Research & Vehicle Technology
“Infotainment Systems Product Development”

Infotainment System Model Specification

UNCONTROLLED COPY IF PRINTED

Version Date: September 27, 2016

FORD CONFIDENTIAL

Table of Contents

1 SPSS Document Overview

1.1 General

1.1.1 Application of Document

This document defines how to read and implement the SPSS specifications and includes high level block diagrams for the different Infotainment architectures.

1.1.2 Versions and Releases

The SPSS is updated frequently and released periodically. Please consult the release notes at the beginning of the SPSS for a list of changes/updates made between each version/revision.

1.1.3 Conflict of Documentation

Any conflict of documentation should be brought to the attention of the Ford Infotainment team for proper resolution.

1.2 Purpose, Scope, and Style

The Subsystem Part Specific Specification (SPSS) describes the distributed functionality of infotainment features. This specification is organized by feature chapters that detail the system elements/interactions required to implement each feature.

This specification is purposely authored in a generic style so that it can be applied to multiple physical architectures/configurations. Rather than using actual module/signal names, system elements are represented as generic server/client objects, and functional relationships between them are described using logical interfaces. Many of the description elements used in the SPSS are artifacts of a UML/SysML model-based development process. The generic system structure can then be mapped to the desired physical/implementation architecture using supplemental documents.

The *Description Elements* section of this chapter outlines the descriptive elements used in this document, and the *Preparing for Implementation* section provides information on mapping generic system objects to implementation architecture.

Note: The SPSS may be released as a single module based SPSS specification with the different features in one specification or as separate feature based SPSS specifications specific for a module.

IMPORTANT NOTE

The SPSS includes many features with different complexity and legacy. As a result, the chapter structure may vary slightly by feature, and not every feature will include every description element outlined in this chapter. However, each feature contains an ample amount of specification to successfully implement its distributed functionality.

1.3 SPSS Description Elements

The SPSS includes many description elements that collectively define the distributed functionality of an infotainment feature. This section details the common elements.

1.3.1 Requirements

SPSS features are broken down into individual requirements to support a feature.

SPSS requirements whether they are system requirements, functional requirements, sequence diagram requirements, HMI requirements etc will have associated implementation guides which will say whether individual SPSS requirements apply or not for a specific module.

Timing Requirements:
Throughout the SPSS there will be timing requirements with an example shown below. The Range and Resolution is the range and resolution the software should be able to support but the default value is the specific value to be used to implement a feature.

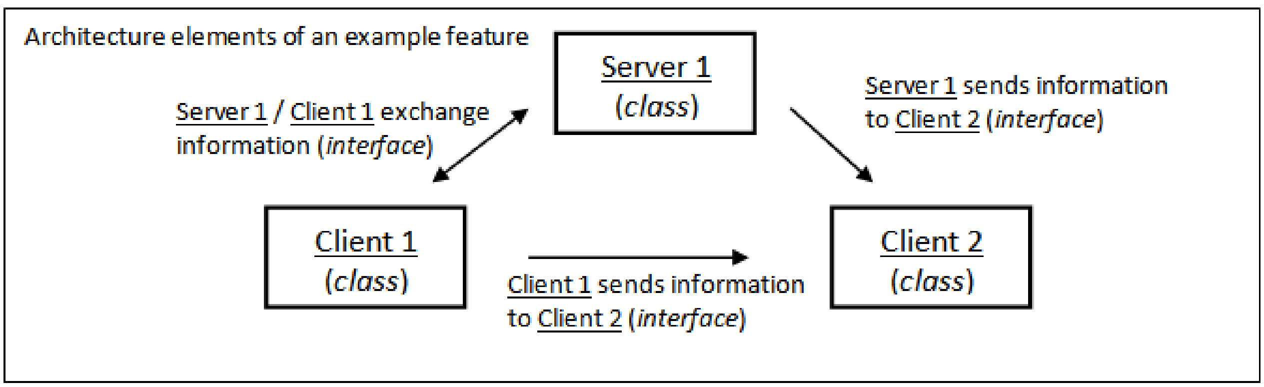
An example timing requirement is below.

Name	Description	Units	Range	Resolution	Default
T_Feature_Response	The maximum time from when feature X request is received until a feature X response is put on the bus	msec	0-1000	5	75

For this example the module that receives feature x request must respond with the feature x response on the network bus within 75 msec from the time of receiving the request.

1.3.2 Architecture

The system architecture of each feature are usually defined using classes and class interfaces. These objects are fundamental system building blocks and frequently referenced in the specification. A conceptual drawing of these objects for an example feature is shown below.



1.3.2.1 Class Definitions

Each feature includes server and client classes. These classes are based on a logical partitioning of feature functions/operations, and work jointly to allow a distributed feature to act as a single unit. Servers typically handle core logic, while clients perform ancillary operations such as user interactions, secondary calculations, and coordination to other parts of the infotainment system. Note that the actual physical implementation may contain multiple classes realized within a single module. Also, a feature is not required to have both a server and client class.

Note that the terms client/server and master/slave are used interchangeably in this document. Master/slave is the legacy name and client/server is used for new features.

1.3.2.2 Interface Definitions

Class interfaces (also called *logical signals* or *ports and interfaces*) describe the logical information flow between server/client classes. A server/client class may transmit, receive, or both transmit/receive logical signals. Additionally, logical signals may contain multiple parameters/methods.

An example a way a signal could be defined in the SPSS:

FEATURE-IIR-REQ-XXX-Feature Interface Table

Method	Notes	Parameters
FeatureSignal_Rq	A Request signal sent from the Feature Client requesting an action from the Feature Server	0x0 Null 0x1 Request State X 0x2 Request State Y 0x3 Reserved
FeatureSignal_Rsp	A Response signal from the Feature Server indicating of the requested action was accepted or not	0x0 Null 0x1 Accepted 0x2 Denied
FeatureSignal_St	A status signal from the Feature Server indicating the status of the feature	0x0 Inactive 0x01 State X 0x02 State Y 0x3 Reserved

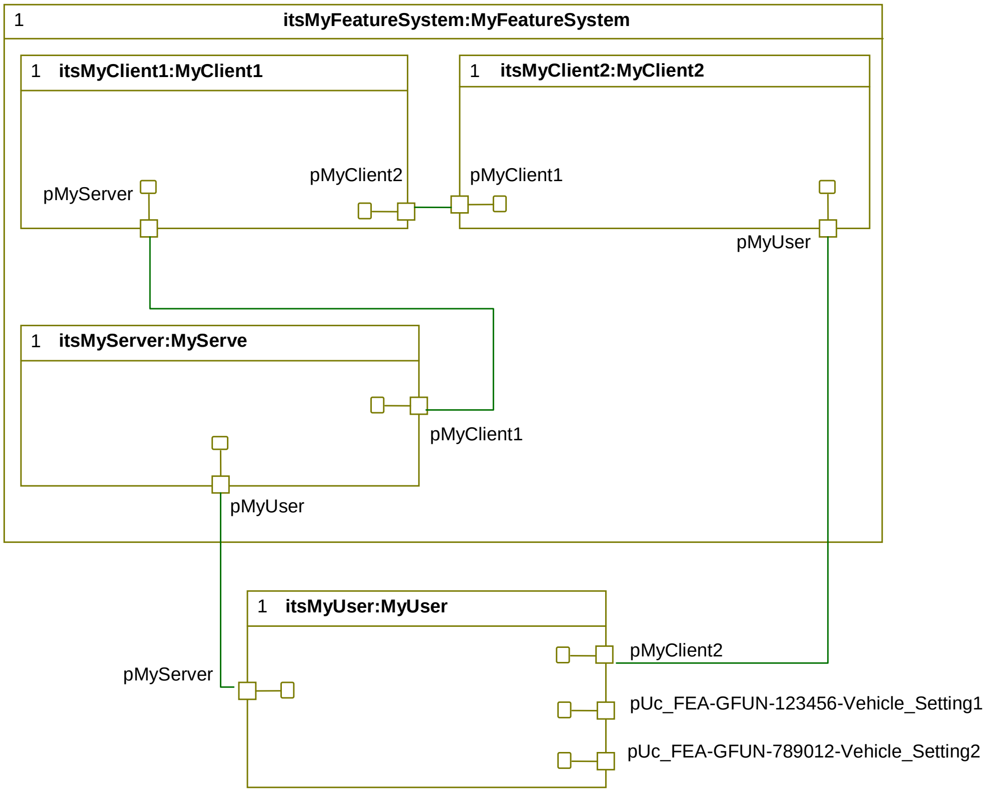
An example interface definition is shown below, where logical signal *MyFeature_St* contains two parameters/methods: *Status*, *MinPower*:

Interface Definition: MyFeature_St

Name	Literals	Value	Description
Status	-	-	Indicates if MyFeature is active.
	Invalid	0x0	
	OFF	0x1	
	ON	0x2	
MinPower	-	-	Current acceptable Minimum Power in Watts (0-100)
	0	0x0	
	1	0x1	
	…	…	
	100	0x64	
	Invalid	0xFF	

1.3.2.3 Internal Block Diagrams

Internal Block Diagrams provide a graphical representation of the system architecture. This diagram shows the general topography of system classes and their interfaces (ports/connections), but does not detail the quantity or direction of data managed through the interfaces. An example internal block diagram is shown below.

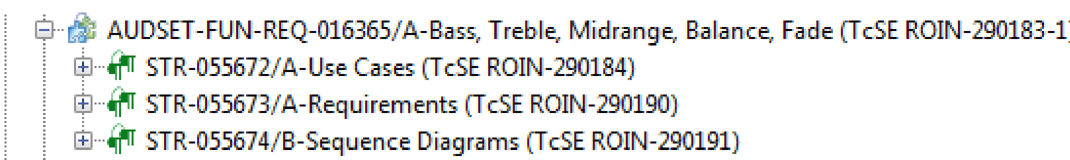


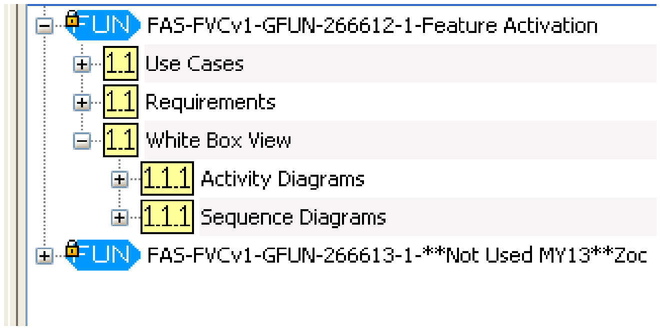
In this example, *MyFeature* includes a server and two client classes. The two clients interact with each other and MyServer interacts with MyClient1, but not MyClient2. Additionally, a user of this feature can interact with the system through two vehicle setting functions, which have an interface to MyServer and MyClient2.

1.3.3 Functions

Functions break down a feature into specific specialized areas to support a feature. Functions are typically organized into Use Cases, Requirements and Sequence Diagrams / White Box Views (White Box Views when have both activity diagrams and sequence diagrams).

Example Function Set-ups:





1.3.3.1 Use-Cases

Use-cases describe a specific operational aspect of a feature function. Each use-case is written at a high-level and independent of system architecture. Use-cases include the following elements:

- **Actors**: Roles/Entities that interact with the feature. Examples are: Vehicle Occupant, System, User…
- **Pre-conditions**: Existing conditions that must be satisfied for a use-case scenario to occur.
- **Scenario Description**: Detailed definition of situations where a use-case occurs. Each use-case will have at least one scenario for normal conditions, but may also include additional scenarios for error situations.
- **Post-conditions**: Resulting conditions after a use-case scenario execution.
- **List of Execption Use Cases**: An Exception Use Case describes a non-happy (rainy-day) scenario. It includes anticipated Error or unforeseen conditions that need to be handled for troubleshooting purposes. Exception use case details are captured in a separate use case.
- **Interfaces**: Relevant interfaces in feature design. Examples are: G-HMI (Graphic HMI), V-HMI (Voice HMI), SWC (Steering Wheel Control), WIFI Interface, Vehicle System Interface.
- **Notes (optional)**: sometimes a note row is on the use case and is used to provide additional information

Actors	
Pre-conditions	
Scenario Description	
Post-conditions	
List of Exception Use Cases	
Interfaces	
Notes	

1.3.3.2 Requirements

Whereas the Use Cases typically give a high level overall description of the Function, the Requirements specify the details of what is needed to implement a feature/function.

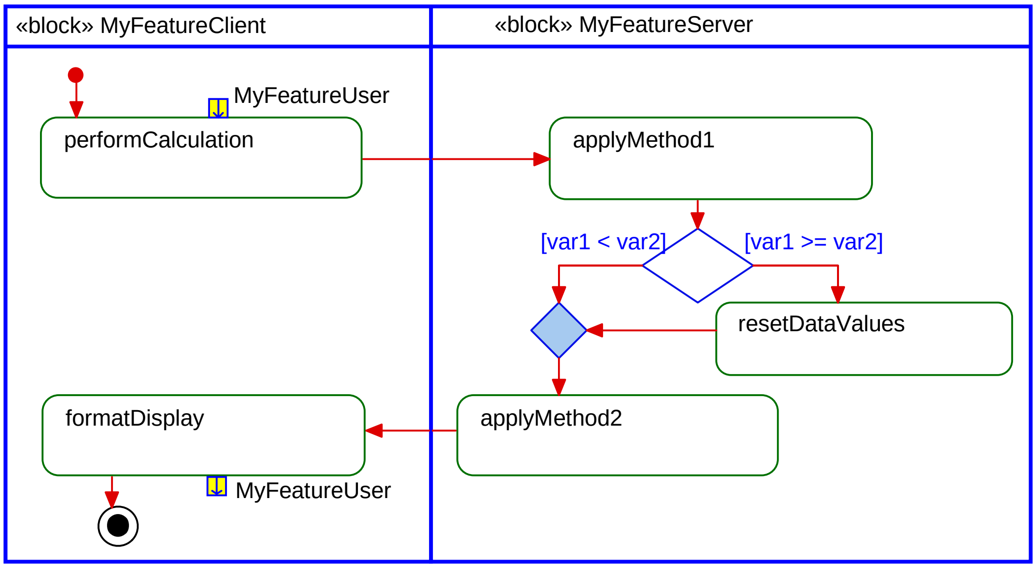
Note: the sequence diagram requirements are typically broken out separately in the Function structure.

1.3.3.3 White-box Views

White-box views provide a partial graphical representation of the system. Two white-box views are included in this document: *Activity Diagrams* and *Sequence Diagrams*.

1.3.3.4 Activity Diagrams

Activity diagrams provide a graphical representation of the overall functional workflow (storyboard) of a function/operation. The diagram style is similar to a flow chart, but also shows class boundaries to reveal system architecture. These diagrams typically capture one or more use-cases. An example activity diagram is shown below.

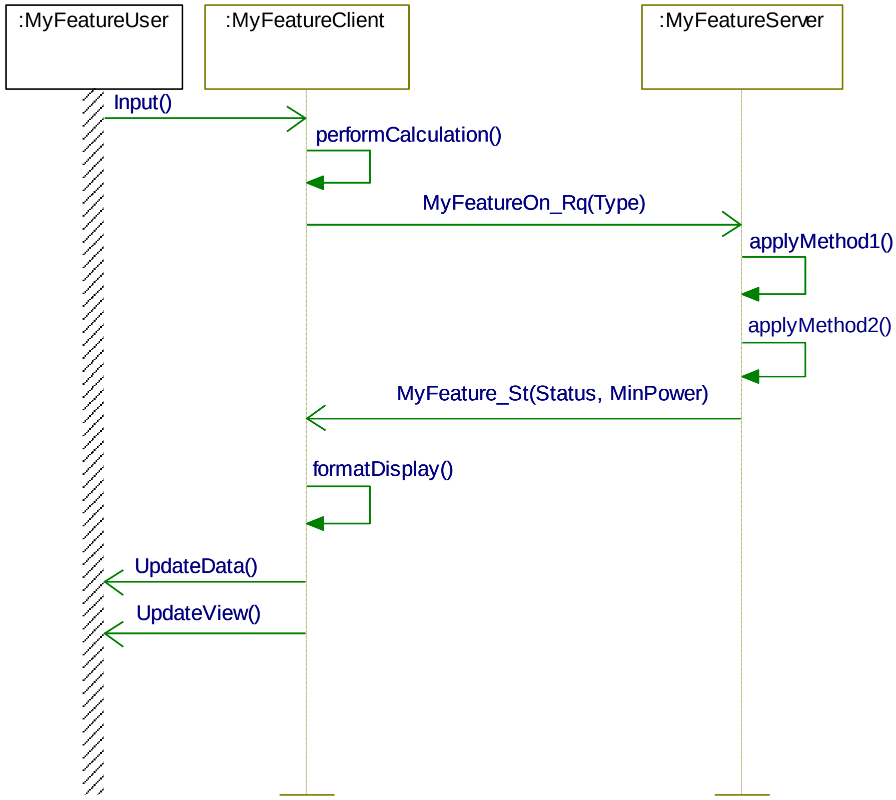


In this example, a user (MyFeatureUser) provides an input to the MyFeatureClient class, which then completes *performCalculation*. MyFeatureServer then uses that result to apply methods 1 and 2 to the data. Note that a data reset might occur between application of method 1 and 2, depending on the values of var1 and var2. After *applyMethod2* is complete, the result is used by the MyFeatureClient, which formats the data for display and communicates it to the user.

Activity diagrams are useful to understand the overall workflow and how it is partitioned between classes. However, they do not provide detailed interface/signal information.

1.3.3.5 Sequence Diagrams

Sequence diagrams provide a graphical representation of interface/signal communication between classes. The diagrams are structured with vertical lines that denote classes and horizontal arrows that show the signals/data exchanged between them. An example sequence diagram is shown below.



In this example, a user (MyFeatureUser) first provides an input to the MyFeatureClient class. This class completes an internal *performCalculation* and sends logical signal *MyFeatureOn_Rq()* to the MyFeatureServer. The server applies two internal methods and replies to the client via logical signal *MyFeature_St()*. The client then formats the data for display and provides information to the user via *UpdateData()* and *UpdateView()*.

If your FeatureClient module displays HMI information then when UpdateView / UpdateData is shown in the sequence diagrams reference the HMI specifications for what to display.

1.3.3.5.1 Internal / External Signals

Sequence diagrams show both internal/external logical signals for a class. Logical signals that originate and terminate on the same class are internal actions. These are typically, but not always closed-tip arrows. Signals that originate and terminate in different classes show communication between them. These are typically, but not always open-tip arrows. In the example above, the communication between client/server is limited to *MyFeatureOn_Rq()* (request to server) and *MyFeature_St()* (reply/status from server).

1.3.3.5.2 Parameters

Logical signals may include parameters/methods. In the example above, logical signal *MyFeatureOn_Rq()* has one parameter/method (*Type*) and *MyFeature_St()* has two parameters/methods (*Status*, *MinPower*).

1.3.3.5.3 Naming Convention for External Signals

External signals are typically named with a *Rq, St, or Rsp* appended to the end of the signal name, denoting a request, status, or response signal, respectively. These are shown in the format *SignalName.Rq* or *SignalName_Rq*. In the above example, the client sends a signal request to the server, which replies with a status signal.

1.3.3.5.4 User Classes and Signals

Most sequence diagrams will include a *user class.* This is not an actual feature class, but reveals how a user/actor can interface with the system. A user class is denoted with a different vertical line style from the other system classes. MyFeatureUser represents a user class in the example above.

Additionally, the following generic signals are used to communicate with user classes:

- *Input()*: Represents an input action from a user.
- *UpdateData() / UpdateView()*: Represents a tangible output to the user.

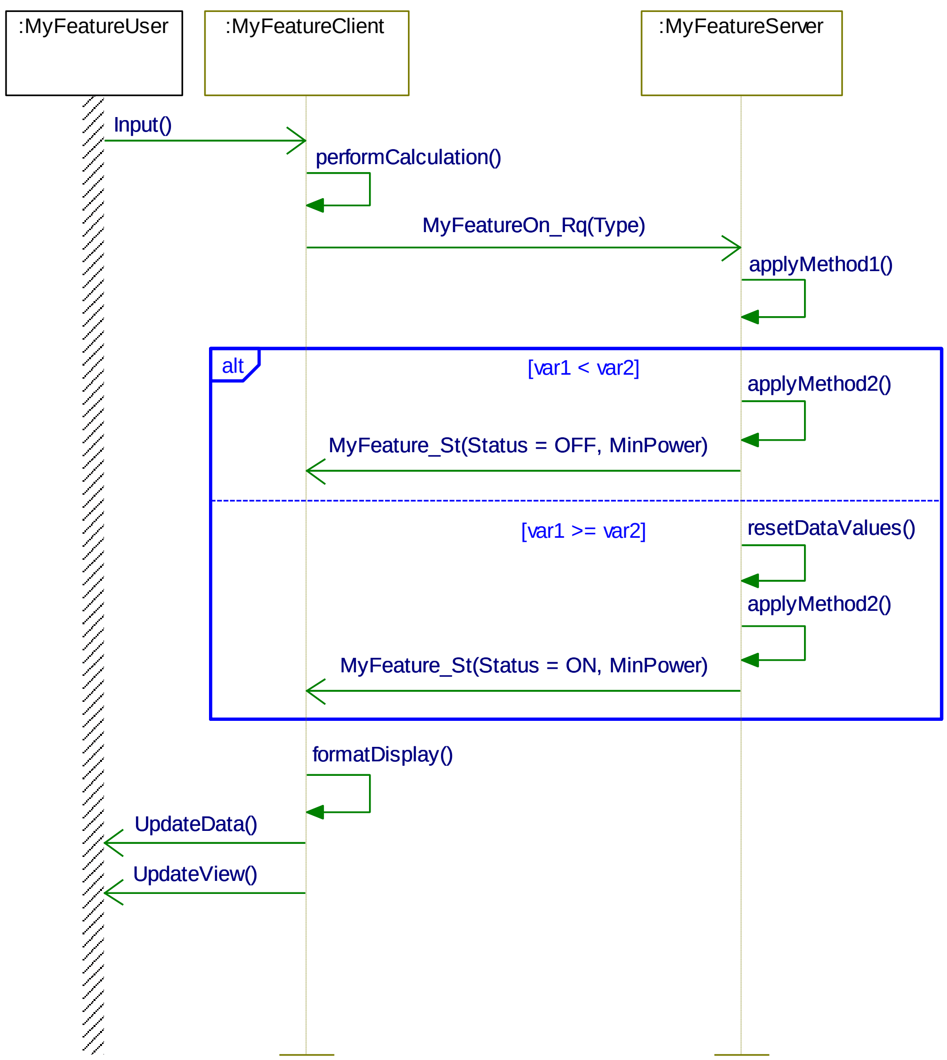
These signals do not map to actual physical/implementation signals.

1.3.3.5.5 Conditional Logic in Sequence Diagrams

Sequence diagrams may contain condition boxes, which show multiple signal paths depending on a particular scenario. The most common condition boxes are:

- Alternative Path (ALT): Used to denote multiple alternative signal paths.
- Optional Path (OPT): Used to denote optional paths.
- Loop (LOOP): Used to indicate a logical loop.
- Break (BREAK): Used to provide a secondary exit path in the diagram.

An example sequence diagram that uses alternative paths (ALT) is shown below. Here, the alternative paths are based on the values of var1 and var2. When var1 >= var2, *resetDataValues()* will be applied and the status will be set as ON instead of OFF.



1.3.4 Linking of Elements

At the beginning of each specification section may be a list of linked elements. These provides additional context to how these descriptive elements together provide a complete specification.

1.4 Preparing for Implementation (Mapping client/servers, requirements and logical signals to modules and network signals)

The main purpose of the "Preparing for implementation" section is how to map the generic method / logical signal, requirements and class descriptions in the SPSS to individual modules and network signals.

1.4.1 Mapping requirements and class descriptions to physical components/modules

Individual modules shall use the implementation guide to map what requirements, classes, feature, functions etc apply to their particular module for an individual feature.

An example is shown below. It shows what requirements apply or don't apply for a particular module. For details on how to use the implementation guide see the legend tab

Part Name: Accessory Protocol Interface Module
Supplier: ______________________

Please see the Legend tab for details on implementation guide layout
SPSS Version: 1.2

Implementation Guide Report

Feature/Function/Requirment/Use Case	Comments	B/C1MCA-E	Programs Implement Option Y/N	B/C1MCA-N	Programs Implement Option Y/N	CGEA1.2E	Programs Implement Option Y/N	CGEA1.2N	Programs Implement Option Y/N	CGEA1.3E	Programs Implement Option Y/N	CGEA1.3N	Programs Implement Option Y/N
AUDSET-FUR-REQ-014938/B-Error State for Convertible Roof Open Occupancy Mode (TcSE ROIN-280696-1)		N		N		N		N		N		N	
AUDSET-SD-REQ-014941/A-Activating Convertible Roof Open Occupancy Mode (TcSE ROIN-280698-1)		N		N		N		N		N		N	
AUDSET-SD-REQ-014942/A-Activating Convertible Roof Closed Occupancy Mode (TcSE ROIN-280706-1)		N		N		N		N		N		N	
AUDSET-FUN-REQ-016390/A-Audio Demonstration Mode (TcSE ROIN-290208-1)		OPT		OPT		OPT		OPT		OPT		OPT	
AUDSET-UC-REQ-016391/B-Audio Demo Mode - Enable (TcSE ROIN-290166-1)		Y		Y		Y		Y		Y		Y	
AUDSET-UC-REQ-016392/B-Audio Demo Mode - Cancel (TcSE ROIN-290180-1)		Y		Y		Y		Y		Y		Y	
AUDSET-SR-REQ-014922/B-Chimes and Prompts during Audio Demonstration (TcSE ROIN-39723-1)		Y		Y		Y		Y		Y		Y	
AUDSET-SR-REQ-014923/B-Audio Demonstration during dual play (TcSE ROIN-39724-1)		N		N		N		N		N		N	
AUDSET-SR-REQ-014924/C-Audio Demo Client activation of an Audio Demo event (TcSE ROIN-39725-1)		Y		Y		Y		Y		Y		Y	
AUDSET-SR-REQ-014925/B-Audio Demo Server response to Audio_Demo_CMND = ON from the Audio Demo Client (TcSE ROIN-39726-1)		N		N		N		N		N		N	
AUDSET-SR-REQ-014926/B-Audio during an Audio Demonstration event (TcSE ROIN-39733-2)		N		N		N		N		N		N	
AUDSET-SR-REQ-014927/C-Audio Demo Server response when an Audio Demonstration event is complete (TcSE ROIN-39734-1)		N		N		N		N		N		N	
AUDSET-SR-REQ-014928/B-Audio Demo Client ending an Audio		Y		Y		Y		Y		Y		Y	

mentation Guide / Legend

Client/Server classes in the SPSS can also be mapped to a module and architecture using the implementation guide.

Example:
For this particular module_X the Class Description objects in the sequence diagrams for Alerts can be mapped to module_X whenever it is marked "Yes" for the Class Description objects "CLD below" in the module_X implementation guide.

Implementation Guide Report

Feature/Function/Requirment/Use Case	Comments	CGEA1.3-BlueOvalArch4.0	Programs Implement Option Y/N
ALERT-CLD-REQ-014717/A-Chime Client (TcSE ROIN-202535-1)		N	
ALERT-CLD-REQ-014718/A-Chime Generator Server (TcSE ROIN-202536-1)		Y	
ALERT-CLD-REQ-014719/A-Chime Audio Source Server (TcSE ROIN-202537-1)	See SPSS whether DSP AMP or AHU is the Chime Audio Source Server	OPT	
ALERT-CLD-REQ-014720/A-Prompt Client/Generator Server (TcSE ROIN-202538-1)		Y	
ALERT-CLD-REQ-014721/A-Prompt Audio Source Server (TcSE ROIN-202539-1)		Y	
ALERT-CLD-REQ-014722/A-Beep Client (TcSE ROIN-202534-1)		Y	
ALERT-CLD-REQ-014723/A-Beep Generator Server (TcSE ROIN-202533-1)		Y	
ALERT-CLD-REQ-014724/A-Beep Audio Source Server (TcSE ROIN-202532-1)		Y	
ALERT-CLD-REQ-014727/A-Turn Signal Visual Indicator Server (TcSE ROIN-263779-1)		N	
ALERT-CLD-REQ-014728/A-Turn Signal Chime Server (TcSE ROIN-263778-1)		Y	
ALERT-CLD-REQ-014729/A-Turn Signal Client (TcSE ROIN-263777-1)		N	

1.4.2 Mapping Interfaces (Signals)

This document is applicable for the Ford Blue Oval Architecture Infotainment System and for all subscribing vehicle lines.

1.4.2.1 Searching the CAN Database File

Implementation/CAN signal names are with their logical method/signal names in the CAN database file. Note that the implementation/CAN signal name is not always the same as the logical signal parameter/method name in the SPSS.

To map a SPSS logical signal/method to its implementation signal counterpart, first search the CAN dB file for the method/signal name as it is defined in the SPSS (Note: if using excel CAN dB file can use the find function - ignore the _Rq, .Rq, _St, .St, _Rsp, .Rsp in the search text). A search match could mean the following:

- If the logical method/signal name is located in the Comments column, then the actual CAN implementation signal is found by scrolling over to the CAN signal name column. Note that a logical method/signal may be listed in multiple locations as it can be sent from multiple modules.

- If the logical method/signal name is located in the CAN signal name column, then the implementation/CAN signal has the same name as its logical method/signal counterpart.

1.4.2.2 Transport Protocol Interfaces

Transport protocol interfaces are denoted with a "CAN-ISO" in the interface requirements and some sequence diagrams (mostly legacy EA style diagrams).

For the newer sequence diagrams there will usually be no “CAN-ISO” marking in front of the Logical signal name in the sequence diagram (ex Rhapsody sequence diagrams won’t have “CAN-ISO”).

A brief description of these Transport Protocol signals will appear in the feature Interface/Method description part of the SPSS for a particular feature. The interface requirements/method description will call out the use of ISO Transport Protocol for a particular signal.

More detailed information on the actual implementation/CAN interface can be found in the *Signal and Channel Catalog* in the *Transport Protocol* chapter of the SPSS.

1.4.2.3 Input Translation Matrix

Button encodings can be found by referencing the *Input Translation Matrix* file.

- For example the button encodings in the network message from the SWCM, CAN ECP/EFP, LIN ICP to the button receivers (ex SYNC Gen 3, CHR…) are defined in the input translation matrix.

Some modules have Button HMI mapping to input translation matrix ID to map a graphic on the button transmitter to a specific input translation matrix ID (ex A65 HMI spec for SYNC Gen 3 and the ECP, EFP, LIN ICP and SWCM).

2 CGEA 1.3C - Physical Architecture

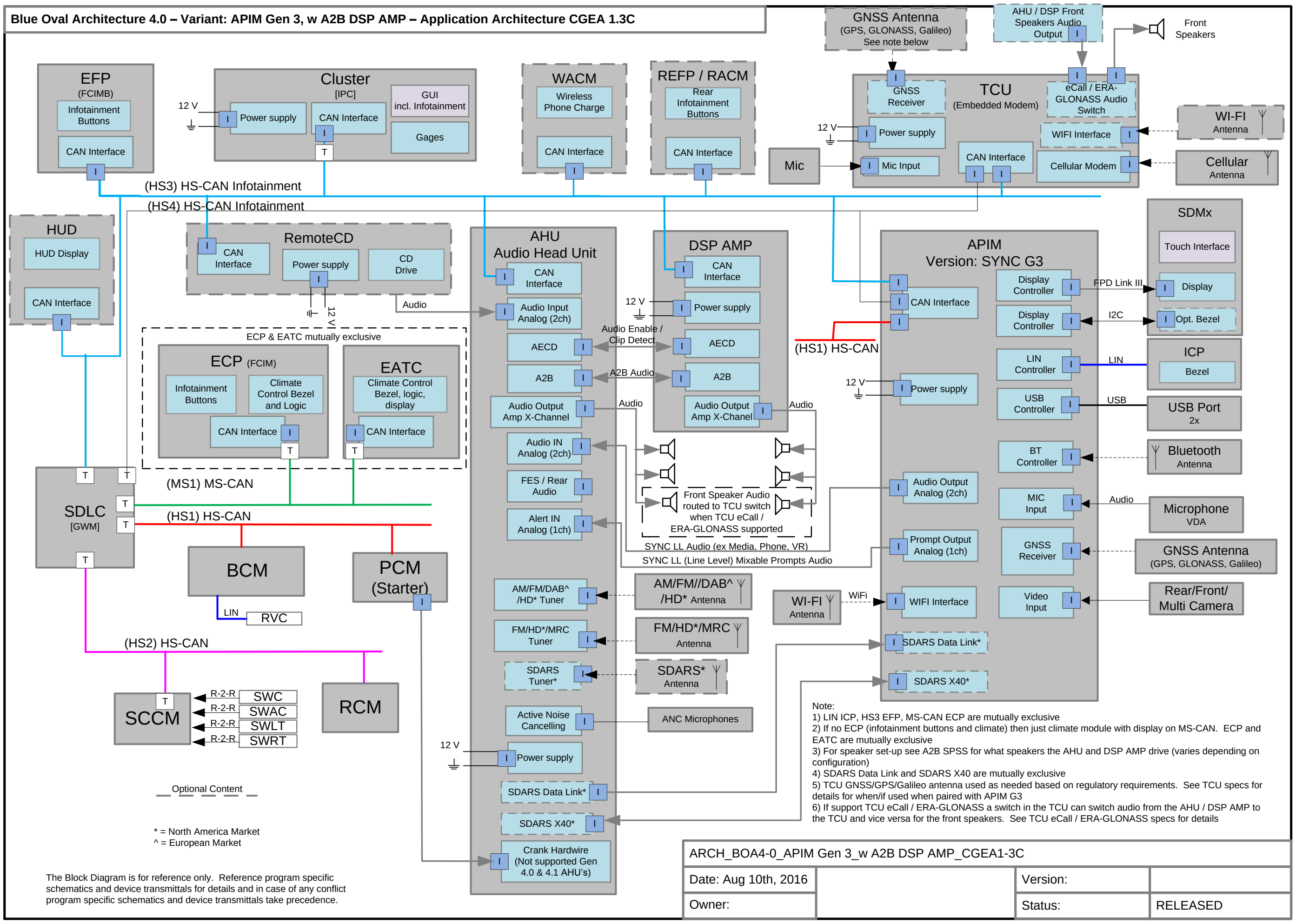


Figure 1 – CGEA 1.3C - APIM Gen 3 with A2B DSP AMP (BOA 4.0)

3 CGEA 1.3 Physical Architecture

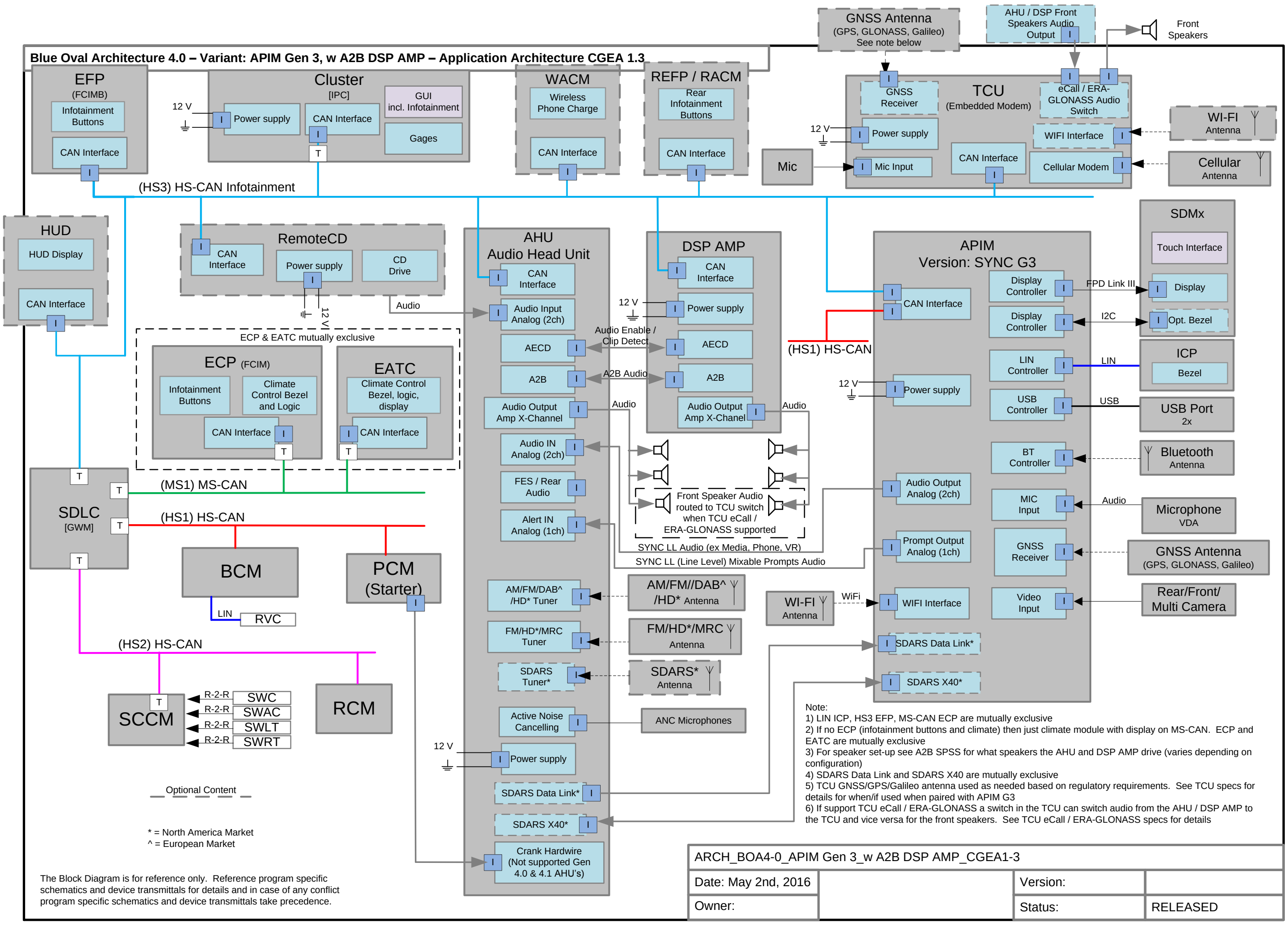


Figure 2 – CGEA 1.3 - APIM Gen 3 with A2B DSP AMP (BOA 4.0)

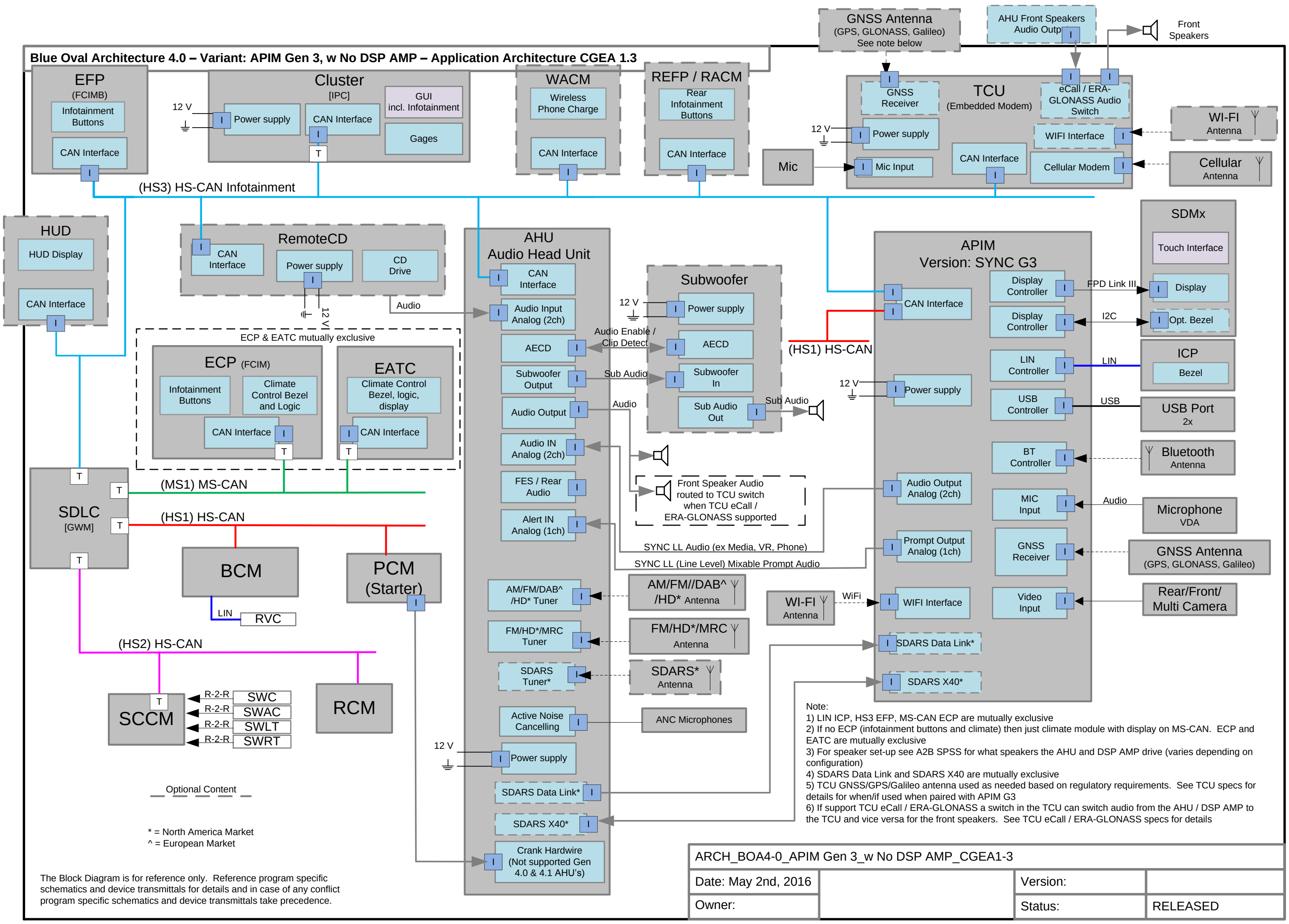


Figure 3 – CGEA 1.3 - APIM Gen 3 with No DSP AMP (BOA 4.0)

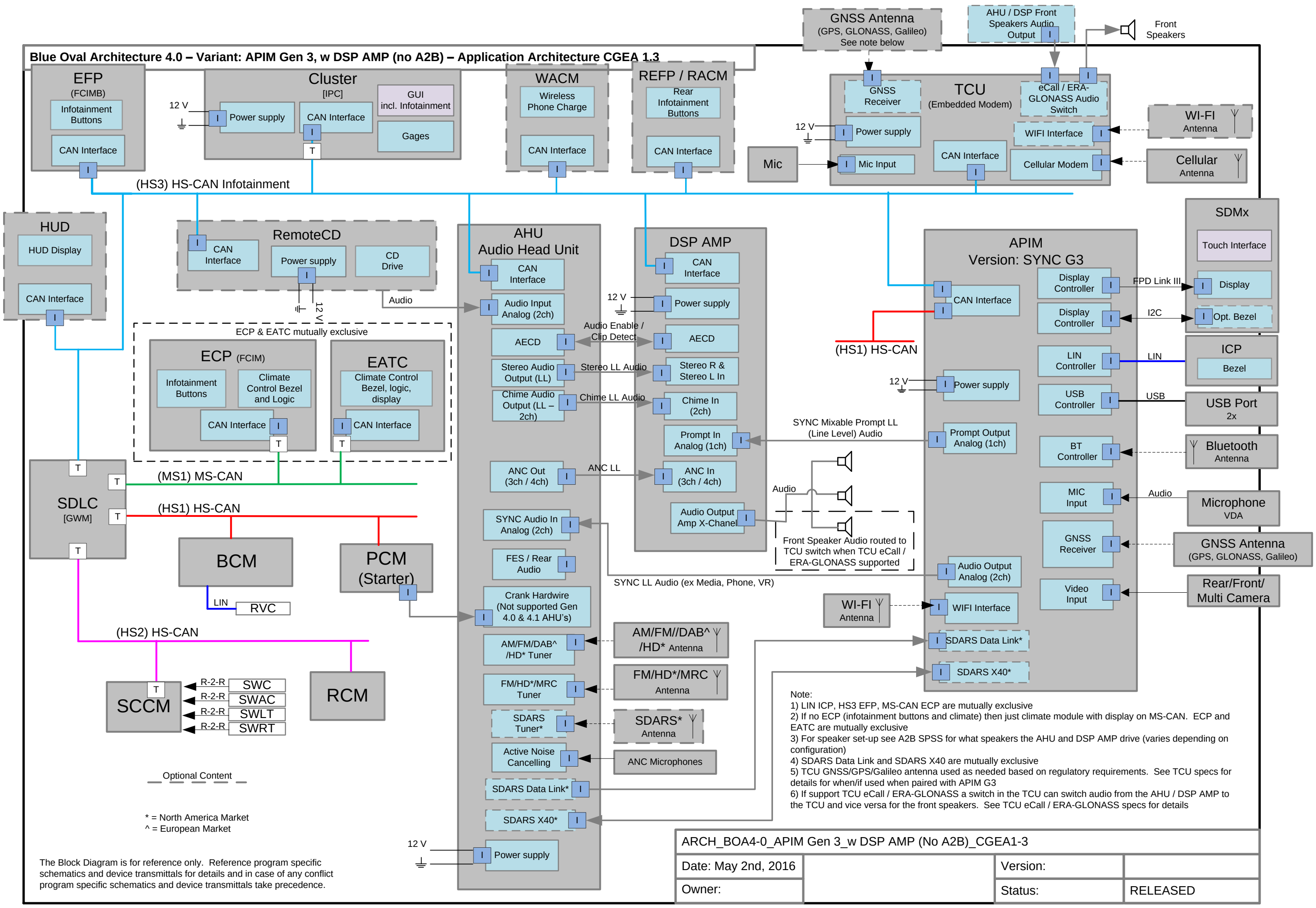
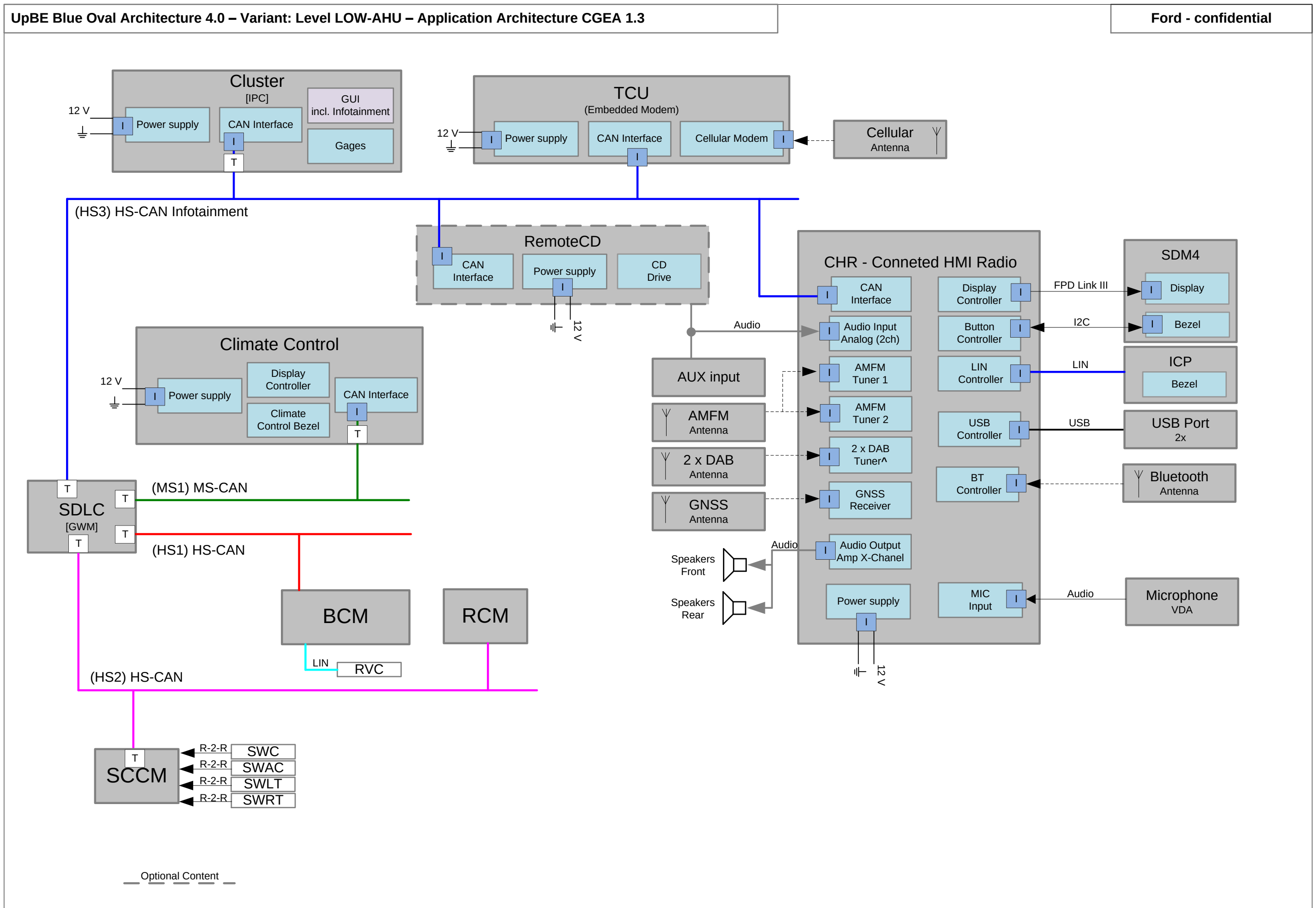


Figure 4 – CGEA 1.3 - APIM Gen 3 with Non A2B DSP AMP (BOA 4.0)



ARCH_BOA4-0_LOW-1_CGEA1-3			
Date:	June 26. 2015	Version:	V4E
Owner:	Peter Patzelt	Status:	RELEASED

Figure 5 – CGEA 1.3 - CHR (Connected HMI Radio) with LIN ICP (BOA 4.0)

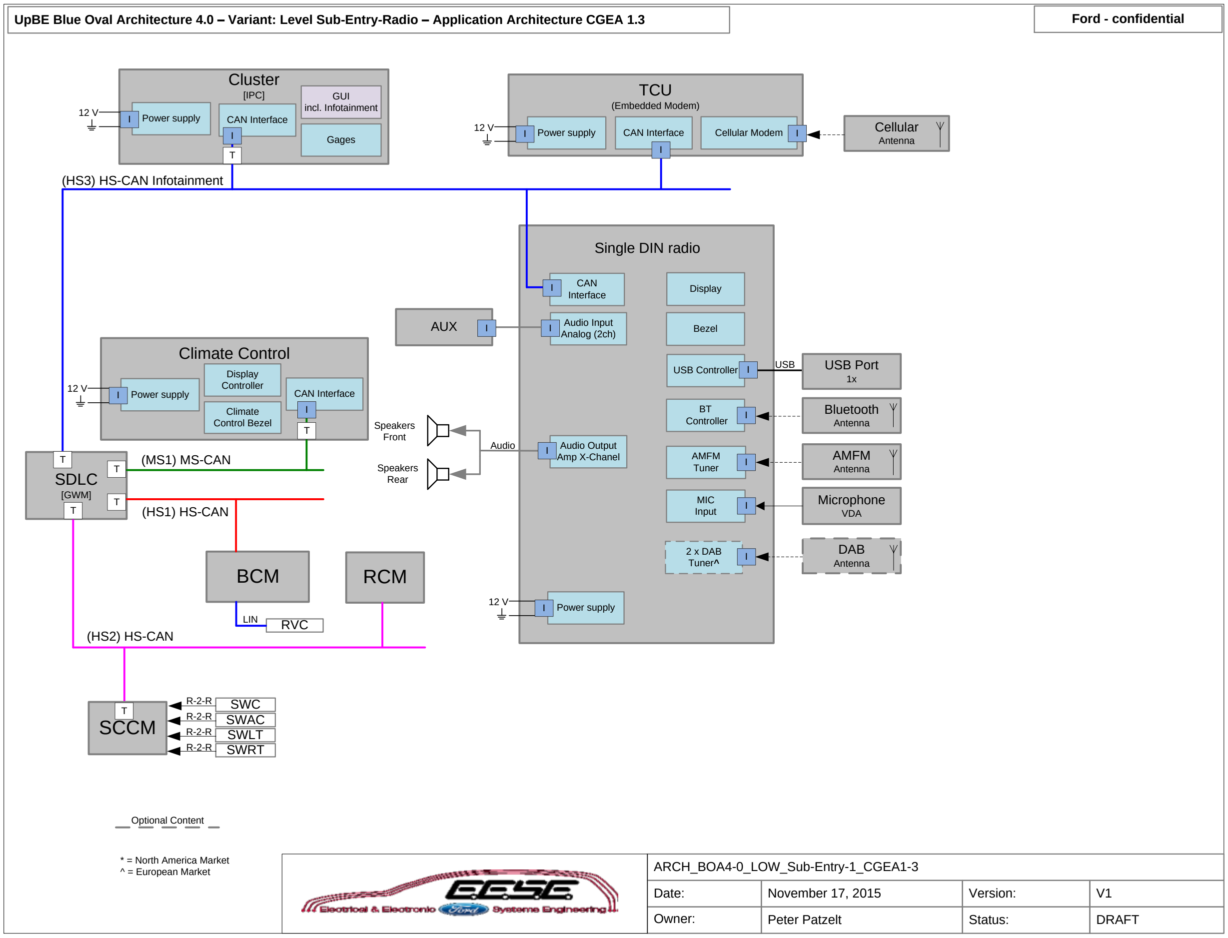


Figure 6 – CGEA 1.3 - Sub-Entry Radio_Standalone AHU (BOA 4.0)

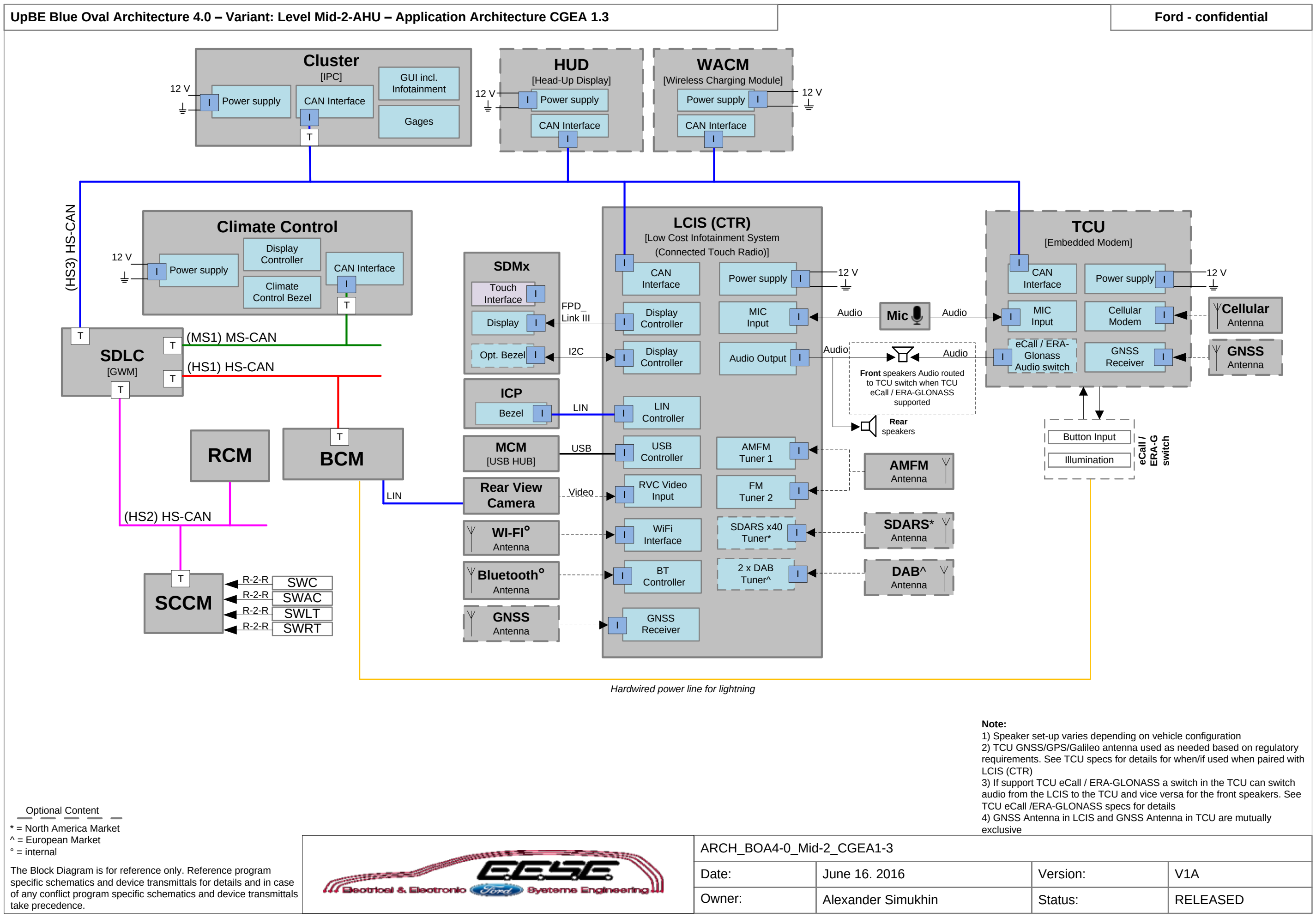


Figure 7 – CTR (Connected Touch Radio) (BOA 4.0)

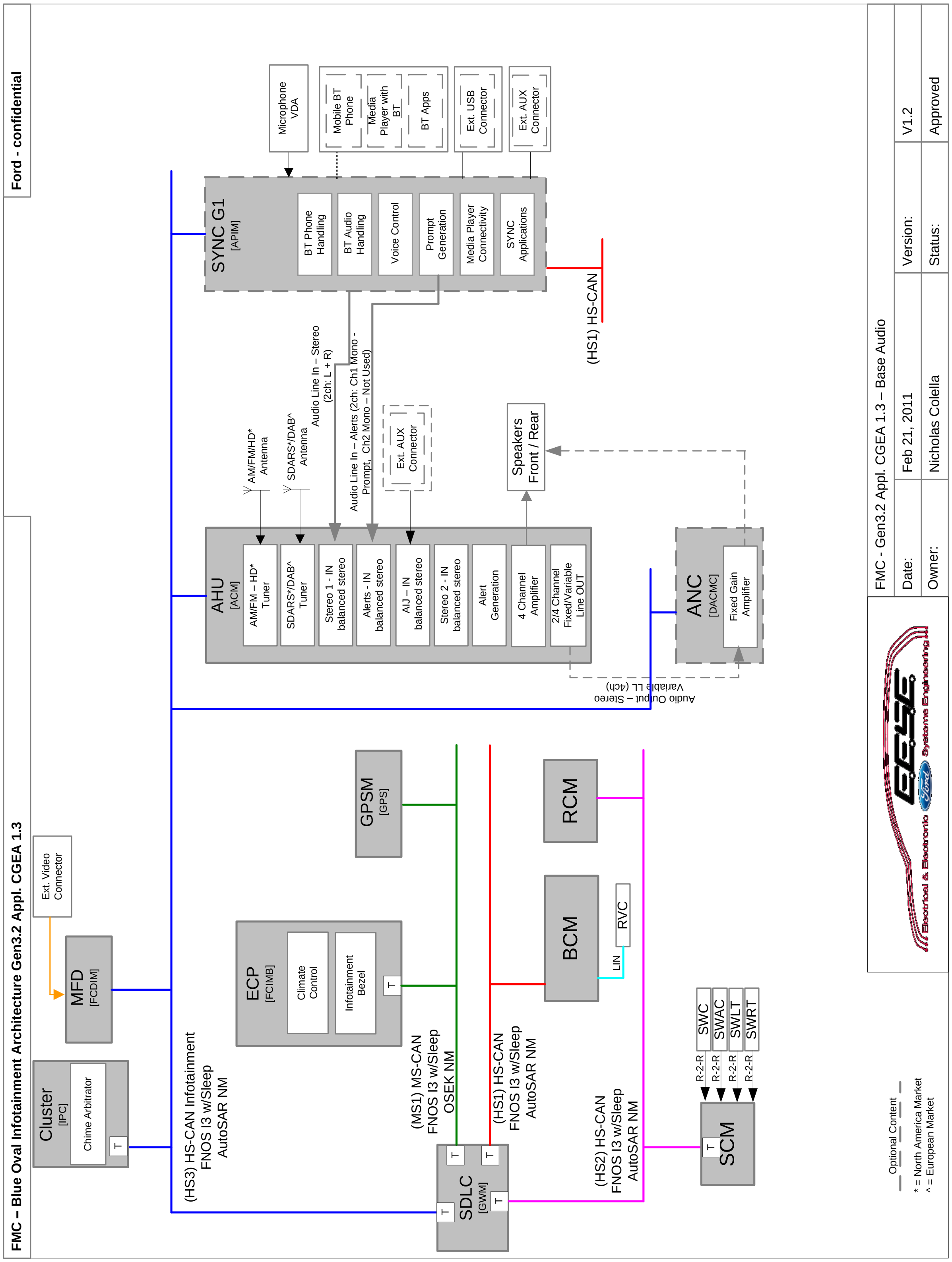


Figure 8 – CGEA 1.3 - MFD with SYNC Gen 1

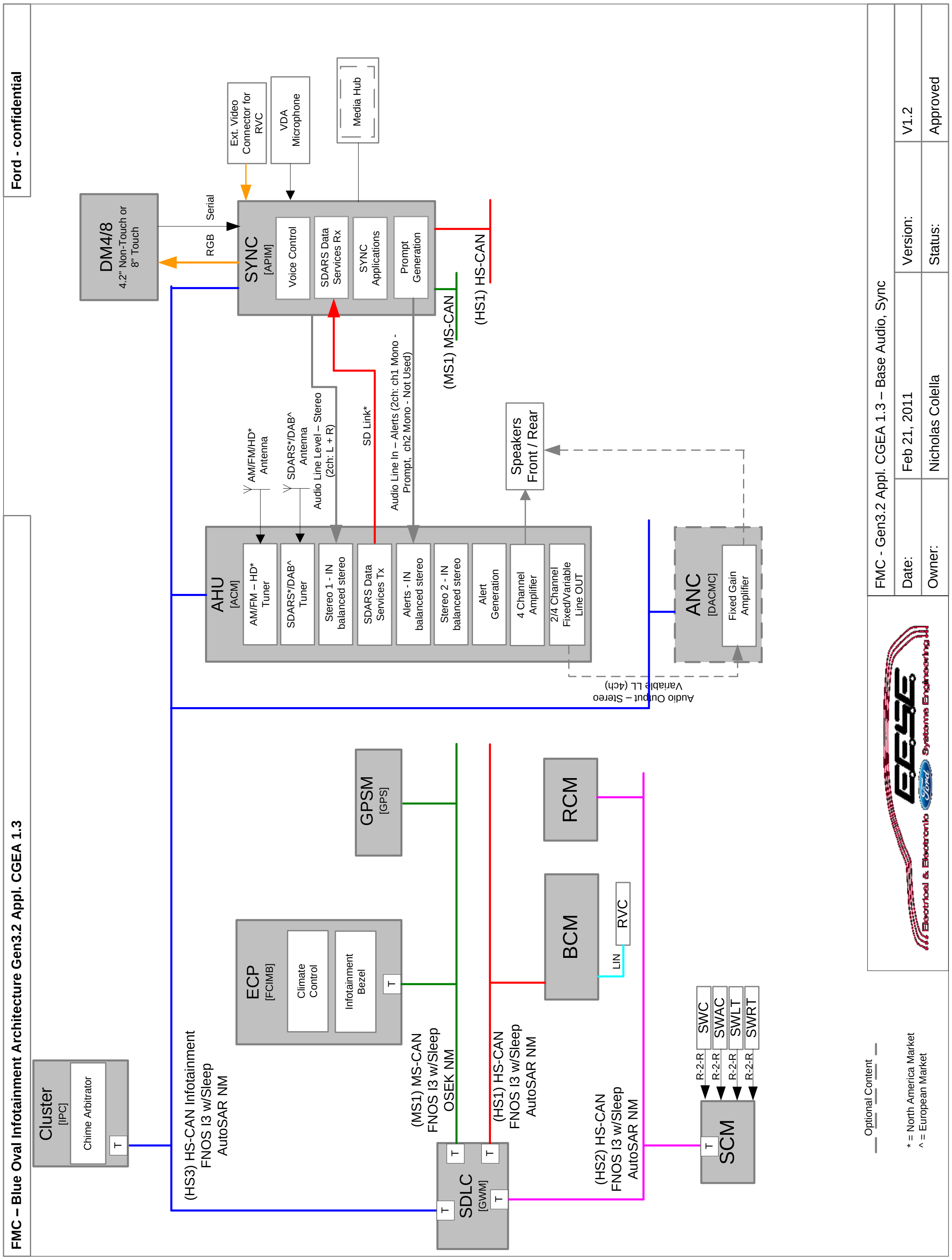


Figure 9 – CGEA 1.3 - APIM Gen 2, No DSP AMP, ECP on MS-CAN

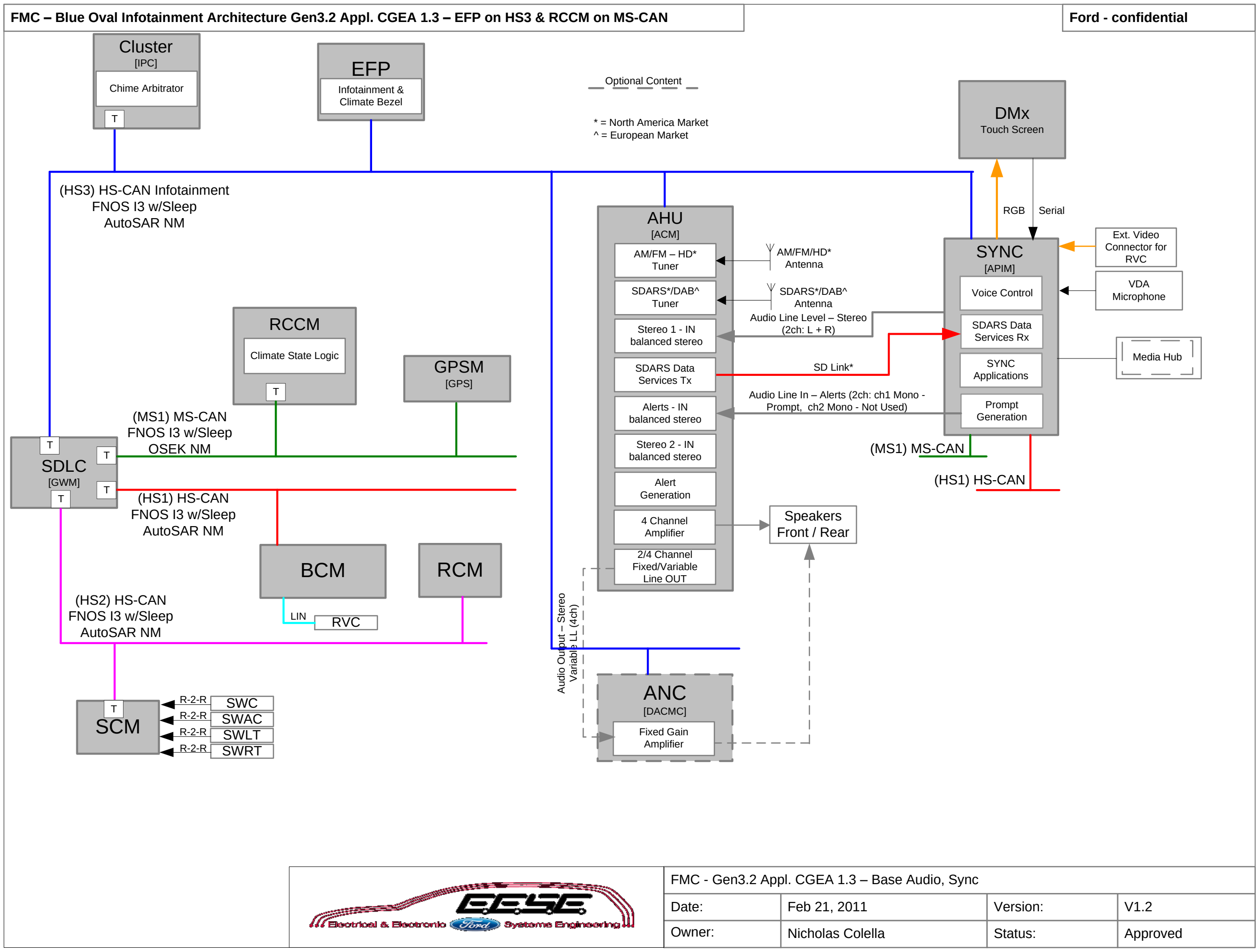


Figure 10 – CGEA 1.3 - APIM Gen 2, No DSP AMP, EFP on HS3

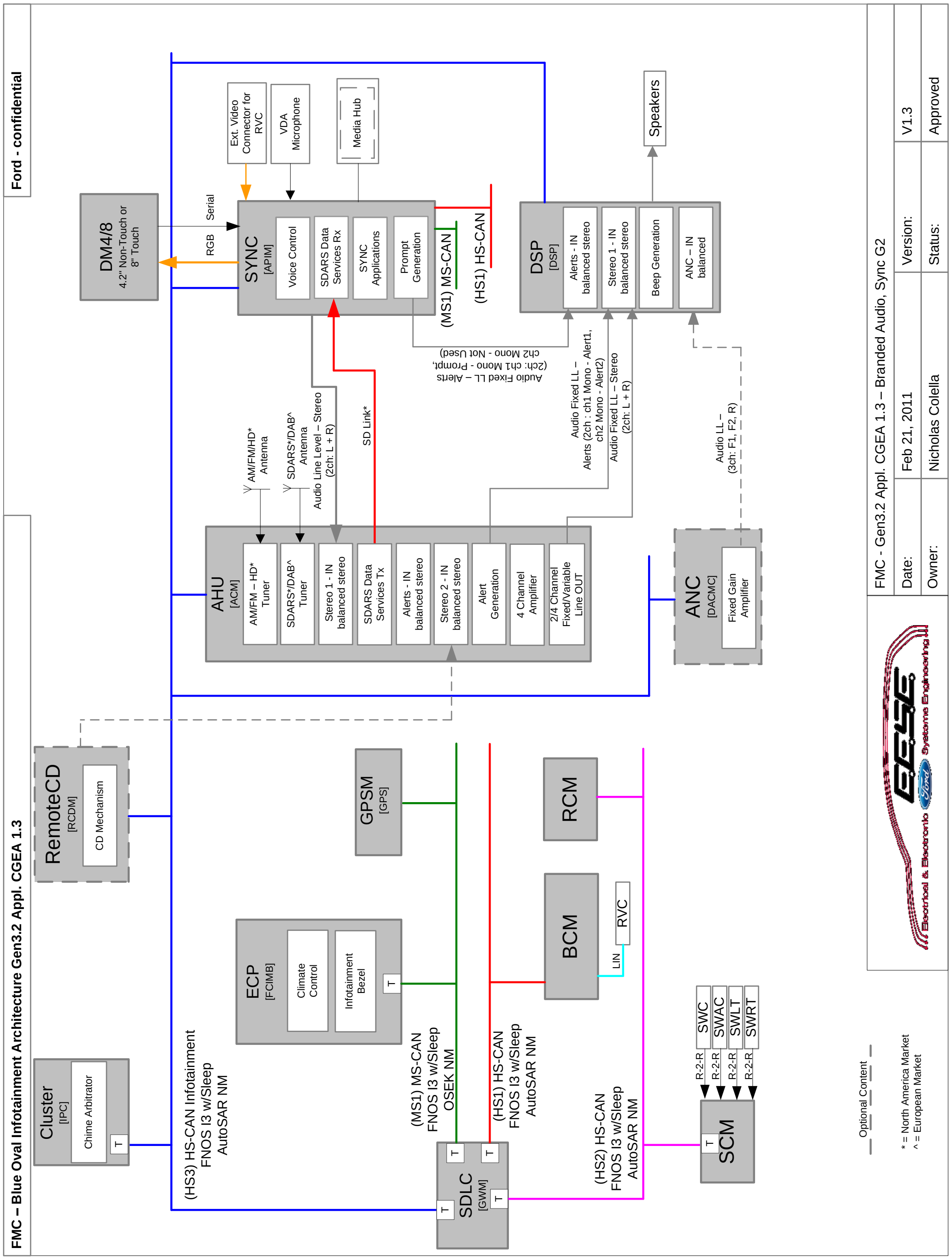


Figure 11 – CGEA 1.3 - APIM Gen 2, w/ DSP AMP (No A2B), AHU without integrated ANC

4 C1MCA Physical Architecture - Sync G2 / Sync G3

Note:
SYNC Gen 2 architecture has a separate GPSM module and SYNC Gen 3 the GPS is integrated

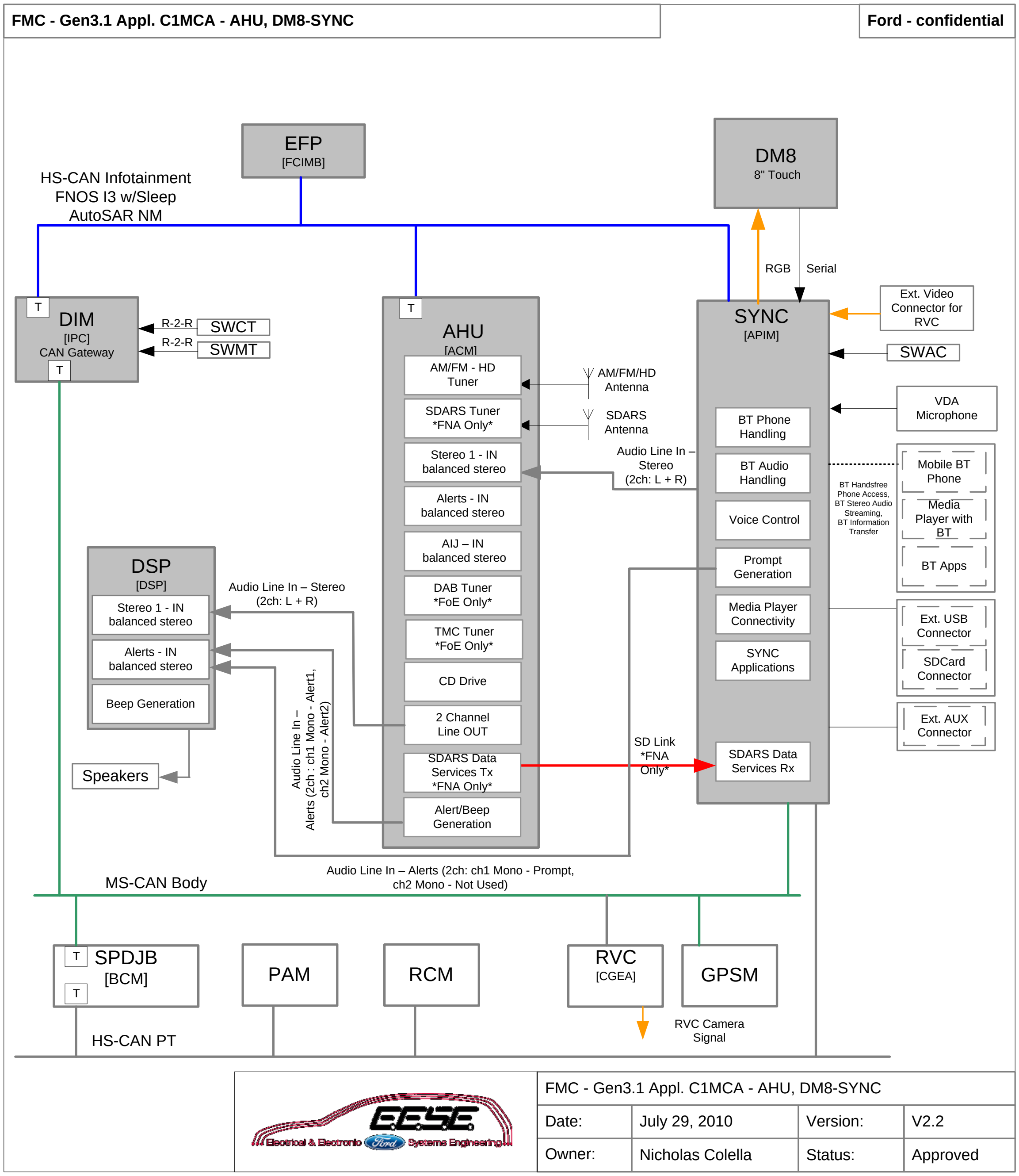


Figure 12 – C1MCA - APIM w DSP

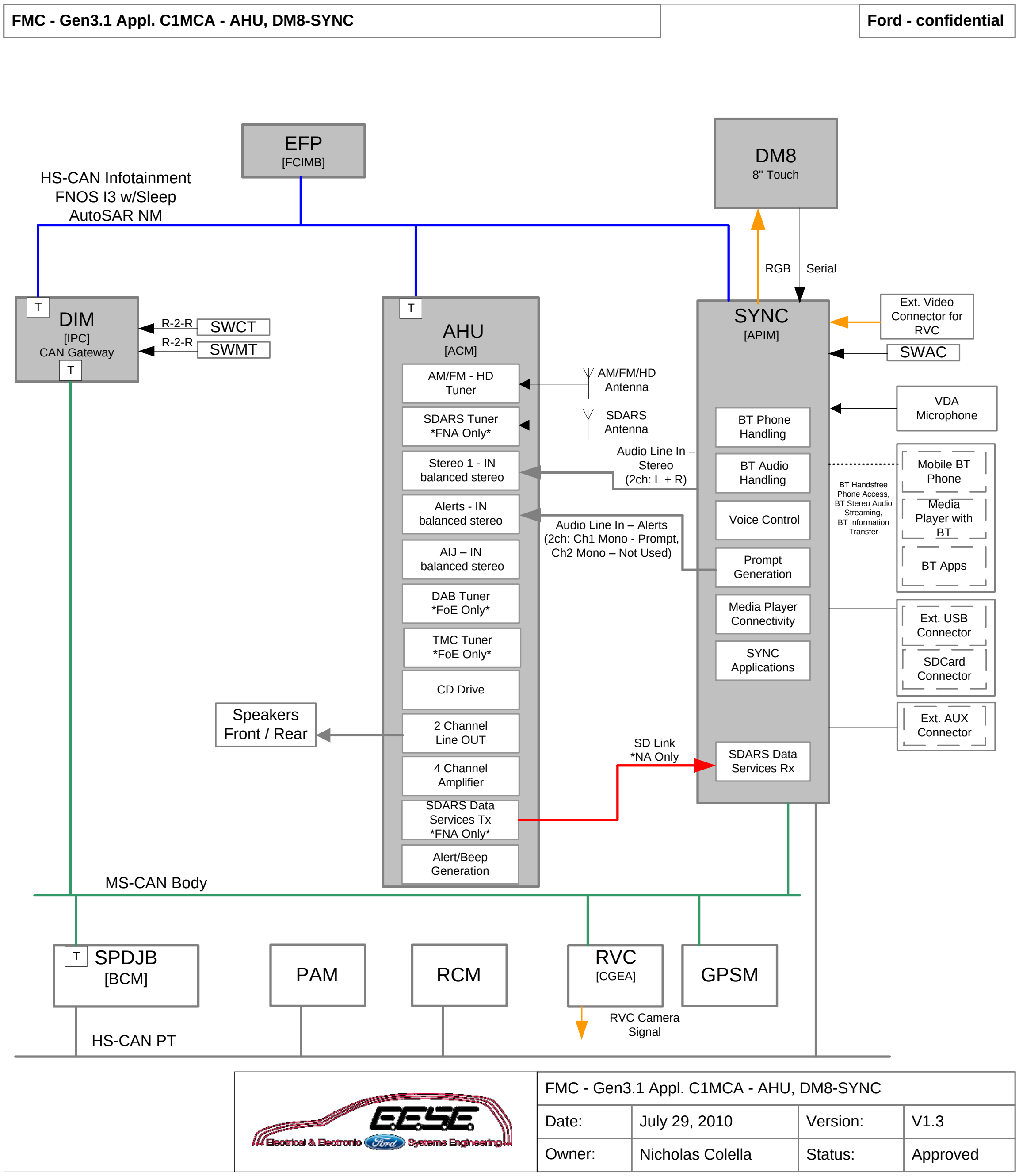


Figure 13 – C1MCA - APIM w/o DSP

5 CGEA 1.2 Physical Architecture

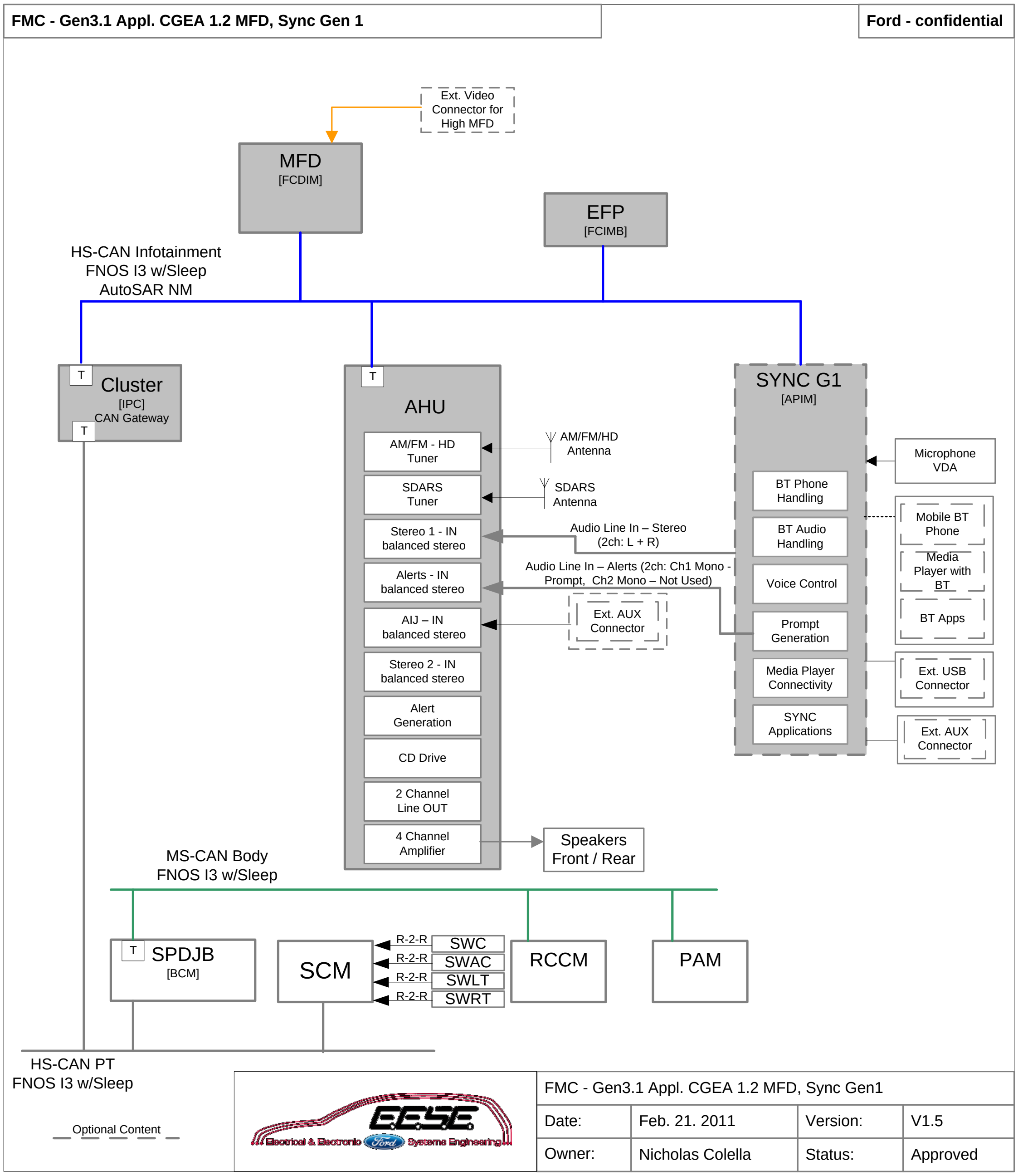


Figure 14 – CGEA 1.2 - MFD

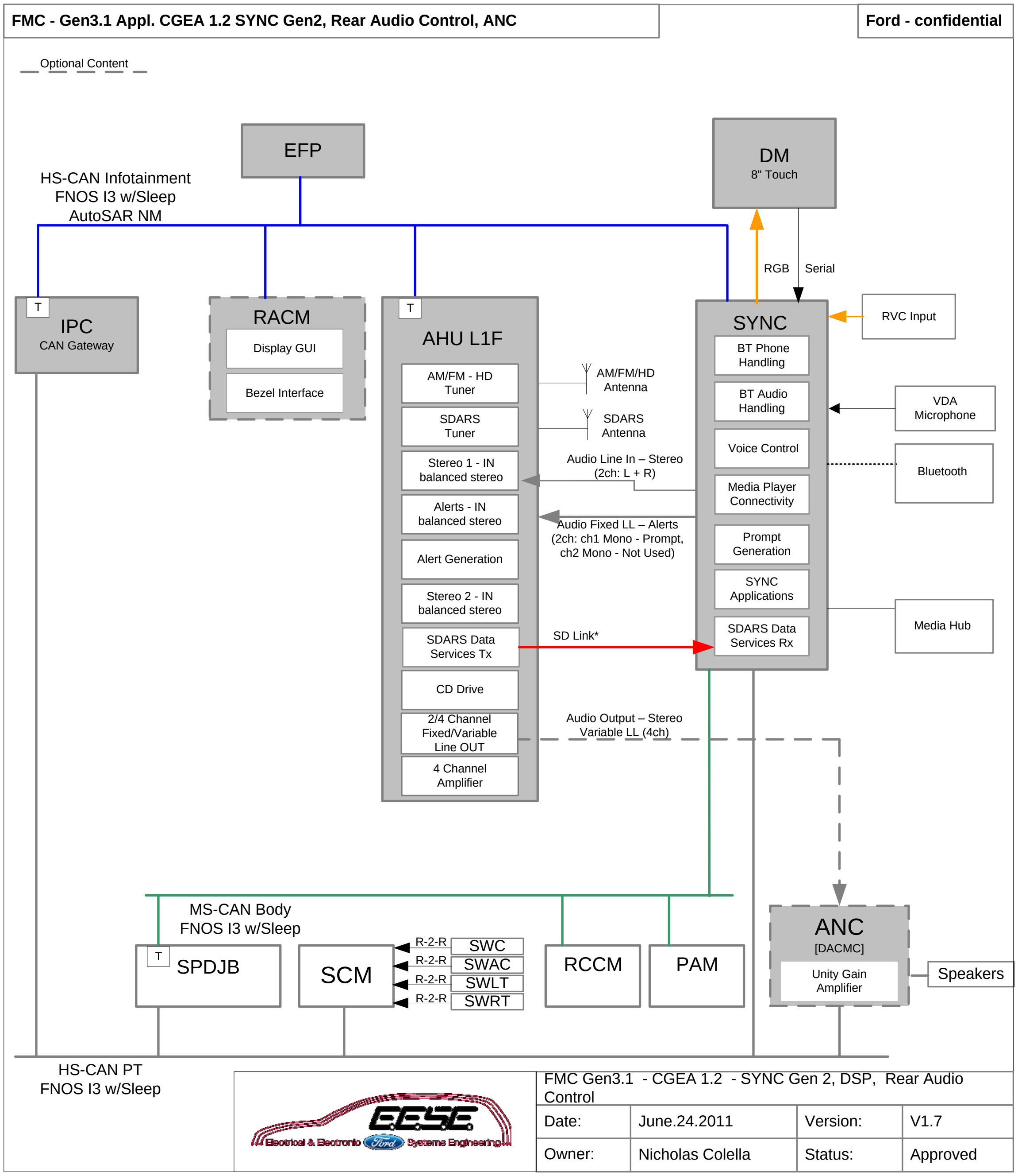


Figure 15 – CGEA 1.2 - APIM w/ ANC, RACM

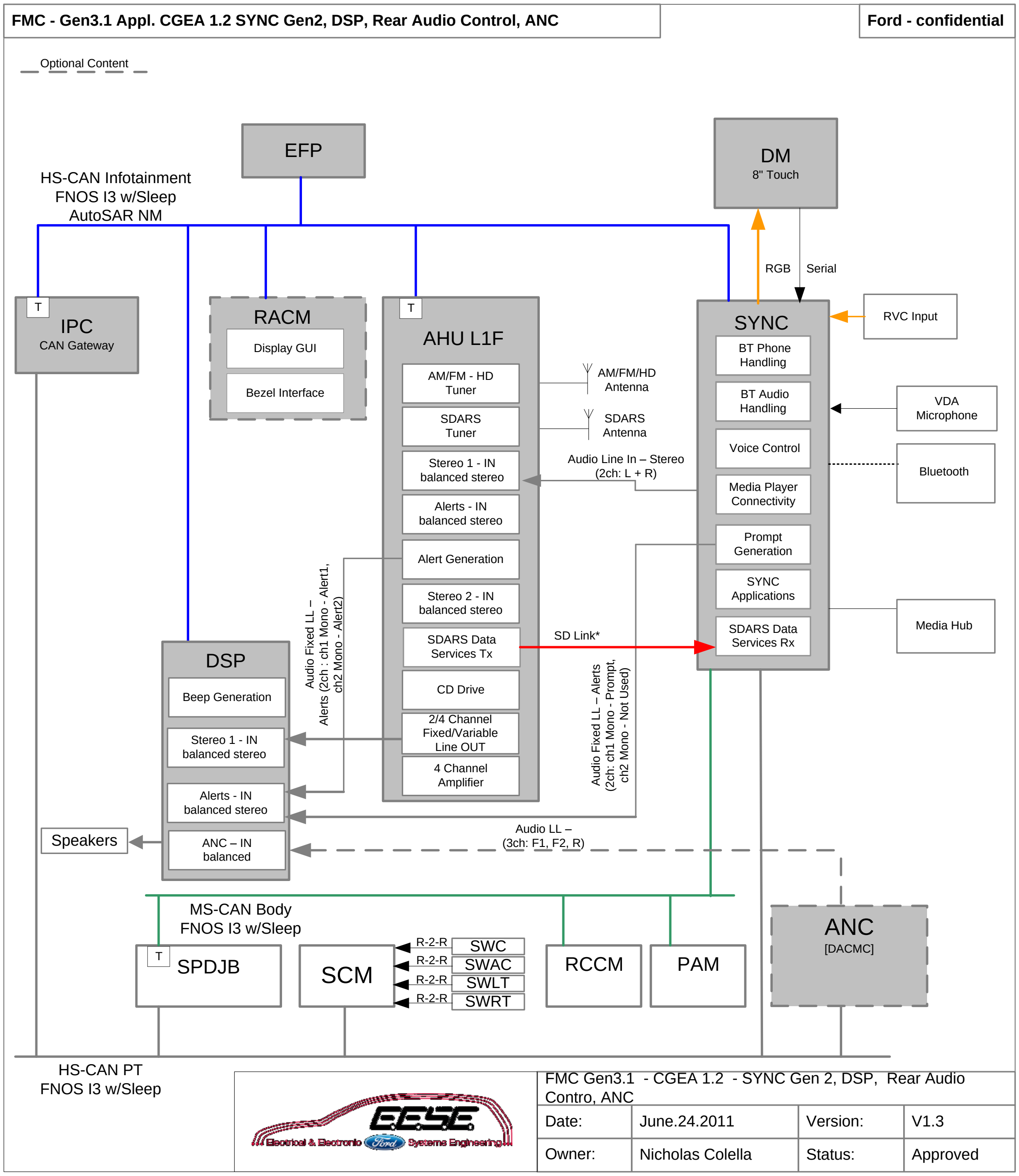


Figure 16 – CGEA 1.2 - APIM w/ DSP,ANC,RACM

6 Appendix A: Definitions / Acronyms

AA – Audible Alert
AAM – Audio Amp Module
ACC – Accessory
ACU – Audio Control Unit
AF - Alternative Frequency - This allows a receiver to re-tune to a different frequency providing the same station when the first signal becomes too weak (e.g. when moving out of range)
AHU – Audio Head Unit = IAM
AM - Amplitude Modulation - AM radio band
AM autoset - AM Autoset preset bank
ANC – Active Noise Cancellation
APA – Asia Pacific & Africa
APIM – Auxiliary Protocol Interface Module (ie SYNC module)
ASM – Audio Source Module
BCM – Body Control Module
BCP – Button Control Panel
BVC – Bluetooth Voice Control (legacy-FoE)
C1MCA – C1 Platform Mid-Cycle Action (current global C-Car Platform e.g. C346, C520 etc)
CAN – Controller Area Network
CC – Climate Control
CD – Compact Disc
CES – Command Execution Status
CGEA – Common Global Electrical Architecture (note: has CGEA electrical body architecture – ex. U38X)
CHR – Connected HMI Radio
CPPM – Cell Phone Passport Module
CT - Central Time
CTR – Connected Touch Radio (LCIS radio)
DAB - Digital Audio Broadcast
DCSM – Dual Climate Seat Module
Del ACC – Delayed Accessory
DLC – Data Link Connector (SAE J1962)
DSP Amp – Digital Signal Processor external amplifier module
Dual EATC – Dual Electronic Air Temperature Control
Dual Play -- The RSE Audio Source is ON (ie an Audio Source is granted in the rear audio stack)
DVD - Digital Video Disc
EBU - European Broadcasting Union - Similar to the National Radio Systems Committee (NRSC)
ECC - Enhanced Country Code
ECU – Electronic Control Unit
ECP – Electronic Control Panel (Infotainment EFP + Climate module in one)
EFP – Electronic Finish Panel (CAN Button Panel)
ES – Engineering Specification
ESE – Engine Sound Enhancement
EON - Enhanced Other Network
FAS – Functional Area Specification
FCDIM – Front Control Display Interface Module (Info-CAN name for the MFD)
FCIM – Front Control Interface Module (CAN dB name for the EFP and ICP)
FES – Family Entertainment System (Rear seat DVD)
FM - Frequency Modulation - FM radio band
FM autoset - FM Autoset preset bank
FNA – Ford North America
FNOS – Ford Network Operating System
FoE – Ford of Europe
FSE – Front Seat Entertainment
GPSM – GPS Module
HD - High Definition Radio - High Definition Radio
HEC – Hybrid Electronic Cluster
HMI – Human Machine Interface
HSWM – Heated Steering Wheel Module
HUD – Heads Up Display
IAM – Integrated Audio Module = AHU
IAHU – Integrated Audio Head Unit
ICP – Instrument Control Panel (LIN Button Panel)
IHS – Infotainment System HS-CAN Network
IKD – Ignition Key Device
Info-CAN – Infotainment System HS-CAN Network

IPC – Instrument Panel Cluster
LIN – Local Interconnect Network
LCIS – Low Cost Infotainment System (CTR touch sense radio)
Manual EATC – Manual Electronic Air Temperature Control
MBP – Multiple Button Press
MC – Message Center
MFD – Multifunction Display
MM Active – Multimedia Active
MM Inactive – Multimedia Inactive
MM System – Multimedia System
MS-CAN – Medium Speed CAN
OHCM – Overhead Console Module
OME – Operation Mode Event
PAM – Park Aid Module
PATS – Passive Anti Thieft System
PHS – Powertrain HS-CAN Network
PI - Program Identification - This is the unique code that identifies the station. Every station receives a specific code with a country prefix
PS - Program Service Name - This is an eight-character static display that represents the call letters or station identification name
PTT – Push To Talk
PTY - Programme Type
PTY31 - Emergency (PTY31) - Alarm Message causes volume to be set to preset level
RBAP – Receiver Button Activation Process
RBDS - Radio Broadcast Data System
RCCM – Remote Climate Control Module
RDS – Radio Data System
RPA – Reverse Park Aid
RSE – Rear Seat Entertainment
RACM – Rear Audio Control Module (same as RSEM / REFP)
REFP – Rear EFP (rear audio controls plus display module – higher end then RSEM)
RSEM – Rear Seat Entertainment Module
RT - Radio Text
SCM – Steering Control Module (same as module SCCM)
SCCM – Steering Column Control Module (Steering wheel controls)
SDARS – Satellite Digital Audio Receiver System
SER – Sub-Entry Radio
Single Play -- The FSE Audio Source shall have control of the entire audio system (ie no audio source granted in rear audio stack)
SPDJB – Smart Power Distribution Junction Box
SPSS – Subsystem Part Specific Specification
SWC – Steering Wheel Control
SWCM – Steering Wheel Control Module
SWAC – Steering Wheel Audio Control
SWLT – Steering Wheel Left Toggle
SWRT – Steering Wheel Right Toggle
TA - Traffic Announcement
TBAP – Transmitter Button Activation Process
TCM – Traction Control Module
TCU – Telematics Control Unit
TMC - Traffic Message Channel
TP - Traffic Program
TTS – Text to Speech
VR – Voice Recognition
VRM – Voice Recognition Module
WACM – Wireless Accessory Charging Module

7 Appendix B: SMS Reference Documents

Reference #	Document Title *Note: talk with Ford D&R for what specifications apply for a particular module. There may be more or less than what is listed below.*
1	Module SPSS Implementation Guides
2	Input Translation Matrix
3	Network CAN dB Message List
4	IDS (Infotainment Diagnostic Specification)
5	Global Power Supply Start/Stop Voltage Curve Specification
6	C1MCA, CGEA 1.2, CGEA 1.3 Illumination Specifications
7	Clock Specification
7	MUX SOW (including all referenced specs)
8	Diagnostics SOW (including all referenced specs)
9	C1MCA, CGEA 1.2, CGEA 1.3 Climate Control System Requirements Specification(s)
10	HMI Specifications
11	EDS SDS (Electrical Distribution System) Ver XXX
12	EESYS SDS (Vehicle Electrical System SDS) Ver XXX
13	Interface Standard Vehicle Speed Signal Specification
14	ECU Software Requirements (Part of the Statement of Work)