are not limited by *VBattGuard*. We decided that if the FET was disabled and the lifecount incremented, we want to leave tracks about which circuit was affected. This is especially important when $F00049 is also set due to the lifecount. For consistency, the BJT outputs also follow this same rule even though the lifecount doesn't matter. |
| **Note 4** | Once $F00049 is set, do not set any other DTCs. Ensure that other pending DTCs are set before $F00049 (don't want to lose the reason $F00049 was set due to short circuit DTC). Possibly only evaluate DTCs for one more second. This is intended to limit using all DTC memory and causing EEPROM memory issues. |

Note: AVAS does not use FETs or $F00049 and as such some of these notes will not apply.

#### 

#### DTC Definitions and Requirements Table

| **Rqmt. No.** | **DTC** | **DTC Name** | **Config\_Reqts** | **Rate(msec)** | **Self-Test** | **VBatt Guard** | **Inc\_Criteria** | **Inc\_Val** | **Max Action** | **Dec\_Criteria** | **Dec\_Val** | **Min Action** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| R: 2.10.146 | 0x9A0101 | Speaker #1 General Electrical Failure | DTC\_Ctrl = WATCH &Supplier Defined | 200 | TEST | n/a | Speaker fails during self-test. | 33 | DTCSTest[].MonitorMAX=MAX | Speaker does not fail during self-test. | 33 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.147 | 0x9A0101 | Speaker #1 General Electrical Failure | DTC\_Ctrl = WATCH &Supplier Defined | 1000 | NULL | n/a | Speaker Failure during normal operation. | 33 | DTCMaxAction() | Speaker Failure during normal operation. | 33 | DTCMinAction() |
| R: 2.10.148 | 0x9a0111 | Speaker #1 Circuit Short To Ground | DTC\_Ctrl = WATCH &Supplier Defined | 200 | TEST | n/a | Test fails speaker short to ground. | 127 | DTCSTest[].MonitorMAX=MAX | Test Passes speaker not short to ground | 33 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.149 | 0x9a0111 | Speaker #1 Circuit Short To Ground | DTC\_Ctrl = WATCH &Supplier Defined | 1000 | NULL | n/a | Short to Ground detected during normal operation (Continuous DTC) | 127 | DTCMaxAction() | Short to Ground detected during normal operation (Continuous DTC) | 33 | DTCMinAction() |
| R: 2.10.150 | 0x9A0112 | Speaker #1 Circuit Short to Battery | DTC\_Ctrl = WATCH &Supplier Defined | 200 | TEST | n/a | Test fails speaker short to battery. | 33 | DTCSTest[].MonitorMAX=MAX | Test Passes speaker not short to battery | 33 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.151 | 0x9A0112 | Speaker #1 Circuit Short to Battery | DTC\_Ctrl = WATCH &Supplier Defined | 1000 | NULL | n/a | Short to battery detected during normal operation (Continuous DTC) | 33 | DTCMaxAction() | Short to battery not detected during normal operation (Continuous DTC) | 33 | DTCMinAction() |
| R: 2.10.152 | 0x9A0113 | Speaker #1 Circuit Open | DTC\_Ctrl = WATCH &Supplier Defined | 200 | TEST | n/a | Test fails speaker open circuit | 33 | DTCSTest[].MonitorMAX=MAX | Test Passes speaker not open circuit | 33 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.153 | 0x9A0113 | Speaker #1 Circuit Open | DTC\_Ctrl = WATCH &Supplier Defined | 1000 | NULL | n/a | Open Circuit Detected in normal operation (Continuous DTC) | 33 | DTCMaxAction() | Open Circuit not detected during normal operation (Continuous DTC) | 33 | DTCMinAction() |
| R: 2.10.154 | 0x9A0201 | Speaker #2 General Electrical Failure | AVAS\_Spkr\_Cfg = 0x0 & DTC\_Ctrl = WATCH &Supplier Defined | 200 | TEST | n/a | Speaker fails during self-test. | 33 | DTCSTest[].MonitorMAX=MAX | Speaker does not fail during self-test. | 33 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.155 | 0x9A0201 | Speaker #2 General Electrical Failure | AVAS\_Spkr\_Cfg = 0x0 & DTC\_Ctrl = WATCH &Supplier Defined | 1000 | NULL | n/a | Speaker Failure during normal operation. | 33 | DTCMaxAction() | Speaker Failure during normal operation. | 33 | DTCMinAction() |
| R: 2.10.156 | 0x9A0211 | Speaker #2 Circuit Short To Ground | AVAS\_Spkr\_Cfg = 0x0 & DTC\_Ctrl = WATCH &Supplier Defined | 200 | TEST | n/a | Test fails speaker short to ground. | 127 | DTCSTest[].MonitorMAX=MAX | Test Passes speaker not short to ground | 33 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.157 | 0x9A0211 | Speaker #2 Circuit Short To Ground | AVAS\_Spkr\_Cfg = 0x0 & DTC\_Ctrl = WATCH &Supplier Defined | 1000 | NULL | n/a | Short to Ground detected during normal operation (Continuous DTC) | 127 | DTCMaxAction() | Short to Ground detected during normal operation (Continuous DTC) | 33 | DTCMinAction() |
| R: 2.10.158 | 0x9A0212 | Speaker #2 Circuit Short to Battery | AVAS\_Spkr\_Cfg = 0x0 & DTC\_Ctrl = WATCH &Supplier Defined | 200 | TEST | n/a | Test fails speaker short to battery. | 33 | DTCSTest[].MonitorMAX=MAX | Test Passes speaker not short to battery | 33 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.159 | 0x9A0212 | Speaker #2 Circuit Short to Battery | AVAS\_Spkr\_Cfg = 0x0 & DTC\_Ctrl = WATCH &Supplier Defined | 1000 | NULL | n/a | Short to battery detected during normal operation (Continuous DTC) | 33 | DTCMaxAction() | Short to battery not detected during normal operation (Continuous DTC) | 33 | DTCMinAction() |
| R: 2.10.160 | 0x9A0213 | Speaker #2 Circuit Open | AVAS\_Spkr\_Cfg = 0x0 & DTC\_Ctrl = WATCH &Supplier Defined | 200 | TEST | n/a | Test fails speaker open circuit | 33 | DTCSTest[].MonitorMAX=MAX | Test Passes speaker not open circuit | 33 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.161 | 0x9A0213 | Speaker #2 Circuit Open | AVAS\_Spkr\_Cfg = 0x0 & DTC\_Ctrl = WATCH &Supplier Defined | 1000 | NULL | n/a | Open Circuit Detected in normal operation (Continuous DTC) | 33 | DTCMaxAction() | Open Circuit not detected during normal operation (Continuous DTC) | 33 | DTCMinAction() |
| R: 2.10.162 | 0xC10000 | Lost Communication with ECM/PCM “A” | DTC\_Ctrl = WATCH & Ignition\_Status = RUN (last known) & PwPckTq\_D\_Stat <> PwPckStrtInPrgrss\_TqNotAvail (last known) | 500 | TEST | n/a | GearRverse\_Status\_Available= Lost & GearRevrse\_D\_Actl\_Signal\_Received\_Flag = NULL | 127 | DTCMaxAction(),DTCSTest[].MonitorMAX=MAX | GearRverse\_Status\_Available= Available | 128 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.163 | 0xC10000 | Lost Communication with ECM/PCM “A” | DTC\_Ctrl = WATCH & Ignition\_Status = RUN (last known) & PwPckTq\_D\_Stat <> PwPckStrtInPrgrss\_TqNotAvail (last known) | 1000 | NULL | n/a | GearRverse\_Status\_Available= Lost & GearRevrse\_D\_Actl\_Signal\_Received\_Flag = NULL | 32 | DTCMaxAction() | GearRverse\_Status\_Available= Available | 16 | DTCMinAction() |
| R: 2.10.164 | 0xc14000 | Lost Communication with Body Control Module | DTC\_Ctrl = WATCH & Ignition\_Status = RUN (last known) & PwPckTq\_D\_Stat <> PwPckStrtInPrgrss\_TqNotAvail (last known) | 500 | TEST | n/a | Ignition\_Status\_Available = Lost & Ignition\_Status\_Signal\_Received\_Flag = NULL | 127 | DTCMaxAction(),DTCSTest[].MonitorMAX=MAX | Ignition\_Status\_Available = Available | 128 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.165 | 0xc14000 | Lost Communication with Body Control Module | DTC\_Ctrl = WATCH & Ignition\_Status = RUN (last known) & PwPckTq\_D\_Stat <> PwPckStrtInPrgrss\_TqNotAvail (last known) | 1000 | NULL | n/a | Ignition\_Status\_Available = Lost & Ignition\_Status\_Signal\_Received\_Flag = NULL | 32 | DTCMaxAction() | Ignition\_Status\_Available = Available | 16 | DTCMinAction() |
| R: 2.10.166 | 0xC42200 | Invalid Data Received from Body Control Module | PwPckTq\_D\_Stat <> PwPckStrtInPrgrss\_TqNotAvail (last known) | 500 | TEST | n/a | Ignition\_Status\_Available = Available & Ignition\_Status in message 0x3B3 is equal to invalid for five (5) seconds | 127 | DTCMaxAction(),DTCSTest[].MonitorMAX=MAX | Ignition\_Status\_Available = Available | 128 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.167 | 0xC42200 | Invalid Data Received from Body Control Module | PwPckTq\_D\_Stat <> PwPckStrtInPrgrss\_TqNotAvail (last known) | 500 | NULL | n/a | Ignition\_Status\_Available = Available & Ignition\_Status in message 0x3B3 is equal to invalid for five (5) seconds | 127 | DTCMaxAction() | Ignition\_Status\_Available = Available | 128 | DTCMinAction() |
| R: 2.10.168 | 0xE01A51 | Control Module Main Calibration Data Not Programmed | key in Run, ACC or Delayed Acc & Voltage between 10 and 15.5volts | Check at power on, ODST and changes in reverse status | TEST | n/a | Set when calibration data area is in an erased state. | 127 | DTCMaxAction(),DTCSTest[].MonitorMAX=MAX | Set when calibration data area is valid. | 128 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.169 | 0xE01A51 | Control Module Main Calibration Data Not Programmed | key in Run, ACC or Delayed Acc & Voltage between 10 and 15.5volts | Check at power on, ODST and changes in reverse status | NULL | n/a | Set when calibration data area is in an erased state. | 127 | DTCMaxAction() | Set when calibration data area is valid. | 128 | DTCMinAction() |
| R: 2.10.170 | 0xF00041 | Control Module General Checksum Failure | key in Run, ACC or Delayed Acc & Voltage between 10 and 15.5volts | Check at power on and ODST | TEST | n/a | Flash ROM Checksum Failure | 127 | DTCMaxAction(),DTCSTest[].MonitorMAX=MAX | Flash ROM Checksum – No failure detected | 128 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.171 | 0xF00041 | Control Module General Checksum Failure | key in Run, ACC or Delayed Acc & Voltage between 10 and 15.5volts | Check at power on and ODST | NULL | n/a | Flash ROM Checksum Failure | 127 | DTCMaxAction() | Flash ROM Checksum – No failure detected | 128 | DTCMinAction() |
| R: 2.10.172 | 0xF00042 | Control Module General Memory Failure | key in Run, ACC or Delayed Acc & Voltage between 10 and 15.5volts | Checked during serial number flash write. | TEST | n/a | EEPROM Write Error | 127 | DTCMaxAction(),DTCSTest[].MonitorMAX=MAX | No EEPROM Write Error detected | 128 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.173 | 0xF00042 | Control Module General Memory Failure | key in Run, ACC or Delayed Acc & Voltage between 10 and 15.5volts | Checked during serial number flash write. | NULL | n/a | EEPROM Write Error | 127 | DTCMaxAction() | No EEPROM Write Error detected | 128 | DTCMinAction() |
| R: 2.10.174 | 0xF00096 | Control Module Component Internal Failure | key in Run, ACC or Delayed Acc & Voltage between 10 and 15.5volts | Check at power on, ODST and changes in reverse status | TEST | n/a | Set when calibration data contains an invalid checksum. | 127 | DTCMaxAction(),DTCSTest[].MonitorMAX=MAX | Set when calibration data contains a valid checksum. | 128 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.175 | 0xF00096 | Control Module Component Internal Failure | key in Run, ACC or Delayed Acc & Voltage between 10 and 15.5volts | Check at power on, ODST and changes in reverse status | NULL | n/a | Set when calibration data contains an invalid checksum. | 127 | DTCMaxAction() | Set when calibration data contains a valid checksum. | 128 | DTCMinAction() |
| R: 2.10.176 | 0xF00316 | Batter Voltage Circuit Voltage Below Threshold | key in Run, ACC or Delayed Acc | 50 | NULL | n/a | Voltage transitions to UNDER\_VOLT as specified in 2.9.3 Voltage Range State Machine & VoltBelowThresholdDelayCfg time has passed. | 127 | DTCMaxAction() | Voltage transitions to NORM\_VOLT as specified in 2.9.3 Voltage Range State Machine | 128 | DTCMinAction() |
| R: 2.10.177 | 0xF00317 | Battery Voltage Circuit Voltage Above Threshold | key in Run, ACC or Delayed Acc | 50 | NULL | n/a | Voltage transitions to OVER\_VOLT as specified in 2.9.3 Voltage Range State Machine | 32 | DTCMaxAction() | Voltage transitions to NORM\_VOLT as specified in 2.9.3 Voltage Range State Machine | 16 | DTCMinAction() |
| R: 2.10.177a | $E02B-61 | TBD | (DTCSuppression\_Cfg[DTCSuppression\_Cfg\_Index], bit0 = 1) | 1000 | NULL | n/a | SWSigning\_KeyMode = DEVELOPMENT  Note: Supplier needs to define this dataflow. | 127 | DTCMaxAction() | SWSigning\_KeyMode = PRODUCTION | 255 | DTCMinAction() |
| R: 2.10.177b | $E02B-61 | TBD | (DTCSuppression\_Cfg[DTCSuppression\_Cfg\_Index], bit1 = 1) | 200 | TEST | n/a | SWSigning\_KeyMode = DEVELOPMENT  Note: Supplier needs to define this dataflow. | 127 | DTCSTest[].MonitorMAX=MAX | SWSigning\_KeyMode = PRODUCTION | 255 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.177c | $E02B-62 | TBD | (DTCSuppression\_Cfg[DTCSuppression\_Cfg\_Index], bit0 = 1) | 1000 | NULL | n/a | SWSigning\_KeyMode = PRODUCTION  & SWSignedKeysFound = DEVELOPMENT  Note: Supplier needs to define these dataflows. | 127 | DTCMaxAction() | SWSigning\_KeyMode = PRODUCTION  &  SWSignedKeysFound = PRODUCTION | 255 | DTCMinAction() |
| R: 2.10.177d | $E02B-62 | TBD | (DTCSuppression\_Cfg[DTCSuppression\_Cfg\_Index], bit1 = 1) | 200 | TEST | n/a | SWSigning\_KeyMode = PRODUCTION  & SWSignedKeysFound = DEVELOPMENT  Note: Supplier needs to define these dataflows | 127 | DTCSTest[].MonitorMAX=MAX | SWSigning\_KeyMode = PRODUCTION  &  SWSignedKeysFound = PRODUCTION | 255 | DTCSTest[].MonitorMIN=MIN |

#### DTCMaxAction(index) Function

This function will only be called for continuous DTCs (in continuous and SelfTest DTC checks).

Table ‑ DTCMaxAction(index) Requirements

|  |  |
| --- | --- |
| **Rqmt No.** | **Minimal Requirements** |
| R: 2.10.178 | Set DTCRun[index].Monitor = MAX |
| R: 2.10.179 | Allocate NVM aging storage for the requested DTC |
| R: 2.10.180 | If NVM aging storage is full, it shall commandeer the NVM of the oldest (as defined by the value of the aging count) DTC. If there are multiple "oldest" DTCs, then it is left to the supplier which NVM location to commandeer. |
| R: 2.10.181 | If the DTC already exists in NVM, it shall restart its aging count. |
| R: 2.10.182 | Set AVAS as Faulted |
| R: 2.10.183 | (can perform other tasks – clearing functional latches, for example) |

#### DTCMinAction(index) Function

This function will only be called for continuous DTCs.

Table ‑ DTCMinAction(index) Requirements

| **Rqmt. No.** | **Minimal Requirements** |
| --- | --- |
| R: 2.10.184 | Shall set DTCRun[index].Monitor = MIN |
| R: 2.10.185 | If there are no active DTCs set AVAS as ACTIVE/INACTIVE (Not Faulted) |
| R: 2.10.186 | (can perform other tasks) |

#### DTC Suppression

DTCs mapped in **Table 2.10‑29 DTC Index Map for DTC Suppression** shall be able to be suppressed by disabling the DTC mask. This does not change the detection requirements, nor does it eliminate the need of the module to manage the failure mode or take mitigating actions. It only controls the actual logging of the DTC.

Table ‑ Suppressing DTCs

|  |  |
| --- | --- |
| **Rqmt No.** | **Requirements** |
| R: 2.10.187 | Each DTC shall be represented by a bit map in the DTC Suppression array – DTCSuppression\_Cfg[n]. Bitmap values as follow:  Hex 00 (Binary 00) : both self-test and continuous DTC are suppressed, they cannot be set or cleared  Hex 1 (Binary 01) : only continuous can be set and cleared, self-test DTC is suppressed  Hex 2 (Binary 10) : only self-test can be set and cleared, continuous DTC is suppressed  Hex 3 (Binary 11) : both self-test and continuous DTC can be set and cleared  Example: When ( DTCSuppression\_Cfg[n] = 1 | DTCSuppression\_Cfg[n] = 3): Continuous DTC can be set and cleared  When ( DTCSuppression\_Cfg[n] = 2 | DTCSuppression\_Cfg[n] = 3): Self-test DTC can be set and cleared |
| R: 2.10.188 | Refer to **Table 2.10‑29 DTC Index Map for DTC**  for mapping details. |
| R: 2.10.189 | Continuous DTCs are defined by SelfTest = NULL |
| R: 2.10.190 | Self-test or On-Demand DTCS are defined by SelfTest = TEST |

Table ‑ DTC Index Map for DTC Suppression

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rqmt. No.** | **DTCSuppression\_Cfg\_Index** | **DTC Number** | **Description** | **DTCSuppression\_Cfg[DTCSuppression\_Cfg\_Index] Initial Value** |
| R: 2.10.191 | AVAS is currently not supporting DTC Suppression thus no DTCs are included in this index. | | | |

#### Definitions for DTC Status Bits

Table ‑ Definitions for DTC Status Bits

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Rqmt. No.** | **Bit** | **Name** | **Supported** | **Value** | |
| **R: 2.10.192** | 0 | TestFailed | No | Always 0 |
| **R: 2.10.193** | 1 | TestFailedThisOperationCycle | Yes | If (DTC in EEPROM  & DTCRun[x].MonitorMax = MAX) then 1  Else 0 |
| **R: 2.10.194** | 2 | PendingDTC | No | Always 0 |
| **R: 2.10.195** | 3 | ConfirmedDTC | Yes | If (DTC in EEPROM) then 1  Else 0 |
| **R: 2.10.196** | 4 | TestNotCompletedSinceLastCleared | No | Always 0 |
| **R: 2.10.197** | 5 | TestFailedSinceLastCleared | No | Always 0 |
| **R: 2.10.198** | 6 | TestNotCompletedThisOperationCycle | Yes | If (DTCRun[x].MonitorMax = MAX | DTCRun[x].MonitorMin = MIN) then 0  Else 1 |
| **R: 2.10.199** | 7 | WarningIndicatorRequested | No | Always 0 |

#### Diagnostic Cycle Monitor

This state machine controls the diagnostic cycle monitor. The cycle monitor is defined to restart on a transition to ignition OFF. When this transition occurs, there is some maintenance work that needs to be done in the AgeAppropriateDTCs() function, the ClearLatches() function, and the ResetDTCcounters() function.



Figure ‑ Control DTC Setting State Transition Diagram

Table ‑ Diagnostic Cycle Monitor State Transition Diagram Requirements

| **Rqmt No.** | **Source State -> Destination State** |
| --- | --- |
| R: 2.10.200 | 0->1.1 |
| R: 2.10.201 | 1->2.1 |
| R: 2.10.202 | 2->1.1 |

#### ResetDTCCounters() Function

|  |  |
| --- | --- |
|  | **ResetDTCcounters()** |
|  | **/\* Set all DTCRun[].Count Continuous DTC instances to 0\*/**  For j = 0 to MAXDTCIDX  DTCRun[j].Count = 0  End For |
|  | **/\* Set all DTCRun[].Monitor Continuous DTC instances to NULL\*/**  For j = 0 to MAXDTCIDX  DTCRun[j].MonitorMax = NULL  DTCRun[j].MonitorMin = NULL  End For |

#### AgeAppropriateDTCs() Function

Step through all the DTCs and any of them with MIN set get aged. Need to call the *CheckLatchDTCs()* function before calling this function

|  |  |
| --- | --- |
|  | **AgeAppropriateDTCs()** |
|  | **/\* Set all Continuous DTCs with DTCRun[j].MonitorMin=MIN, AgeDTC()\*/**  For j = 0 to MAXDTCIDX  If DTCRun[j].MonitorMin=MIN, then AgeDTC(j)  End For |
|  |  |

#### AgeDTC(index)

Table ‑ AgeDTC(index) Requirements

|  |  |
| --- | --- |
| **Rqmt. No.** | **Minimal Requirements** |
| **R: 2.10.203** | If the DTC exists in NVM, it shall age the DTC |
| **R: 2.10.204** | If the DTC has been aged and it is now completely aged, then free up the DTC aging NVM |
| **Note:** | It is left to the supplier to select the maximum aging value that meetings Ford NetCom requirements. |

### AVAS Configuration Data (Method 2 & 3)

#### A note about Method 2 data versus Method 3 data

In general, Method 2 (DID written) data is used for configuration values that can change between specific instances of vehicles or classes of vehicles within a platform.

Method 3 (file download) data is used to configure general behaviors of the module, such as timers and levels; as well as data that will not vary within a platform, such as vehicle geometries.

Note: AVAS does not use any Method 2 or Method 3 configuration data.

#### Method 2 Data

Table ‑ Miscellaneous Service $2E Requirements

|  |  |
| --- | --- |
| R: 2.10.205 | All Method 2 Configuration Parameters must be stored using "Double Redundant" EEPROM events or better. |

**Inhale/Exhale** - Configuration Data which is located in a configuration DID (DE00….DExx) prior to the buffer block.

**As-Built -** Configuration data which is written at Ford EOL (Vehicle Operation Column) AND in the inhale exhale range.

**Module Manufacturer** - Configuration items which are just written at the tier 1 suppliers manufacturing plant.

**FCSD Customer Preference** - Items which may be written/configured at Ford Customer Service Division due to Customer Preferences

#### Method 2 Configuration Parameters

The following parameters are used to configure AVAS using Method 2 configuration. Default values are taken from the Data Dictionary.

Table ‑ Method 2 Configuration Parameters

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Rqmt. No** | **Name of Data** | **Description** | **Initial Value** | **Unit** | **Inhale/Exhale** | **As-Built/Vehicle Operations** | **Module Manufacturer** | **FCSD Customer Preference** |
| Note: AVAS does not use Method 2 data for Configuration. | | | | | | | | |

#### AVAS Configuration Data (Supplier Range FD00-FEFF)

The parameters listed in **Table 2.10‑35 Supplier Range Configuration Parameters** are configurable parameters set by the supplier in the FD00-FEFF range of DIDs. FMC does not maintain these parameters and will not write to them at end of line nor store them in the As-Built database.

Table ‑ Supplier Range Configuration Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rqmt. No** | **Name of Data** | **Description** | **Initial Value** | **Unit** |
| **R: 2.10.206** | Max\_Speed\_Sound\_Cfg | Maximum limit value to produce sound | See Data Dictionary for Initial/Default value | KPH |
| **R: 2.10.207** | Vehicle\_Type\_Cfg | Vehicle Brand |  |
| **R: 2.10.208** | Speaker\_On\_Off\_Cfg | Indicates if the system should use a switch to turn on/off the sound. |  |
| **R: 2.10.209** | AVAS\_Spkr\_Cfg | Number of Speakers in actual hardware |  |
| **R: 2.10.210** | TransitionDelayCfg | Delay in msec before transitioning from playing sound in the front speaker to playing sound in the rear speaker and vice versa. | Milliseconds |
| **R: 2.10.211** | The supplier shall ensure these parameters are set to their correct value as defined in the data dictionary before delivery to Ford. | | | |

|  |
| --- |
| **Parameter Information** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameter # 1 | | | | | |
| Parameter Name | | Max\_Speed\_Sound\_Cfg | | | |
| Description | | Maximum limit value to produce sound | | | |
| DataIdentifier Size (bits) | | *8* | | | |
| Format Information: | | | | | |
| **Size (bits)** | **Parameter Info** | | **Units** | **Min Scaled Value** | ***Max Scaled Value*** |
| *8* | Max\_Speed\_Sound\_Cfg | | *kph* | *0* | *655* |

|  |  |
| --- | --- |
| Parameter # 2 | |
| Parameter Name | Vehicle\_Type\_Cfg |
| Description | Vehicle brand |
| DataIdentifier Size (bits) | *1* |
| Format Information: | |
| *Value* | *State Description* |
| *0* | *FORD* |
| *1* | *LINCOLN* |

|  |  |
| --- | --- |
| Parameter # 3 | |
| Parameter Name | Speaker\_On\_Off\_Cfg |
| Description | Indicates if the system should use a switch to turned ON/OFF the sound |
| DataIdentifier Size (bits) | *1* |
| Format Information: | |
| *Value* | *State Description* |
| *0* | *SWITCH\_DISABLED* |
| *1* | *SWITCH\_ENABLED* |

|  |  |
| --- | --- |
| Parameter # 4 | |
| Parameter Name | AVAS\_Spkr\_Cfg |
| Description | The number of speakers supported by the AVAS’s actual hardware. Currently, it can be one or two. |
| DataIdentifier Size (bits) | *1* |
| Format Information: | |
| *Value* | *State Description* |
| *0x0* | *2 Speaker Configuration* |
| *0x1* | *1 Speaker Configuration* |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameter # 5 | | | | | |
| Parameter Name | | TransitionDelayCfg | | | |
| Description | | Delay in msec before transitioning from playing sound in the front speaker to playing sound in the rear speaker and vice versa. | | | |
| DataIdentifier Size (bits) | | *10* | | | |
| Format Information: | | | | | |
| **Size (bits)** | **Parameter Info** | | **Units** | **Min Scaled Value** | ***Max Scaled Value*** |
| *10* | TransitionDelayCfg | | *msec* | *0* | *1000* |

### Fault conditions for Speaker

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DTC (HEX) | DTC | DTC Type | Root Description | Failure Type Byte Description | |
| 0x9A0101 | B1A01-01 | D | Speaker #1 | General Electrical Failure |
| 0x9A0111 | B1A01-11 | D | Speaker #1 | Circuit Short To Ground |
| 0x9A0112 | B1A01-12 | D | Speaker #1 | Circuit Short To Battery |
| 0x9A0113 | B1A01-13 | D | Speaker #1 | Circuit Open |
| 0x9A0201 | B1A02-01 | D | Speaker #2 | General Electrical Failure |
| 0x9A0211 | B1A02-11 | D | Speaker #2 | Circuit Short To Ground |
| 0x9A0212 | B1A02-12 | D | Speaker #2 | Circuit Short To Battery |
| 0x9A0213 | B1A02-13 | D | Speaker #2 | Circuit Open |

### Fault conditions for Lost Communication

When the system detects lost Communication with any of the signals received, the module should set a DTC.

In the document: “DiagnosticFaultCoverageAndDTCNumbersDesignConsideration” Table 3.2, the recommended filtering time is 10 samples. So for each lost DTC, the time to wait before setting a LOST communication DTC with a module is 10 times the signal transmission rate.

When Communication is lost with the ECM/PCM (Veh\_V\_ActlEng, VehVActlEng\_D\_Qf, GearLvrPos\_D\_Actl, GearRvrse\_D\_Actl, PwPckTq\_D\_Stat) the AVAS module should set a DTC after MediumLostTime\_Cfg.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DTC (HEX) | DTC | DTC Type | Root Description | Failure Type Byte Description | |
| 0xC100 00 | | U0100-00 | C | Lost Communication With ECM/PCM "A" | **No Sub Type Information** |

When Ignition\_Status is not received from BCM, the AVAS module should set a DTC after SlowLostTime\_Cfg.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DTC (HEX) | DTC | DTC Type | Root Description | Failure Type Byte Description | |
| 0xC14000 | U0140-00 | C | Lost Communication With Body Control Module | **No Sub Type Information** |

When data from BCM is not valid, module should set the following DTC:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DTC (HEX) | DTC | DTC Type | Root Description | Failure Type Byte Description | |
| 0xC42200 | | U0422-00 | C | Invalid Data Received From Body Control Module | **No Sub Type Information** |

### Fault conditions for Internal Failures with AVAS ECU:

Following faults represent internal failures of the module:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DTC (HEX) | DTC | DTC Type | Root Description | Failure Type Byte Description |
| 0xF00041 | U3000-41 | CD | Control Module | General Checksum Failure | |
| 0xF00042 | U3000-42 | C | Control Module | General Memory Failure | |
| 0xF00096 | U3000-96 | CD | Control Module | Component Internal Failure | |

For issues related with configuration, module should set the corresponding DTC:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DTC (HEX) | DTC | DTC Type | Root Description | Failure Type Byte Description |
| 0xE01A51 | U201A-51 | CD | Control Module Main Calibration Data | No Sub Type Information | |

For issues related with battery conditions, module should set the corresponding DTC:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DTC (HEX) | DTC | DTC Type | Root Description | Failure Type Byte Description |
| 0xF00316 | U3003-16 | CD | Battery Voltage | Circuit Voltage Below Threshold | |
| 0xF00317 | U3003-17 | CD | Battery Voltage | Circuit Voltage Above Threshold | |

The DTC to report circuit voltage below threshold should only be set after a configurable delay (VoltBelowThresholdDelayCfg):

### CAN Based Data Identifiers

#### DID 0x1505 Vehicle Speed - High Resolution

|  |  |
| --- | --- |
| DataIdentifier Value | *0x1505* |
| DataIdentifier Name | *Vehicle Speed - High Resolution* |
| DataIdentifier Size (bytes) | *2* |
| *DataIdentifier Type* | *unsigned* |

|  |
| --- |
| **Read Information (Service 0x22 - ReadDataByIdentifier)** |

|  |  |
| --- | --- |
| *Readable in Sessions* | *0x01,0x03* |

|  |
| --- |
| **DataIdentifier Format Information** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Parameter Info** | **Offset** | **Resolution** | **Units** | **Min Scaled Value** | ***Max Scaled Value*** |
| *Vehicle Speed - High Resolution* | *0* | *0.0078* | *Kph* | *0* | *511.9921875* |

**Note: Signal *Veh\_V\_ActlEng* does not have the same resolution as DID 0x1505, in order to have compatible values, module should report the most accurate value from signal *Veh\_V\_ActlEng.***

#### DID 0x7218 Reverse Gear Position

|  |  |
| --- | --- |
| DataIdentifier Value | *0x7218* |
| DataIdentifier Name | *Reverse Gear Position* |
| DataIdentifier Size (bytes) | *1* |
| *DataIdentifier Type* | *State encoded* |

|  |
| --- |
| **Read Information (Service 0x22 - ReadDataByIdentifier)** |

|  |  |
| --- | --- |
| *Readable in Sessions* | *0x01,0x03* |

|  |
| --- |
| **DataIdentifier Format Information** |

|  |  |  |
| --- | --- | --- |
| **Value** | ***State Description*** | ***GearRvrse\_D\_Actl*** |
| 0x00 | *Not Engaged* | *Inactive\_Not\_Confirmed || Inactive\_Confirmed || Active\_Not\_Confirmed* |
| 0x01 | *Engaged* | *Active\_Confirmed* |
| 0x02 | *Malfunction* | *Fault* |
| *0xFF* | *Invalid Error State* | *N/A* |

#### DID 0x0599 PowerPack State

|  |  |
| --- | --- |
| DataIdentifier Value | *0x0599* |
| DataIdentifier Name | *PowerPack State* |
| DataIdentifier Size (bytes) | *1* |
| *DataIdentifier Type* | *State encoded* |

|  |
| --- |
| **Read Information (Service 0x22 - ReadDataByIdentifier)** |

|  |  |
| --- | --- |
| *Readable in Sessions* | *0x01,0x03* |

|  |
| --- |
| **DataIdentifier Format Information** |

|  |  |  |
| --- | --- | --- |
| **Value** | ***State Description*** | ***PwPckTq\_D\_Stat*** |
| 0x00 | *Power Pack is not available, e.g., key-off, accessory mode, key in run (but requiring state 2 before entering state 1 or 3).* | *OFF\_NO\_TQ* |
| 0x01 | *Power Pack is available for non-propulsive purposes only. Typically, due to a remote start to warm up or cool the vehicle.* | *ON\_NO\_TQ* |
| 0x02 | *Customer has initiated a start request (to transition to state 1 or 3), and that procedure is imminent or in progress.* | *START\_IN\_PROGRESS* |
| *0x03* | *Power Pack is available for propulsive and non-propulsive purposes. Put in gear & step on accel pedal and vehicle will move.* | *ON\_TQ\_AVAILABLE* |

#### DID 0x40B5 Ignition Position Final Status

|  |  |
| --- | --- |
| DataIdentifier Value | *0x40B5* |
| DataIdentifier Name | *Ignition Position Final Status* |
| DataIdentifier Size (bytes) | *1* |
| *DataIdentifier Type* | *State encoded* |

|  |
| --- |
| **Read Information (Service 0x22 - ReadDataByIdentifier)** |

|  |  |
| --- | --- |
| *Readable in Sessions* | *0x01,0x03* |

|  |
| --- |
| **DataIdentifier Format Information** |

|  |  |  |
| --- | --- | --- |
| **Value** | ***State Description*** | ***Ignition\_Status*** |
| 0x00 | *Null – Unknown Info* | *UNKNOWN || INVALID* |
| 0x01 | *Off* | *OFF* |
| 0x02 | *Acc* | *ACCESORY* |
| 0x03 | *Run* | *RUN* |
| *0x04* | *Start* | *START* |

#### DID 0x0130 Transmission Shift Lever Position

|  |  |
| --- | --- |
| DataIdentifier Value | *0x0130* |
| DataIdentifier Name | *Transmission Shift Lever Position* |
| DataIdentifier Size (bytes) | *1* |
| *DataIdentifier Type* | *State encoded* |

|  |
| --- |
| **Read Information (Service 0x22 - ReadDataByIdentifier)** |

|  |  |
| --- | --- |
| *Readable in Sessions* | *0x01,0x03* |

|  |
| --- |
| **DataIdentifier Format Information** |

|  |  |  |
| --- | --- | --- |
| **Value** | ***State Description*** | ***Transmission Shift Lever Position*** |
| 0x00 | *Park* | *Park* |
| 0x01 | *Reverse* | *Reverse* |
| 0x02 | *Neutral* | *Neutral* |
| 0x03 | *Drive* | *Drive* |
| 0x05 | *Sport* | *Sport/Drive Sport* |
| 0x06 | *Low* | *1* |
| 0x07 | *Manuel Gear Select – 1st* | *2* |
| 0x08 | *Manuel Gear Select – 2nd* | *3* |
| 0x09 | *Manuel Gear Select – 3rd* | *4* |
| 0x0A | *Manuel Gear Select –4th* | *5* |
| 0x0B | *Manuel Gear Select –5th* | *6* |
| 0x0F | *Fault* | *Fault* |

### Bootloader

The bootloader will be responsible for signature verification of downloaded VBF files to an ECU. To verify the authenticity of a VBF, the bootloader will require the production/development key(s) to be flashed into a read only area of the bootloader (i.e. data block). Depending on an ECUs capability the bootloader can have the following public key storage solution.

1) Single Key Storage

2) Two Key Storage

Note: Public key shall NOT be allowed to be modified via diagnostics or internal ECU application(s).

Once the public key(s) have been flashed into the bootloader, an ECU can verify the Software Signatures of the VBF files downloaded to ensure the software is secure and authentic. The signature verification will occur for each Logical Block in the VBF file. Each downloadable SW VBF needs to be individually signed. The bootloader can erase multiple SW download areas, but due to the way the signature process works each VBF must be handled sequentially. The bootloader will be responsible for the signature verification process below.

The ECU supplier will be responsible for populating the SW Signature, Public Key Hash, and Verification Structure fields in the VBF for development builds using HexView. For production builds, the ECU supplier is only responsible for populating the Public Key Hash, and Verification Structure fields. The SW Signature field can only be populated by Ford IVS Team.

**Figure 2‑7 Software Signature & VS Generation Process**



Figure ‑ Software Signature Verification Process for One Key



Figure ‑ Software Signature Verification Process for Two Keys

#### Single Key Storage

An ECU can store a single public key in the bootloader which can be used in the SW signature verification. This method allows only one public key to be store which could be a development key or production key based on the target build. For all Development builds (before TT builds), the Development public key will be included into the bootloader. Development SW shall be signed by the SW developers. For production builds (TT builds and beyond), the production public key will be included into the bootloader. At any point in time, when a development SW build needs to be tested on a production vehicle then the module will need to be replaced with a module that supports development builds.

**Implications**

* Two different HW part numbers are required for Development and Production
* Cannot install Development SW on a Production ECU
* Cannot install Production SW on a Development ECU
* Production SW has to be signed Via IVS
* Testing SW on a Production ECU will require
  + Swapping the Production ECU with a Development ECU
  + Or Releasing Experimental SW Production signed by IVS

Table ‑ – Single Key Storage Requirements

|  |  |
| --- | --- |
| **Rqm’t Num.** | **Requirement** |
| R: 2.10.177e | The ECU shall store only one public key in the primary bootloader for SW signing verification. |
| R: 2.10.177f | The development public key shall be store in the PBL for pre TT vehicles. |
| R: 2.10.177g | The production public key shall be store in the PBL for TT vehicles and beyond. |
| R: 2.10.177h | The public key shall be store in read only area. |

#### Two Key Storage

An ECU can store two public keys in the bootloader which can be used in the SW signature verification. This method allows two public keys to be stored which will be a development key and production key. By default, the bootloader will be configured to use the production public key. When development builds are required, the ECU can switch to using the development public key using a diagnostic routine. If the ECU is using the production public key, it can only be switched to using the development key with backend authentication (i.e. FIMCO).

**Implications**

* Requires only one HW part number
* Cannot install Development SW on an ECU with a disabled Development key
* Can install Production SW on an ECU with an enabled or disabled Development Key
* Production SW has to be signed by IVS
* Installing Development SW on an ECU with a disabled Development Key requires
  + Enabling the Development key, but this requires using a test tool connected to that vehicle. (OTA will not support this)
  + Then you can install Development or Production SW (OTA supports both)
* Switching from Production to Development requires a secure method approved by the SW Security team.
  + This is only available in the application. The bootloader cannot do the switch.
  + The bootloader must be able to determine if the Development key is enabled or not even if the application is missing. This implies the Development key enable is store in a special Data Flash
* Switching from Development to Production does not require any security.
* The secure method for switching to Development requires the ECU manufacturer to maintain a database with ECU serial number to ESN mapping. The diagnostic routine that supports switching to Development will use the ESN in encrypted format.
* The security team didn’t want a common method for this diagnostic routine.
* Service Test tool will not support Switching Routine.

Table ‑ – Two Key Storage

|  |  |
| --- | --- |
| **Rqm’t Num.** | **Requirement** |
| R: 2.10.177i | The ECU shall store both the development and production public keys in the primary bootloader or HSM for SW signing verification.  Note: Storing the keys in the bootloader introduces issues when supporting AB Swap since the application needs to access the keys. |
| R: 2.10.177j | The ECU shall use the production key as the default public key. |
| R: 2.10.177k | The ECU shall disable the development public key using a type1 diagnostic routine with no unique ESN or FIMCO access required. |
| R: 2.10.177l | The ECU shall allow the enable development public key using a type1 diagnostic routine only after gaining access using a unique ESN password or backend authentication access “FIMCO”. |
| R: 2.10.177m | The public key shall be stored in read only area. |

## Factory Mode and Transport Mode

|  |  |
| --- | --- |
| Requirement Number | Description |
| **R: 2.11.1** | The behavior of the AVAS module in FACTORY mode and TRANSPORT mode is identical to its behavior in NORMAL mode and does not change. |

## Data Dictionary

AVAS

FS-ML3T-14G113-AA

Data Dictionary v1.0

Tuesday, August 21, 2018 Revision 1.0

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Avas Output**

**Definition** **Describes the module output.**

Attributes

Units Range Default Number of Values

?? ??

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **AVAS\_Ignition\_Status**

**Definition** **Internal dataflow to store the value of Ignition\_Status**

Attributes

Units Range Default Number of Values

Hex Value 0x0, 0x1, 0x2, 0 6

0x4, 0x8, 0xF

DataFlow Value Description

0x0 UNKNOWN

0x1 OFF

0x2 ACCESORY

0x4 RUN

0x8 START

0xF INVALID

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **AVAS\_PwPckTq\_D\_Stat**

**Definition** **Internal dataflow to store the value of PwPckTq\_D\_Stat**

Attributes

Units Range Default Number of Values

Integer Value 0-3 0 4

DataFlow Value Description

0x0 OFF\_NO\_TQ

0x1 ON\_NO\_TQ

0x2 START\_IN\_PROGRESS

0x3 ON\_TQ\_AVAILABLE

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **AVAS\_Spkr\_Cfg**

**Definition** **The number of speakers supported by the AVAS's actual hardware. As of**

**FS-KU5T-14G113-AA it can be one or two.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 2 Speaker Configuration

0x1 1 Speaker Configuration

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **AVAS\_State**

**Definition** **Indicates when module should be producing sound if the conditions are**

**met.**

Attributes

Units Range Default Number of Values

Integer Values 0-2 1 3

DataFlow Value Description

0 INACTIVE: AVAS is not producing sound.

1 ACTIVE: AVAS is producing sound.

2 FAULT: AVAS is faulted and cannot produce sound.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **AVAS\_Temp\_Disable**

**Definition** **Dataflow which indicates whether the AVAS module is temporarily**

**disabled by the customer.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 OFF: AVAS is operating normally and not temporarily disabled.

0x1 ON: Customer has temporarily disabled the AVAS manually.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **controlDTCSetting**

**Definition** **0x85 Service to request ECU to suppress detection of DTCs.**

Attributes

Units Range Default Number of Values

Discrete Off, On 0 2

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DiagOutputCtrlSpeedLimit\_Cfg**

**Definition** **Speed limit above which SelfTest and extended diagnostic session (I/O**

**control) stop running.**

Attributes

Units Range Default Number of Values

KPH 5-255 5 250

DataFlow Value Description

5 KPH above which SelfTest and extended diagnostic session (I/O

control) stop running.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DiagResetDelay\_Cfg**

**Definition** **Amount of time in msec to delay before resetting due to (1) Test tool**

**request to RESET; (2) Exiting Extended Session.**

Attributes

Units Range Default Number of Values

msec 0-12750 500 255

DataFlow Value Description

500 Since various ECUs can misbehave, the request is that we use a

Method 2 config to allow VO to adjust the delay.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DiagSession**

**Definition** **Indicates the active session of Diagnostics**

Attributes

Units Range Default Number of Values

Discrete DEFAULT, UNKNOWN 4

EXTEND,

PROGRAM,

UNKNOWN

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DTC\_Ctrl**

**Definition** **Used by the controlDTCSetting state machine to determine if a DTC**

**should be watched or ignored.**

Attributes

Units Range Default Number of Values

Discrete IGNORE, IGNORE 2

WATCH

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DTCRun[].Count**

**Definition** **A unique 8-bit signed counter for every continuous test DTC which is**

**incremented/decremented by a specified value based on if the failure is**

**Present/Not\_Present**

Attributes

Units Range Default Number of Values

Numeric (-128)-127 0 256

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DTCRun[].Monitor**

**Definition** **Array to hold values of whether a DTC has reached max increment or**

**minimum increment for continuous DTCs**

Attributes

Units Range Default Number of Values

Discrete NULL, MIN, NULL 3

MAX

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DTCRun[].MonitorMax**

**Definition** **When this is set to MAX, a continuous DTC has reached a threshold**

**where it must be set.**

Attributes

Units Range Default Number of Values

Discrete NULL, MIN, NULL 3

MAX

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DTCRun[].MonitorMin**

**Definition** **When this is set to MIN, a continuous DTC has reached a threshold**

**where it has aged to where it can be cleared.**

Attributes

Units Range Default Number of Values

Discrete NULL, MIN, NULL 3

MAX

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DTCSTest[].Count**

**Definition** **A unique 8-bit signed counter for every self-test DTC which is**

**incremented/decremented by a specified value based on if the failure is**

**Present/Not\_Present.**

Attributes

Units Range Default Number of Values

Numeric (-128)-127 0 256

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DTCSTest[].Monitor**

**Definition** **Array to hold values of whether a DTC has reached a max increment or**

**minimum increment for Self-Test (on demand) DTCs**

Attributes

Units Range Default Number of Values

Discrete NULL, MIN, NULL 3

MAX

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DTCSTest[].MonitorMax**

**Definition** **When this is set to MAX, an on demand, self-test DTC has reached a**

**threshold where it must be set.**

Attributes

Units Range Default Number of Values

Discrete NULL, MIN, NULL 3

MAX

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DTCSTest[].MonitorMin**

**Definition** **When this is set to MIN, an on demand, self-test DTC has reached a**

**threshold where it can be cleared.**

Attributes

Units Range Default Number of Values

Discrete NULL, MIN, NULL 3

MAX

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **DTCSuppression\_Cfg**

**Definition** **Array containing a representation of each DTC that may be suppressed.**

**See the Diagnostics section of the FS for more details.**

Attributes

Units Range Default Number of Values

Bitmap 00, 01, 10, 11 None 4

DataFlow Value Description

00 Both self-test and continuous DTCs are suppressed, they cannot

be set or cleared

01 Only continuous DTCs can be set and cleared, self-test DTCs are

suppressed

10 Only self-test DTCs can be set and cleared, continuous DTCs are

suppressed

11 Both self-test and continuous DTC can be set and cleared

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **GearLvrPos\_Available**

**Definition** **This value is LOST when the GearLvrPos\_D\_Actl signal is lost or its value**

**is Fault. This value is AVAILABLE when signal is present and the value is**

**not Fault.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 LOST: GearLvrPos\_D\_Actl is missing or its value is Fault.

0x1 AVAILABLE: GearLvrPos\_D\_Actl is present and its value is not

Fault.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **GearLvrPos\_D\_Actl**

**Definition** **NETCOM signal indicating automatic transmission gear lever position.**

**Obsolete and incorrect usage starting with MY2020 Gen4 HEV. This**

**signal is in 0x230 message "TransGearData."**

Attributes

Units Range Default Number of Values

Integer Value 0-7 0 16

DataFlow Value Description

0x0 Park

0x1 Reverse

0x2 Neutral

0x3 Drive

0x4 Sport/Drive Sport

0x5 Low

0x6 1

0x7 2

0x8 3

0x9 4

0xA 5

0xB 6

0xC undefined

0xD undefined

0xE unknown position

0XF fault

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **GearLvrPos\_D\_Actl\_Signal\_Received\_Flag**

**Definition** **Dataflow that contains NULL when no GearLvrPos\_D\_Actl signal was**

**received and RECEIVED when the signal was received. After checking,**

**AVAS must set the value of this dataflow to NULL.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 NULL: No GearLvrPos\_D\_Actl signal was received.

0x1 RECEIVED: A GearLvrPos\_D\_Actl signal was received.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **GearLvrPos\_Status\_TimeLost\_Cfg**

**Definition** **Time to wait when the GearLvrPos\_D\_Actl received flag is NULL before**

**transitioning to LOST state.**

Attributes

Units Range Default Number of Values

Milliseconds 0-5000 5000 5000

DataFlow Value Description

5000 Delay in mSec before declaring GearLvrPos\_D\_Actl as lost.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **GearPark\_Status**

**Definition** **This dataflow shows whether or not the vehicle is in a state where the**

**gear lever position is park or not. It also indicates if the state is**

**unknown because the signal is LOST.**

Attributes

Units Range Default Number of Values

Integer Value 0-2 0 3

DataFlow Value Description

0x0 LOST: GearLvrPos\_D\_Actl signal is lost or is faulty and we cannot

determine if the vehicle gear lever is in park.

0x1 ACTIVE: The gear lever position is in park.

0x2 INACTIVE: The gear lever position is not in park.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **GearRverse\_Status**

**Definition** **Indicates if the reverse gear is active, inactive, or lost.**

Attributes

Units Range Default Number of Values

Integer Values 0-2 0 3

DataFlow Value Description

0 LOST: The transmitter of GearRvrse\_D\_Actl has gone missing or

is faulty and cannot be trusted.

1 INACTIVE: The gear lever position is not in Reverse.

2 ACTIVE: The gear lever position is in Reverse.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **GearRverse\_Status\_Available**

**Definition** **Set the status communication of the signal indicating if reverse signal is**

**available in system.**

Attributes

Units Range Default Number of Values

Integer Values 0-1 0 2

DataFlow Value Description

0 LOST: signal not available in the system.

1 AVAILABLE: Signal is available in the system.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **GearRvrse\_D\_Actl**

**Definition** **NETCOM Signal indicating if the reversing gear is in use or not.**

Attributes

Units Range Default Number of Values

Integer Values 0-7 0 8

DataFlow Value Description

0x0 Inactive\_not\_confirmed

0x1 Inactive\_confirmed

0x2 Active\_not\_confirmed

0x3 Active\_confirmed

0x4 Not used

0x5 Not used

0x6 Not used

0x7 Fault

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **GearRvrse\_D\_Actl\_Signal\_Received\_Flag**

**Definition** **Dataflow that contains NULL when no GearRvrse\_D\_Actl signal was**

**received and RECEIVED when the signal was received. After checking,**

**AVAS must set the value of this dataflow back to NULL.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 NULL: No GearRvrse\_D\_Actl signal was received.

0x1 Received: A GearRvrse\_D\_Actl signal was received.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **GearRvrseStatus\_TimeLost\_Cfg**

**Definition** **Time to wait when GearRvrse\_D\_Actl\_Signal\_Received\_Flag remains**

**NULL before transitioning to LOST.**

Attributes

Units Range Default Number of Values

Milliseconds 0-5000 5000 5000

DataFlow Value Description

5000 Time in mSec from when signal goes missing to wait before setting

DTC when GearRvrse\_D\_Actl is lost.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **HiOvrTime\_Cfg**

**Definition** **Defines the amount of time to wait in a high voltage (HI\_V) condition**

**before declaring an OVER\_V condition.**

Attributes

Units Range Default Number of Values

Milliseconds 0-9000 500 900

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **HiRecovTime\_Cfg**

**Definition** **Defines the amount of time that the voltage must be below Vmax\_Cfg[i]**

**before recovering from an OVER\_V condition and allowing input**

**sampling again.**

Attributes

Units Range Default Number of Values

Milliseconds 0-6000 2000 600

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **HiTime\_Cfg**

**Definition** **After Vbatt pops high, this is the minimum amount of time normal voltage**

**must be present before allowing inputs to be read again (return to**

**NORM\_V). This value must be based on worst-case analysis of the**

**slowest circuit that uses this voltage range.**

Attributes

Units Range Default Number of Values

Milliseconds 0-2000 55 400

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Ignition\_Status**

**Definition** **Indicates current ignition. This signal is in 0x3B3 message "BodyInfo\_3"**

Attributes

Units Range Default Number of Values

Integer Values 0-F 0 6

DataFlow Value Description

0x0 UNKNOWN

0x1 OFF

0x2 ACCESORY

0x4 RUN

0x8 START

0xF INVALID

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Ignition\_Status\_Available**

**Definition** **Indicates whether the Ignition\_Status signal is lost or available.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 LOST: The Ignition\_Status signal was lost or is INVALID or

UNKNOWN. AVAS cannot determine the state of the ignition status.

0x1 AVAILABLE: The Ignition\_Status signal was received and is not

INVALID or UNKNOWN. AVAS can determine the state of the

ignition status.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Ignition\_Status\_Signal\_Received\_Flag**

**Definition** **Dataflow that contains NULL when no Ignition\_Status signal was received**

**and RECEIVED when the signal was received. After checking, AVAS must**

**set the value of this dataflow to NULL.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 NULL: No Ignition\_Status signal was received.

0x1 RECEIVED: The Ignition\_Status signal was received.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **InvalidDataTime\_Cfg**

**Definition** **Delay in milliseconds to wait when receiving faulty or invalid data before**

**setting an Invalid Data DTC.**

Attributes

Units Range Default Number of Values

Milliseconds 0-5000 2100 50

DataFlow Value Description

2100 Delay in milliseconds to wait when receiving faulty or invalid data

before setting an Invalid Data DTC.

**-------------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Ignition\_Status\_TimeLost\_Cfg**

**Definition** **Time to wait when Ignition Status received flag is NULL before**

**transitioning to lost state.**

Attributes

Units Range Default Number of Values

Milliseconds 0-5000 5000 5000

DataFlow Value Description

5000 Delay in mSec before declaring Ignition\_Status signal as lost.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **LoOvrTime\_Cfg**

**Definition** **Defines the amount of time to wait in a low voltage (LO\_V) condition**

**before declaring an UNDER\_V condition.**

Attributes

Units Range Default Number of Values

Milliseconds 0-6000 See FS 600

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **LoRecovTime\_Cfg**

**Definition** **Defines the amount of time that the voltage must be above Vmin\_Cfg[i]**

**before recovering from an UNDER\_V condition and allowing input**

**sampling again.**

Attributes

Units Range Default Number of Values

Milliseconds 0-6000 1000 600

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **LoTime\_Cfg**

**Definition** **After Vbatt dips low, this is the minimum amount of time normal voltage**

**must be present before allowing inputs to be read again (return to**

**NORM\_V). This value must be set based on worst-case analysis of the**

**slowest circuit that uses this Voltage Range.**

Attributes

Units Range Default Number of Values

Milliseconds 0-2000 15 400

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **LowVoltTimer\_Cfg**

**Definition** **Duration that AVAS will wait when it detects voltage as under voltage**

**before it sets a DTC to report the undervoltage event.**

Attributes

Units Range Default Number of Values

Milliseconds 0-10000 10000 10

DataFlow Value Description

10000 Delay in mSec before reporting an undervoltage DTC when first

detecting an undervoltage event.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Max\_Speed\_Sound\_Cfg**

**Definition** **Indicates the MAX speed above which the AVAS ECU will not produce**

**sound.**

Attributes

Units Range Default Number of Values

KPH 0-655.35 655

DataFlow Value Description

0-655.35 Max Speed to play sounds in KPH

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **MediumLostTime\_Cfg**

**Definition** **The medium time duration for detecting a missing CAN signal.**

Attributes

Units Range Default Number of Values

Milliseconds 1000 1000 1

DataFlow Value Description

1000 The medium time duration in milliseconds for detecting a missing

CAN signal.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **PdstrnAlrt\_B\_Falt**

**Definition** **Module's NETCOM TX signal that indicates current status of the module.**

**This signal indicates if the module is working OK. "Yes" indicates**

**module is in failure mode.**

Attributes

Units Range Default Number of Values

Integer Values 0-1 0 2

DataFlow Value Description

0x0 "NO" Module is working okay and not faulted.

0x1 "YES" Module is not working okay and is in failure mode.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **PIDClear\_Time\_Cfg**

**Definition** **Time interval for clearing all PIDCtrl Signals in DEFAULT Session**

Attributes

Units Range Default Number of Values

Seconds 1-5 5 5

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **PwPckTq\_D\_Stat**

**Definition** **NETCOM Signal that indicates power pack is in a motive (wheel torque**

**producing) or non-motive (non-wheel torque producing) mode. This**

**signal is included in message 0x167 "VehicleOperatingModes."**

Attributes

Units Range Default Number of Values

Integer Values 0-3 0 4

DataFlow Value Description

0x0 PwPckOff\_TqNotAvailable

0x1 PwPckOn\_TqNotAvailable

0x2 PwPckStrInProg\_Tq\_NotAvail

0x3 PwPckOnTqAvailable

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **PwPckTq\_D\_Stat\_Signal\_Received\_Flag**

**Definition** **Dataflow that contains NULL when no PwPckTq\_D\_Stat signal was**

**received and RECEIVED when the signal was received. After checking,**

**AVAS must set the dataflow to NULL.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 NULL: No PwPckTq\_D\_Stat signal was received.

0x1 RECEIVED: PwPckTq\_D\_Stat signal was received.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **PwPckTq\_D\_Stat\_TimeLost\_Cfg**

**Definition** **Time to wait when PwPckTq\_D\_Stat received flag is NJLL before**

**transitioning to lost state.**

Attributes

Units Range Default Number of Values

Milliseconds 0-5000 5000 5000

DataFlow Value Description

5000 Delay in mSec before declaring PwPckTq\_D\_Stat signal as lost.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **PwPckTq\_Status\_Available**

**Definition** **Indicates whether the PwPckTq\_D\_Stat signal is lost or available.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 LOST: The PwPckTq\_D\_Stat signal is lost. AVAS cannot

determine the state of torque.

0x1 AVAILABLE: The PqPckTq\_D\_Stat signal is available and AVAS

can determine the state of torque.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **SlowLostTime\_Cfg**

**Definition** **The slow time duration for detecting a missing CAN signal.**

Attributes

Units Range Default Number of Values

Milliseconds 5000 5000 1

DataFlow Value Description

5000 The slow time duration for detecting a missing CAN signal.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Speaker\_OnOff\_Cfg**

**Definition** **This configuration indicates if the module should work with a local switch**

**to allow the user to turn On or OFF the sound of the module.**

Attributes

Units Range Default Number of Values

Integer Values 0-1 0 2

DataFlow Value Description

0x0 Disable Switch

0x1 Enable Switch

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **TesterPhysicalResPACM**

**Definition** **NETCOM Tx to cluster. Indicates AVAS module status.**

Attributes

Units Range Default Number of Values

Integer Value 0-4 0

DataFlow Value Description

0-655.35 Vehicle Speed as reported over CAN

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Trans\_Signal\_Status**

**Definition** **Combined status of GearRverse\_Status and GearPark\_Status.**

Attributes

Units Range Default Number of Values

Integer Value 0-2 0 3

DataFlow Value Description

0x0 LOST: Either GearRvrse\_Status or GearPark\_Status is lost.

0x1 INACTIVE: GearRvrse\_Status is inactive. GearPark\_Status is not

lost.

0x2 ACTIVE: GearRvrse\_Status is active. GearPark\_Status is not lost.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **TransitionDelayCfg**

**Definition** **Delay in mSec before transitioning from playing sound in the front**

**speaker to playing sound in the rear speaker and vice versa.**

Attributes

Units Range Default Number of Values

Milliseconds 0-1000 500 101

DataFlow Value Description

500 Delay in mSec before transitioning from playing sound in the front

speaker to playing sound in the rear speaker and vice versa.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **VBattRaw**

**Definition** **Measured and calculated voltage for Battery Source in Voltage Range**

**Monitor**

Attributes

Units Range Default Number of Values

Volts 0-25.2 12.5 252

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **VBattState**

**Definition** **Array indicating current state of a specific voltage range.**

Attributes

Units Range Default Number of Values

Discrete OVER\_V to NORM\_V 5

UNDER\_V

DataFlow Value Description

HI\_V Temporary High Voltage

LO\_V Temporary Low Voltage

NORM\_V Normal Voltage

OVER\_V Too high too long, Over Voltage

UNDER\_V Too low too long, Under Voltage

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Veh\_V\_ActlEng**

**Definition** **Indicates vehicle speed. This signal is included in message 0x202**

**"EngVehicleSpThrottle."**

Attributes

Units Range Default Number of Values

KPH 0-655.35 0

DataFlow Value Description

0-655.35 KPH

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Veh\_V\_ActlEng\_Signal\_Received\_Flag**

**Definition** **Dataflow that contains NULL when no Veh\_V\_ActlEng signals was**

**received and RECEIVED when the signal was received. After checking,**

**AVAS must set this dataflow back to NULL.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 NULL: No Veh\_V\_ActlEng signal was received.

0x1 RECEIVED: No Veh\_V\_ActlEng signal was received.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Vehicle\_Mode**

**Definition** **Indicates the current state of the vehicle, this state is the result of**

**evaluating powerpack and ignition status.**

Attributes

Units Range Default Number of Values

Integer Value 0-4 3 7

DataFlow Value Description

0x0 ACCESORY

0x1 DIAGNOSTIC MODE

0x2 CRANKING

0x3 IGNITION OFF

0x4 POWERPACK ON

0x5 GEAR PARK

0x6 VEHICLE MODE LOST

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Vehicle\_Speed**

**Definition** **The speed of the vehicle as reported by the PCM/ECM over CAN in KPH,**

**Veh\_V\_ActlEng \* VehicleSpeedMultiplier**

Attributes

Units Range Default Number of Values

KPH 0-655.35 0 0

DataFlow Value Description

0-655.35 The speed of the vehicle as reported by the PCM/ECM over CAN

in KPH, Veh\_V\_ActlEng \* VehicleSpeedMultiplier

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Vehicle\_Speed\_Available**

**Definition** **Set the status communication of the signal indicating if the vehicle**

**speed signal is available in the system.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0 LOST: The Vehicle Speed signal is unavailable.

1 AVAILABLE: The Vehicle Speed signal is available and not faulty.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Vehicle\_Speed\_TimeLost\_Cfg**

**Definition** **Time to wait when the Veh\_V\_ActlEng\_Signal\_Received\_Flag is NULL**

**before declaring the signal as lost.**

Attributes

Units Range Default Number of Values

Milliseconds 0-5000 500 5000

DataFlow Value Description

500 Delay in mSec before declaring Veh\_V\_ActlEng signal as lost.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Vehicle\_Type\_Cfg**

**Definition** **Indicates the brand name of the vehicle.**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 FORD

0x1 LINCOLN

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **VehicleSpeedMultiplier**

**Definition** **The value to multiply Veh\_V\_ActlEng by to get vehicle speed in KPH.**

Attributes

Units Range Default Number of Values

Double Value .01-.01 0.01 1

DataFlow Value Description

0.01 The value to multiply Veh\_V\_ActlEng by to get vehicle speed in

KPH.

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **VehVActlEng\_D\_Qf**

**Definition** **Quality factor for NETCOM signal Veh\_V\_ActlEng**

Attributes

Units Range Default Number of Values

Integer Value 0-3 1 4

DataFlow Value Description

0x0 Faulty

0x1 No data exists

0x2 Not within specifications

0x3 OK

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Vmax\_Cfg**

**Definition** **Defines maximum voltage for a voltage range. Suppliers will perform a**

**worst case analysis of the circuits and supply updated values for the**

**data dictionary to be included in subsequent builds and verification and**

**validation testing.**

Attributes

Units Range Default Number of Values

Numeric 0-25.2 See FS 252

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **VMaxRcv\_Cfg**

**Definition** **Defines Voltage Hysteresis for recovering from the OVER\_V state.**

**Suppliers will perform a worst case analysis of the circuits and supply**

**updated values for the data dictionary to be included in subsequent**

**builds and verification and validation testing.**

Attributes

Units Range Default Number of Values

Numeric 0-25.2 See FS 252

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **Vmin\_Cfg**

**Definition** **Defines minimum voltage for a voltage range. Suppliers will perform a**

**worst case analysis of the circuits and supply updated values for the**

**data dictionary to be included in subsequent builds and verification and**

**validation testing.**

Attributes

Units Range Default Number of Values

Numeric 0-25.2 See FS 252

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **VMinRcv\_Cfg**

**Definition** **Defines Voltage Hysteresis for recovering from the UNDER\_V state.**

**Suppliers will perform a worst case analysis of the circuits and supply**

**updated values for the data dictionary to be included in subsequent**

**builds and verification and validation testing.**

Attributes

Units Range Default Number of Values

Numeric 0-25.2 See FS 252

**----------------------------------------------------------------------------------------------------------------------------------**

**DataFlow Name** **VoltBelowThresholdDelayCfg**

**Definition** **The time to wait before setting an undervoltage DTC.**

Attributes

Units Range Default Number of Values

Seconds 0-15 10 15

DataFlow Value Description

10 The delay in seconds to wait before setting an undervoltage DTC

once an undervoltage event has been detected.

**----------------------------------------------------------------------------------------------------------------------------------**

## Revision History

### Version 2.0 (Author: Peter Sripinyo 7/30/2019) FS-ML3T-14G113-AB

|  |  |  |
| --- | --- | --- |
| DATE | SUMMARY OF CHANGES | CREATOR/REVIEWER |
|  | Initial Release  Change sound range from 0 to 50kph to 0 to 655 kph  **2.8 Generate AVAS Output** updated with new NVH requirements  **2.8.3 Signal Generation Block Requirements** section added  **2.8.4 Filter Block Requirements** section added  **2.8.5 One Speaker Systems versus Multi-Speaker System** subsection number change  **2.8.6 AVAS Output Timing Requirements** section added requirements from 1.6.3  Added missing values to Vehicle Mode dataflow as per Supplier input  **2.10.8.6 DTC Definitions** add continuous DTCs/timing for faults | Peter Sripinyo |
| 7/31/2019 | **Release AB (Version 2.0)**  **2.10.8** Self-Test DTCs labeled as continuous changed to Self-test/Null (issue 21)  **2.0** OTA functionality for GEN III AVAS\_PACM (issue 23)  **2.10.8.7** Change DTC Max action for Lost Comms to 4 seconds (issue 25)  **2.0, Data Dictionary** Add invalid Data from DTC for PCM/ECM (issue 26)  **2.7 Determine AVAS State** Add check for gear position PARK | Peter Sripinyo |