

Approaching Vehicle Audible System

(AVAS)

FUNCTIONAL SPECIFICATION – Phoenix System

FSR2HT-14G113-AA

Version: 1.5 DRAFT

Date Revised: 5/30/2022

Author: A. Saminathan

**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 | | | | | | | | | | |
| PART NAME | | | | | | | | | | PART NUMBER | | | | | | |
| Functional Specification – AVAS Feature – Phoenix System | | | | | | | | | | FSR2HT-14G113-AA | | | | | | |
|  | | LET | | FR | | REVISIONS | | | | | | DR | CK | REFERENCE | | |
| 5/30/2022 | | DR | | All | | FSR2HT-14G113-AA | | | | | |  |  |  | | |
|  | |  | |  | |  | | | | | |  |  | PREPARED BY | | |
|  | |  | |  | |  | | | | | |  |  | A.Saminathan | | |
|  | |  | |  | |  | | | | | |  |  | APPROVED BY | | DETAILED BY |
|  | |  | |  | |  | | | | | |  |  | AS | | AS |
|  | |  | |  | |  | | | | | |  |  | CONCURRENCE/APPROVAL | | |
|  | |  | |  | |  | | | | | |  |  | SIGNATURES | | |
|  | |  | |  | |  | | | | | |  |  | Design Engineering Supervisor | | |
|  | |  | |  | |  | | | | | |  |  |  | | |
|  | |  | |  | |  | | | | | |  |  | Design Engineering Management | | |
|  | |  | |  | |  | | | | | |  |  |  | | |
|  | |  | |  | |  | | | | | |  |  | 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](#_Toc103768316)

[Content 2](#_Toc103768317)

[1. DOCUMENT OVERVIEW 4](#_Toc103768318)

[1.1 DOCUMENT OVERVIEW 4](#_Toc103768319)

[1.2 Purpose and Scope 4](#_Toc103768320)

[1.2.1 Conflict of Documentation 4](#_Toc103768321)

[1.3 DOCUMENT ROAD MAP 4](#_Toc103768322)

[1.4 DOCUMENT CONVENTIONS 5](#_Toc103768323)

[1.4.1 Methodology & Data Flow Diagrams 5](#_Toc103768324)

[1.4.2 Requirements Representations 5](#_Toc103768325)

[1.4.2.1 Feature Functional Requirements 8](#_Toc103768326)

[1.4.3 Glossary of Terms 9](#_Toc103768327)

[1.4.4 Phoenix Domain Controller and Phoenix Audio Controller Interface to NVRAM Manager 11](#_Toc103768328)

[1.4.5 Timer Mark Event / Time Since Event 11](#_Toc103768329)

[1.5 RELATED DOCUMENTS 13](#_Toc103768330)

[1.5.1 PARSED 16](#_Toc103768331)

[1.5.2 Cybersecurity 16](#_Toc103768332)

[1.6 General Requirements 16](#_Toc103768333)

[1.6.1 Module - Memory / Power Up / Microcontroller Reset 16](#_Toc103768334)

[1.6.1.1 Microcontroller memory Storage Classification Requirements: 16](#_Toc103768335)

[1.6.1.2 Non-volatile Memory Requirements: 17](#_Toc103768336)

[1.6.1.3 Module Power Up / Microcontroller Reset Requirements: 17](#_Toc103768337)

[Timing Requirements 18](#_Toc103768338)

[1.6.1.4 Timing / Response Requirements 18](#_Toc103768339)

[1.6.2 Order of Execution 18](#_Toc103768340)

[1.6.2.1 Order of Execution Requirements 18](#_Toc103768341)

[2 APPROACHING VEHICLE AUDIBLE SYSTEM (AVAS) 20](#_Toc103768342)

[2.1 Feature Behavior Summary 20](#_Toc103768343)

[2.2 Subsystem Overview 21](#_Toc103768344)

[2.3 Feature Behavior Detail 22](#_Toc103768345)

[2.4 Master VIN List/CAN Signal Relationship Table 23](#_Toc103768346)

[2.4.1 Handling Quality Factor 23](#_Toc103768347)

[2.5 Determine Sound State 24](#_Toc103768348)

[2.6 Generate AVAS Output 26](#_Toc103768349)

[2.6.1 General AVAS Output Requirements 26](#_Toc103768350)

[2.6.2 Time to first audio 27](#_Toc103768351)

[2.6.3 One Speaker Systems versus Multi-Speaker Systems 28](#_Toc103768352)

[2.6.4 AVAS Output Timing Requirements 28](#_Toc103768353)

[2.7 Diagnostics 30](#_Toc103768354)

[2.6.5 Overview 30](#_Toc103768355)

[2.6.6 Supported Diagnostic Identifiers 30](#_Toc103768356)

[2.6.7 AVAS Routine Control (0x31) Service 31](#_Toc103768357)

[2.6.8 AVAS Data Identifiers (DIDs) 31](#_Toc103768358)

[2.6.8.1 ReadDataByIdentifer (0x22) Service 31](#_Toc103768359)

[2.6.8.2 Diagnostic Trouble Codes General Requirements 33](#_Toc103768360)

[DTC Definitions 33](#_Toc103768361)

[2.6.8.3 DTC Definitions and Requirements Table 34](#_Toc103768362)

[2.6.9 DTC F00006 - AVAS Failure 36](#_Toc103768363)

[AVAS Configuration Data (Method 2 & 3) 37](#_Toc103768364)

[2.6.9.1 A note about Method 2 data versus Method 3 data 37](#_Toc103768365)

[2.6.9.2 Method 2 Data 37](#_Toc103768366)

[2.6.9.3 Method 2 Configuration Parameters 38](#_Toc103768367)

[2.6.9.4 AVAS Configuration Data (Supplier Range FD00-FEFF) 38](#_Toc103768368)

[2.7 Data Dictionary 40](#_Toc103768369)

[2.7.1 Version 1.0 (Author: Arun Saminathan) FSR2HT-14G113-AA 53](#_Toc103768370)

[2.10.5.3 – A note about Method 2 data versus Method 3 data – removed max\_speed\_sound\_cfg after discussion with the NVH team and SW architects 54](#_Toc103768371)

[2.10.5.3 – A note about Method 2 data versus Method 3 data – included max\_speed\_sound\_cfg as a method 3 config rather than a DE bit or DID. 54](#_Toc103768372)

[Removed “invalid\_timelost\_cfg” from all state machines 55](#_Toc103768373)

[Updated Section 2.7 from AVAS State to Sound State 55](#_Toc103768374)

# DOCUMENT OVERVIEW

## DOCUMENT OVERVIEW

Approaching Vehicle Audible System Module Author Team

|  |  |
| --- | --- |
| SW D&R | Core D&R |
| Karthick Lakshmanan | A. Saminathan |
|  |  |

## Purpose and Scope

This document provides detailed requirement descriptions of Approaching Vehicle Audible System (AVAS) subsystem operating within the Phoenix Audio System.

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 subsystem. It also contains general requirements for the features in the AVAS subsystem and references to related documents. |
| **Section 2.0** | This section provides the detailed specification of all the features for the AVAS subsystem. The subsections in Section 2.0 detail an overview of the AVAS subsystem as well as the specific requirements on how AVAS determines whether to play a sound and whether or not the AVAS subsystem 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.

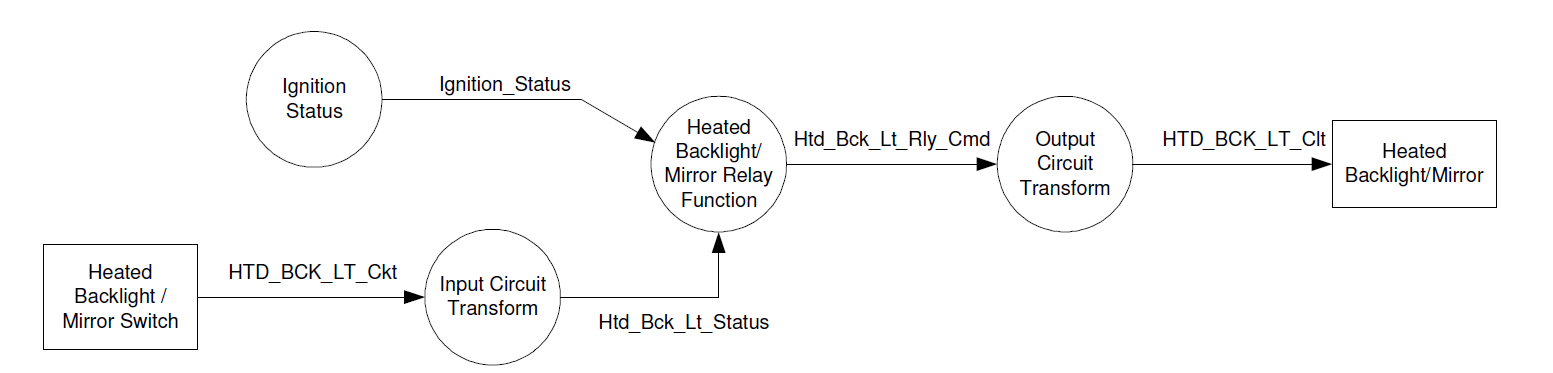


Figure ‑ Context Diagram Illustration

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 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 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***

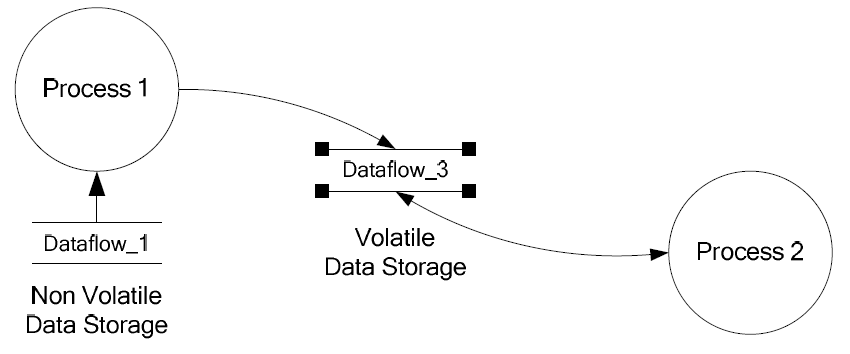


Figure ‑ Example of Detecting a Transition

**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**

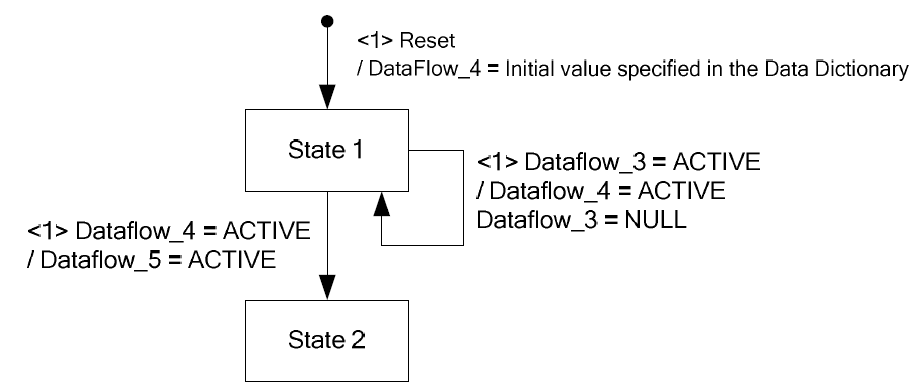
#### Feature Functional Requirements

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 |
| PDC | Phoenix Domain Controller |
| PAC | Phoenix Audio Controller |
| 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) |

### Phoenix Domain Controller and Phoenix Audio Controller 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. PDC and PAC uses the UFS 2.1 for Non-volatile memory.

**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 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 | FSR2HT-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 | FSR2HT-14G113-AA Data Dictionary V 1.0 | 13 OCT 2021 |
|  | 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 are covered in the following IDs)  RQT-001403-020617 (02-0065)  RQT-001403-020626 (02-0069)  RQT-001403-020627 (02-0070)  RQT-001403-020628 (02-0071)  RQT-001403-020629 (02-0072)  RQT-001403-020655 (02-0073)  RQT-001403-020656 (02-0074)  RQT-001403-020657 (02-0075)  RQT-001403-020658 (02-0076)  RQT-001403-020660 (02-0078)  RQT-001403-020661 (02-0079)  RQT-001403-020665 (02-0080)  RQT-001403-020666 (02-0081)  RQT-001403-020667 (02-0082)  RQT-001403-020668 (02-0083)  RQT-001403-020669 (02-0084)  RQT-001403-020671 (02-0086)  RQT-001403-020672 (02-0087) | 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. The Cybersecurity requirements and Statement of work are already covered for the Phoenix Architecture system.

## General Requirements

|  |  |
| --- | --- |
| **Applicability Matrix – Section 1.6.x** | |
| Module Name | Applicable |
| PDC | Yes |
| PAC | Yes |

### 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 | |

### 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/pedestrian’s 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.

**Enabling/Disabling sound**:

When AVAS sound is enabled or disabled, this action takes place only in PDC, where the sound generation is muted or unmuted. In other words, when AVAS sound is disabled, the A2B slots reserved for AVAS will carry PCM samples consisting of all zero values. PAC amplifier channels remain umuted during this disable or enable action.

**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

|  |  |
| --- | --- |
| **Applicability Matrix – Section 2.3** | |
| Module Name | Applicable |
| PDC | Yes |
| PAC | No |

The AVAS module is a component of electric vehicles EV, PHEV or HEV. When the vehicle’s powertrain pack 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 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 should be functionally deactivated (does not produce sound, does not process any feature functionality, does not process faults) when the ignition is off.

## Master VIN List/CAN Signal Relationship Table

The relationship between vehicle signals and the associated VINs are captured in a reference document “NVH Features Master CAN signal and VIN id list” excel spreadsheet.

The spreadsheet is organized in the following tabs:

* AVAS CAN signals
  + This table captures the vehicle CAN signals required for the feature to function.
  + Each signal is associated with a specific DE configuration which corresponds to a specific sub feature AVAS.
  + The NVH Service will subscribe to the corresponding enabled vehicle signals from vehicle\_signal component.
  + Missing message DTC(s) should be set for enabled signals which exceed message timeout condition.
* Master VIN list
  + This table captures the VIN ID for each CAN signal and it’s corresponding LiveAMP name.
  + The relationship between the VIN ID and the corresponding CAN signal(s).
  + Feature fault response on a per signal basis.
  + Derived VINs within the LiveAMP configuration to be read by the NVH service.
  + VINs related to Diagnostic modes
  + Reserved VINs for potential future changes
* Logical Variables

This table captures the logical relationship for NVH Service generated VINs.

## 2.4.1 Handling Quality Factor

Whenever the accuracy of a CAN signal is represented by its own Quality Factor signal, then the value of the CAN signal shall be considered as accurate only when the Quality factor signal is in “OK” state. The NVH Service will only convey updated VIN values when the corresponding Quality Factor is equal to “OK”. A different state (other than “OK”) shall render the associated CAN signal as being inaccurate; and the NVH Service shall not convey or update the corresponding VIN value; thus, the system shall continue to operate on its last known state.

When the Quality Factor is NOT “OK” for more than 5 seconds, then the system shall consider the CAN signal is missing and shall take the appropriate action. If the Quality Factor changes state back to “OK” within this timeout period, then the system shall continue its normal operation.

## Determine Sound State

|  |  |
| --- | --- |
| **Applicability Matrix – Section 2.7** | |
| Module Name | Applicable |
| PDC | Yes |
| PAC | Yes |

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. The maximum speed depends on the value of Max\_speed\_Sound\_Cfg which defines the maximum speed above which the ECU will not produce sound is considered as a “Method 3 configuration” rather than a DE bit or DID.

The subsystem 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 Failures are detected within PAC and speaker fault status information is communicated to PDC over CAN*

* + - Speaker#2 Failure (Generally the rear speaker if it exists on the hardware) \*

*\*Speaker Failures are detected within PAC and speaker fault satus information is communicated to PDC over CAN*

Speaker fault status needs to be managed within NVH services whereas it is the PAC’s responsibility to capture the Speaker faults.

* + - Amplifier failure\*\*

\*\**Amplifier Failures are detected within PAC and amplifier fault satus information is communicated to PDC over CAN*

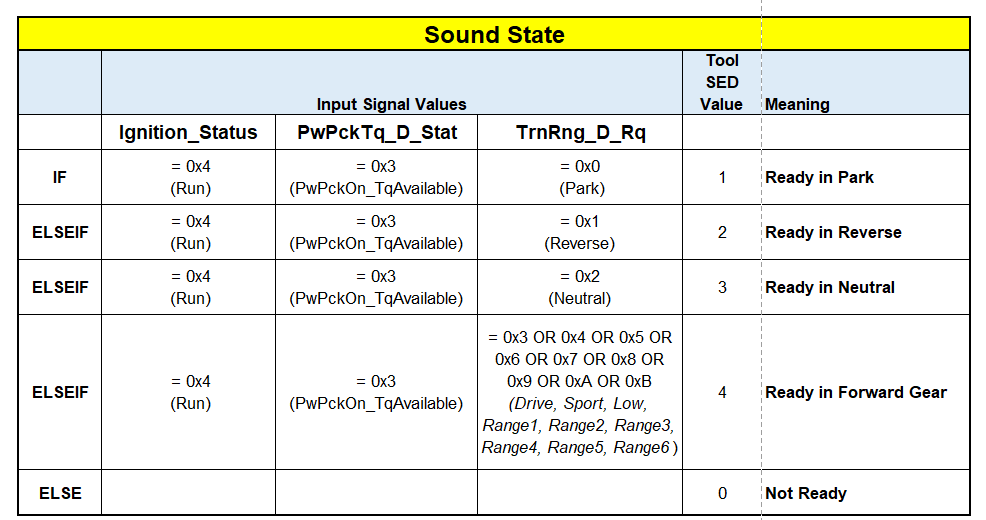
* + - Unknown state
    - Lost communication or signal reported as faulted
    - Internal failures

For more detailed information on specific failures refer to **2.10 Diagnostics**.When the failures above are present, the PDC module should set AVAS\_State to FAULT. DID EE08 represents the AVAS\_State. Size: 1 Byte SED

Values:

* 0x00– AVAS Inactive (AVAS not Producing Sound)
* 0x01 – AVAS Producing Sound
* 0x02 – AVAS is Faulted and Cannot Produce Sound
* 0x03 – 0xFF - Reserved

The below pic represents the internal status of the AVAS module represented with the dataflow AVAS\_State and applies to PDC.



## Generate AVAS Output

### General AVAS Output Requirements

|  |  |
| --- | --- |
| **Applicability Matrix – Section 2.9.1** | |
| Module Name | Applicable |
| PDC | Yes |
| PAC | No |

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. | |

### Time to first audio

AVAS Audio Components shall transition from AVAS Standby/Sleep state to AVAS Functional state (irrespective of Normal, Transport, Factory or during a Load Shed event) within 2 seconds of the Ignition\_Status signal transitioning to Run.

### One Speaker Systems versus Multi-Speaker Systems

|  |  |
| --- | --- |
| **Applicability Matrix – Section 2.9.3** | |
| Module Name | Applicable |
| PDC | Yes |
| PAC | No |

The PDC will communicate to PAC about the Speaker selection via “SpkrSel\_D\_Rq” and 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

|  |  |
| --- | --- |
| **Applicability Matrix – Section 2.9.3** | |
| Module Name | Applicable |
| PDC | Yes |
| PAC | Yes |

| **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 AVAS audio shall mute during the transition and unmute once the transition is complete unless AVAS enters 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 | |

## Diagnostics

This section is already referenced/covered in PDC and PAC specifications respectively.

### 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

|  |  |
| --- | --- |
| **Applicability Matrix – Section 2.9.3** | |
| Module Name | Applicable |
| PDC | Yes |
| PAC | Yes |

Table ‑ General Diagnostic Requirements

|  |  |
| --- | --- |
| **ECU Information** | |
| ECU Name: | Phoenix Domain Controller |
| ECU Acronym: | PDC |
| Diagnostic Communication Link: | HS3-CAN |
| ECU Diagnostic Reception ID: | 7D0 |
| ECU Diagnostic Transmission ID: | 7D8 |
| Functional Diagnostic ID: | 7DF |
| Network Initialization ID: |  |
| **Specification Versions** | |
| GGDS Specification Version: | 004 |
| SWDL Specification Version: | 006 |
| ECU Configuration Version: | 002 |
|  |  |
| **Miscellaneous** | |
| Bootloader Supported | Yes |
| **ECU Information** | |
| ECU Name: | Phoenix Audio Controller |
| ECU Acronym: | PAC |
| Diagnostic Communication Link: | HS3-CAN |
| ECU Diagnostic Reception ID: | 727 |
| ECU Diagnostic Transmission ID: | 72F |
| Functional Diagnostic ID: | 7DF |
| Network Initialization ID: |  |
| **Specification Versions** | |
| GGDS Specification Version: | 005 |
| SWDL Specification Version: | 007 |
| ECU Configuration Version: | 004 |
|  |  |
| **Miscellaneous** | |
| Bootloader Supported | Yes |
|  |  |

### AVAS Routine Control (0x31) Service

This section is already referenced/covered in PDC and PAC specifications respectively.

### 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.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Rqmt. No.** | **DID** | **DID Name/Description** | **Config Reqts** | **Dataflow** | **01** | **02** | **03** |
| **R: 2.10.101** | 0xF17D | ECU Cal-Config Part Number (calibration for AVAS) | n/a | Supplier Defined | R | - | R |

##### 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.

#### 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 | 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 & TrnRng\_D\_Rq\_Signal\_Received\_Flag = NULL & Ignition\_Status < > OFF | 127 | DTCMaxAction(),DTCSTest[].MonitorMAX=MAX | GearRverse\_Status\_Available= Available | 128 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.147 | 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 | Vehicle\_speed\_Available= Lost & Veh\_V\_Actl\_Signal\_Received\_Flag = NULL & Ignition\_Status < > OFF | 127 | DTCMaxAction(),DTCSTest[].MonitorMAX=MAX | Vehicle\_Speed\_Available= Available | 128 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.148 | 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 | PwPckTq\_Status\_Available= Lost & PwPckTq\_D\_Stat\_Signal\_Received\_Flag = NULL & Ignition\_Status < > OFF | 127 | DTCMaxAction(),DTCSTest[].MonitorMAX=MAX | PwPckTq\_Status\_Available= Available | 128 | DTCSTest[].MonitorMIN=MIN |
| R: 2.10.149 | 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 & Ignition\_Status < > OFF | 32 | DTCMaxAction() | Ignition\_Status\_Available = Available | 16 | DTCMinAction() |
| R: 2.10.151 | 0xE02951 | 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 AVAS/PS/ANC Calibration file is missing or corrupted when AVASor Propulsion Sound or ANC is configured as enabled. (CCPU) | 127 | DTCMaxAction(),DTCSTest[].MonitorMAX=MAX | Set when calibration data area is valid. | 128 | DTCSTest[].MonitorMIN=MIN |

### DTC F00006 - AVAS Failure

Separate fault to indicate the AVAS feature is not functional. This may be set in conjunction with other ANC/PS/AVAS related faults that can aid in determining cause. Further status can be obtained by reading NVH Service Status DID.

The AVAS feature will be non-functional/muted when this fault is active.

### 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. Max\_speed\_sound\_cfg is the only Method 3 data.

Method 2 parameters in this case are used to enable/disableAVAS feature in the PDC, to determine if the vehicle has internal (PDC) or external AVAS, to figure out the number of speakers in actual hardware and the Transitiondelaycfg parameter (which will now become the delay for NVH service to allow release of looping sound prior to speaker switch in case of a 2-speaker configuration). TransitionDelayCfg is used to ensure that audio is played back in the proper speaker during a transition. This parameter will be used ‘only’ for a two-speaker system when the AVAS audio is switched between front to rear or vice versa regardless oof the reason for transition be it either a gear position switch or speaker fault. However, for a two-speaker system when the gear is switched from park to either drive/reverse or vice versa, the AVAS event client shall not apply ‘TransitiondelayCfg’ between speaker directionality switch and generator rather this would be taken care internally by the library.

#### 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** |
| **DE00 (PAC)**  **DE0A (PDC)**  **(Byte 7 Bit 5)** | **Internal-External AVAS configuration type** | **Configuration parameter to determine if the vehicle has internal (PDC) or external AVAS. These are both Readable and Writable.** | **Internal enabled/ External disabled** | **CNT** | **YES** | **YES** | **TBD** | **NO** |
| **DE0A (Byte 4 Bit 6)** | **Enable/Disable AVAS** | **Configuration parameter to enable/disable AVAS in the vehicle** | **Enable/Disable** | **CNT** | **YES** | **YES** | **TBD** | **NO** |
|  | | | | | | | | |

#### 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** | AVAS\_Spkr\_Cfg (DE0A Byte 5 Bit 1) | Number of Speakers in actual hardware |  |  |
| **R: 2.10.207** | TransitionDelayCfg (EE0A) | TransitionDelayCfg is used to ensure that audio is played back in the proper speaker during a transition. This will now become the delay for NVH service to allow release of looping sound prior to speaker switch (in case of a 2-speaker configuration) This parameter will be used ‘only’ for a two-speaker system when the AVAS audio is switched between front to rear or vice versa regardless oof the reason for transition be it either a gear position switch or speaker fault. However, for a two-speaker system when the gear is switched from park to either drive/reverse or vice versa, the AVAS event client shall not apply ‘TransitiondelayCfg’ between speaker directionality switch and generator rather this would be taken care internally by the library. | Milliseconds |
| **R: 2.10.208** | 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 | 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 # 2 | | | | | |
| Parameter Name | | TransitionDelayCfg | | | |
| Description | | TransitionDelayCfg is used to ensure that audio is played back in the proper speaker during a transition. This will now become the delay for NVH service to allow release of looping sound prior to speaker switch (in case of a 2-speaker configuration) | | | |
| DataIdentifier Size (bits) | | *10* | | | |
| Format Information: | | | | | |
| **Size (bits)** | **Parameter Info** | | **Units** | **Min Scaled Value** | ***Max Scaled Value*** |
| *10* | TransitionDelayCfg | | *msec* | *0* | *1000* |

## Data Dictionary

AVAS

FSR2HT-14G113-AA

Data Dictionary v2.0

Wednesday, May 18, 2022 Revision 1.5

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

**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 (captured in IDS)**

Attributes

Units Range Default Number of Values

Integer Value 0-1 0 2

DataFlow Value Description

0x0 1 Speaker Configuration

0x1 2 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** **DiagSession**

**Definition** **Indicates the active session of Diagnostics (expectation is that PDC should handle this)**

Attributes

Units Range Default Number of Values

Discrete DEFAULT, UNKNOWN 4

EXTEND,

PROGRAM,

UNKNOWN

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

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

**Definition** This value is LOST when the TrnRng\_D\_Rq 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: TrnRng\_D\_Rq is missing, or its value is Fault.

0x1 AVAILABLE: TrnRng\_D\_Rq is present, and its value is not

Fault.

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

**DataFlow Name** **TrnRng\_D\_Rq**

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

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

**signal is in 0x176 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** **TrnRng\_D\_Rq\_Signal\_Received\_Flag**

**Definition** **Dataflow that contains NULL when no TrnRng\_D\_Rq 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 TrnRng\_D\_Rq signal was received.

0x1 RECEIVED: A TrnRng\_D\_Rq signal was received.

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

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

**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: TrnRng\_D\_Rq 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 TrnRng\_D\_Rq 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** **TrnRng\_D\_Rq**

**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** **TrnRng\_D\_Rq\_Signal\_Received\_Flag**

**Definition** **Dataflow that contains NULL when no TrnRng\_D\_Rq 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 TrnRng\_D\_Rq signal was received.

0x1 Received: A TrnRng\_D\_Rq signal was received.

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

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

**Definition** **Indicates current ignition. This signal is in 0x3B2 message "BodyInfo\_3\_HS3" (captured in IDS)**

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. AVAS cannot determine the state of the ignition status.

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

INVALID. 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** **PdstrnAlrt\_B\_Fault**

**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** **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\_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** **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** TransitionDelayCfg is used to ensure that audio is played back in the proper speaker during a transition. which will now become the delay for NVH service to allow release of looping sound prior to speaker switch (in case of a 2-speaker configuration)

Attributes

Units Range Default Number of Values

Milliseconds 0-1000 500 101

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

**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\_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** **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

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

Revision History

### Version 1.0 (Author: Arun Saminathan) FSR2HT-14G113-AA

|  |  |  |
| --- | --- | --- |
| DATE | SUMMARY OF CHANGES | CREATOR/REVIEWER |
| 10/13/2021 | Initial Release  1.5.2 Cybersecurity requirements  2.1 Feature Behavior summary - Feature Behavior Detail Updated wording to match ignition dependent strategy  2.5 Determine Vehicle Mode Status Removed Table 2.5-1 and related text  2.5 Updated Table 2.5-4 Determining Vehicle Mode  2.7 Determine Sound State Updated  2.8.1 AVAS Fault State Signal Updated wording on the transmission of PdstrnAlrt\_B\_Falt - 2.8.5 Updated One speaker vs Multi-speaker system  2.9 Voltage Range Monitor  2.10 Diagnostics - 2.10.5.2 DTC definitions and Requirements  2.10.7 CAN based data identifiers  2.11.8.7 DTC Definitions and Requirements Table Updated R:2.10.162 and R:2.10.163 to require Ignition\_Status <> OFF for increment criteria | Arun Saminathan |
| 12/7/2021 | 2.10.5.3 – A note about Method 2 data versus Method 3 data – removed max\_speed\_sound\_cfg after discussion with the NVH team and SW architects 2.7 – Determine Sound state - removed a note on max\_speed\_sound\_cfg | Arun Saminathan |
| 2/10/2022 | 2.2 Subsystem Overview – removed Battery voltage monitor as these are handled by the PDC and do not apply anymore  2.4 Determine Transmission status – updated/captured “TransitionDelayCfg” as a Supplier DID (EE0A)  2.7 Determine Sound state – Updated the Speaker fault status  2.8.2 – Removed the Master Switch content that is applicable only to the AVAS Standalone module  2.8.5 – Aligned with the AVAS SPSS and removed the acknowledgement response from PAC as it is no longer needed  2.8.6 – R:2.8.17 – Rephrased the requirement for more clarity  2.10.4.1.2 – Supported DID list – marked the DIDs that are for calibration and specific to AVAS  Removed section 2.10.4.1.3 since that is applicable only for the standalone AVAS module  2.10.5.1 – Updated the new DTC#s, E01B51 replaced with E02951, E01B41 replaced with E02951 and F00042 replaced with F00004  2.10.5.3 – Added all the method 2 configuration parameters  2.10.5.5 – Updated the method 2 configuration parameters table  2.10.5.6 – Updated the AVAS configuration data | Arun Saminathan |
| 2/17/2022 | 2.10.5.3 – A note about Method 2 data versus Method 3 data – included max\_speed\_sound\_cfg as a method 3 config rather than a DE bit or DID. 2.7 – Determine Sound state - included a note on max\_speed\_sound\_cfg as a method 3 config rather than a DE bit or DID.  Figure 2.4-2 – added DTC information for GearRevrse\_Signal\_Status  Figure 2.4-5 – added DTC information for GearLvrPos\_Available  Figure 2.5-2 – added DTC information for PwPckTq\_Status\_Available  Figure 2.5-3 – added DTC information for Ignition\_Status\_Available  Figure 2.6-2 – added DTC information for Vehicle\_Speed\_Available  2.10.5 – added DTCs 0xC10000 for GearRevrse\_Signal\_Status, 0xC10100 for GearLvrPos\_Available, 0xC10000 for PwPckTq\_Status\_Available, 0xC14000 for Ignition\_Status\_Available, 0xC10000 for Vehicle\_Speed\_Available | Arun Saminathan |
| 3/23/2022 | Removed “invalid\_timelost\_cfg” from all state machines Replaced legacy signal GearLvrPos\_D\_Actl with TrnRng\_D\_Rq  Replaced GearRvrse\_D\_Actl with TrnRng\_D\_Rq because this signal will be used for determination of gear  Section 2.7 – Determine Sound state – added DID information for AVAS\_state  Removed CAN based Data identifiers  Repuprose TransitionDelayCfg which will now become the delay for NVH Service to allow release of looping sound prior to speaker switch in case of a 2-speaker configuration  Reworded Max\_Speed\_cfg which will now be a “Method 3” config | Arun Saminathan |
| 4/19/2022 | Updated Section 2.7 from AVAS State to Sound State Reworded TransitionDelayCfg | Arun Saminathan |
| 5/18/2022 | Removed all State diagrams and included Master VIN list, handling Quality Factor sections | Arun Saminathan |