MIH RECOMMENDED SMART CABIN API STANDARD VER 1.0

MIH Consortium

2024/7/29

V1.0

Table of Contergergerents

List of Figure

List of Table

MIH Recommended Smart Cabin API Standard

Document Information

- Document Version: MIH Recommended Standard v1.0
- Editors:
  - Foxconn, Tony Chen / tony.kh.chen@foxconn.com
  - Clientron, IJ Wang / ij_wang@clientron.com
  - Clientron, William Chen / william_chen@clientron.com
  - MIH, Elmo Sun / elmo.sun@mih-ev.org
  - Foxconn, Tony Yang / tony.yt.yang@foxconn.com

1.0 Revision History

* Note: please suggest comments in “Reviewing” Mode

Ver.	Date	Edited by	Company	Changes
0.1	2023/2/24	Tony Chen IJ Chen William Chen	MIH Clientron	Initial draft of Smart Cabin API interface specification
0.2	2023/3/16	Tony Chen IJ Chen William Chen	MIH Clientron	Adds Steering Wheel, Seat Control Command
0.3	2023/3/23	Tony Chen IJ Chen William Chen	MIH Clientron	Adds Body and ADAS Control Commands
0.4	2023/4/27	Tony Chen Mickey IJ Chen William Chen	MIH Clientron	Adds HVAC and Door Control Commands
0.5	2023/5/25	Tony Chen Mickey IJ Chen William Chen	MIH Clientron	Adds Light and Mirror Control Commands
0.6	2023-07-06	Tony Chen IJ Chen William Chen	MIH Clientron	Added Audio, Display domain controller commands

0.7	2023-08-16	Tony Chen IJ Chen William Chen	MIH Clientron	Added connectivity, IMS and Window commands
1.0	2024-07-29	Tony Chen Tony Yang Elmo Sun Ted Lien	Foxconn MIH	Added why &what /scope/ benefit/hardware/ software standard and first version Release

TABLE 1-Revision History

2.0 Acronym List

ABS	Anti-lock Braking Systems
ADAS	Advanced Driver Assistance Systems
AEB	Autonomous Emergency Braking
AVH	Auto Vehicle Hold
AVM	Around View Monitoring
BSD	Blind Spot Detection
DMS	Driver Monitor system
EBD	Electronic Brake Force Distribution
ECO	Economical Mode
EPAS	Electric Power Assisted Steering
ESC	Electronic Stability Control
EQ	Equalizer
FCW	Forward Collision Warning
HAL	Hardware Abstraction Layer
HVAC	Heating, ventilation, and air conditioning
HBB	Hydraulic Brake Boost (HBB)
HDC	Hill Descent Control
LDWS	Lane Departure Warning System
LKA	Lane Keep Assist
MOD	Moving Object Detection
PDS	Pedestrian Detection System
SDV	Software-Defined Vehicle
TCS	Traction Control System

TABLE 2-Acronym List

3.0 Abstract (Summary)

This Smart Cabin API standard was developed and confirmed by Smart Cabin working group partners. The purpose of this document is to help developers understand how to interact with MIH software stack by using MIH Smart Cabin API.

4.0 Status of this document

- MIH Recommended Standard v1.0

5.0 Introduction

The purpose of this document is to help developers understand how to interact with MIH software stack by using MIH Smart Cabin API.

By publishing and/or subscribing to the topics mentioned below, developers will be able to create an application that can control various functions of the smart cabin.

5.1 Description of requirement (Why/What for whom)

- End users need easier control of cabin environment, entertainment, and safety via smart cabin.
- OEMs need competitive, flexible integration and deployment of the latest technologies and functionalities.
- Suppliers need standards to reduce development costs and enhance adaptability to different vehicles and markets.
- Mission: MIH standardizes the Smart Cabin to meet the needs of end users, OEMs, and suppliers for long-life vehicles.

5.2 Smart Cabin Scope

This document outlines the standardization requirements for the Smart Cabin of electric vehicles. This technology integrates multiple systems, including convenience, connectivity and cloud services, infotainment, comprehensive safety, and comfort, to provide a more intelligent and interconnected in-car experience.

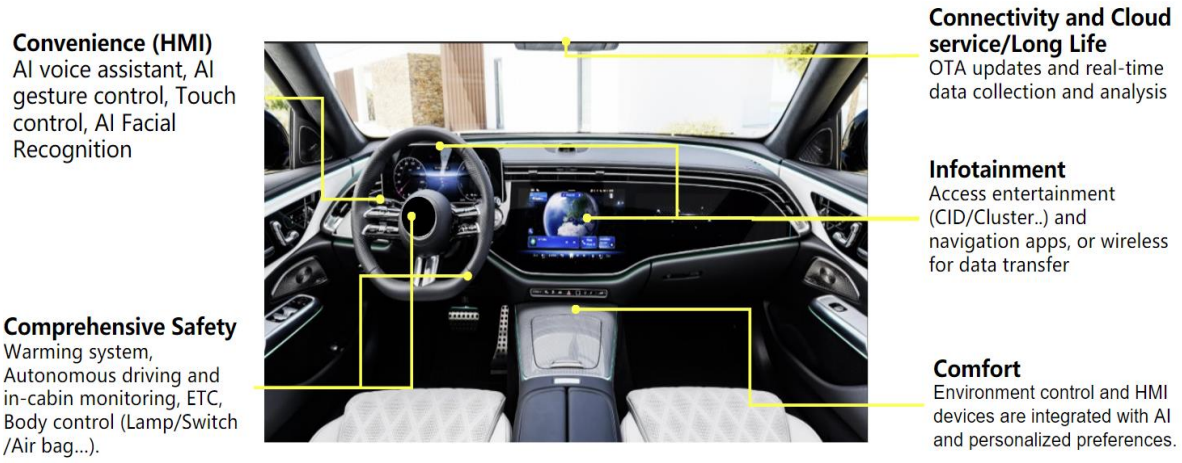


Figure 1 Smart Cabin Scope

5.3 Smart Cabin Benefit

Standardizes the Smart Cabin to meet the needs of End-Users, OEMs, and Suppliers for Long-Life vehicles and benefits of smart cabin are as shown in the table below.

Scope	Traditional Cabin	Smart Cabin	End-User	OEM	Supplier	Remark
Long Life	Limited lifespan, hardware-dependent	Extended lifespan due to software updates	V	-	-	**End-user** through software updates extend the life of the vehicle and provide freshness.
Infotainment System	Basic audio system, physical buttons, lacks advanced media connectivity	Integrate new technology and physical buttons into digital touch systems to reduce supplier costs and transition to different markets and vehicles.	V	V	V	Provides **End-user** with a convenient experience and, through **OEM** integration with other devices, reduces **Suppliers** costs and to fit different vehicles and markets.
Comprehensive Safety	Basic safety features (seat belts, airbags), no advanced driver assistance	Establishing standards for autonomous driving and in-cabin monitoring functions, and applying them to the latest technologies, will enhance driving safety.	V	V	V	**OEM** apply the latest technologies to provide **End-user** with a safe driving experience, through standardization to reduce **Supplier** costs.
Convenience (HMI)	No voice assistant, no gesture control Limited connectivity, often only via radio or	Providing AI voice assistant and gesture control, touch through HMI, and integration with smartphones and other devices. Bring the seamless control experience.	V	-	V	AI technology will provide a seamless experience to **End-user**. Integration and standardization could reduce **Supplier** costs.

	basic infotainment systems					
Comfort	Manual adjustment for seats and climate control	Providing environment control and real-time navigation and traffic updates through HMI/API brings an immersive experience.	V	-	V	Brings an immersive and safe experience to **End-user**, Integration and standardization could reduce **Supplier** costs.
Connectivity and Cloud service	No OTA updates, no real-time data collection and analysis	OTA updates and real-time data collection and analysis will not only provide maintenance reminders but also keep your vehicle up-to-date and extend its lifespan.	V	V	-	Easy to maintain. and update for **End-users** and **OEM.**

TABLE 3- Smart Cabin Benefit

6.0 Smart Cabin SDV architecture

Through SDV (Software-Defined Vehicle) Software and Hardware Decoupling architecture, with standardized interfaces, OEMs and Suppliers achieve seamless control, providing End-Users with an immersive experience.

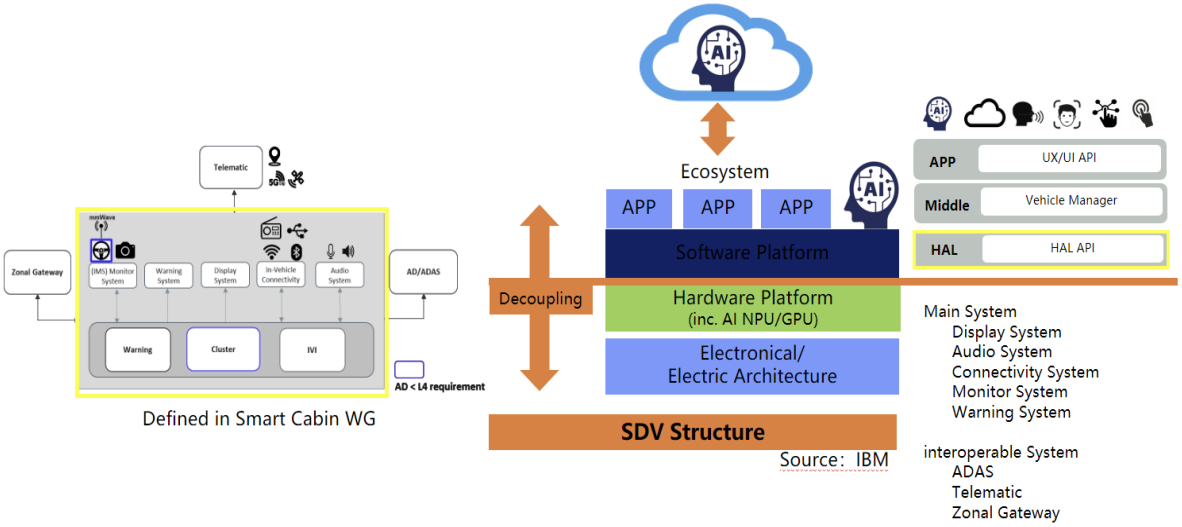


Figure 2 Smart Cabin SDV architecture

6.1 Smart Cabin SDV levels

In addition to architecture, MIH is defining SDV into 5 levels for the convenience and easier understanding of the end users. Basically, the higher the level, the more OTA and capabilities can be configured, controlled and enhanced. In terms of Smart Cabin, the most capabilities can be categorized into 4 levels. We will expand to 5 once more capabilities are available and mature.

Smart Cabin	MIH SDV LEVEL 1	MIH SDV LEVEL 2	MIH SDV LEVEL 3	MIH SDV LEVEL 4	MIH SDV LEVEL 5
Description	Basic SDV Key functions can be digitalized, configured, controlled by software and fixed via OTA. Users feel limited benefit from software.	Partial SDV Most functions can be digitalized, configured, controlled by software and fixed via OTA. Users feel good benefit from software.	Medium SDV + advanced digitalization for personalization like continuous voice recognition, gesture control, multimodal interaction	High SDV + safety functions in addition to digitalization and personalization, like DMS, OMS, IMS	Full SDV to be added
Digitalization					
Audio	V	V	V	V	-
Video	V	V	V	V	-
Phone	V	V	V	V	-
Navigation	V	V	V	V	-
Cluster	V	V	V	V	-
IVI		V	V	V	-
Multitouch		V	V	V	-
HUD		V	V	V	-
Co-driver screen		V	V	V	-
2nd row screen		V	V	V	-
Phone projection		V	V	V	-
HD Map		V	V	V	-
sub-meter GPS		V	V	V	-
Personalization					
wake up word			V	V	-
continuous voice recognition			V	V	-
segmented wake up word			V	V	-
gesture control			V	V	-
Acoustic fingerprint			V	V	-
Multimodal interaction					-
Safety					
DMS				V	-
OMS				V	-

Figure 3 Smart Cabin SDV levels

7.0 Standardization of key components

7.1 Smart Cabin Roadmap

The Smart Cab Roadmap integrates the development and implementation of advanced technologies and systems to enhance passenger comfort, safety and overall experience.

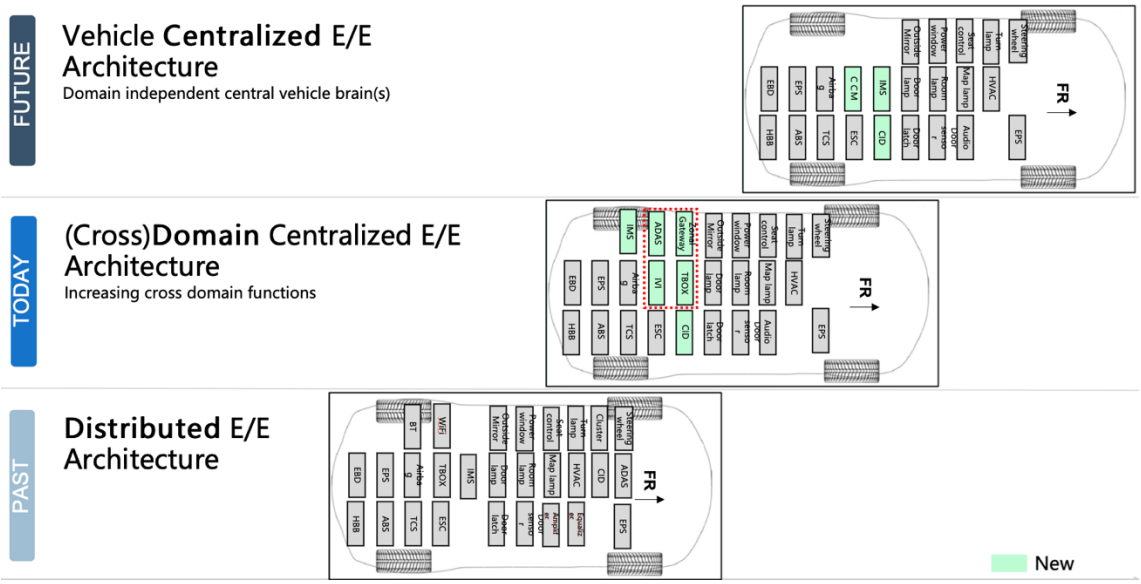
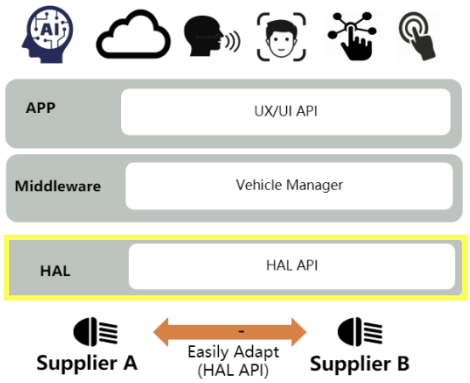


Figure 4 Smart Cabin Roadmap

7.2 Software standard

To ensure vehicle long life and simplify control of the cabin environment, entertainment, and safety via smart cabin, the software standard is needed.

Layer	Item	Description	Layer Structure
APP	UX/UI API	User Experience Interface- ● Tools and interfaces for developers. ● Enhances user experience. ● Creates intuitive, responsive, user-friendly features ● Integrates AI into apps ● Access AI: Machine learning, NLP, image recognition, data analysis ● Use AI: Enhance apps with intelligent features, automate tasks	
Middle ware	Vehicle Manager	Manages and monitors vehicles with tracking, maintenance, data analysis, fleet, and safety features	

HAL	HAL API	Hardware Abstraction Layer Interface ● Easily Adaptive to Different Suppliers: ● Hardware Interface Simplification ● Platform Independence ● Modular Design ● Driver Consistency	

TABLE 4- Software standard

7.3 Hardware standard

The hardware standard based on SDV architecture helps Suppliers reduce development costs and enhance adaptability to different vehicles and markets. For OEMs, it provides competitive and flexible integration and deployment of the latest technologies and functionalities

Scope	Hardware Parts	Distributed	Domain	Centralize
Connectivity and Cloudservice/ Long Life	TBox (External Connectivity)	X	O	◎
	WiFI	X		
	BT	O		
Infotainment	IVI	X	O	
	CID	X	O	O
	Cluster	X	O	O
	Equalizer,	O	O	O
	Amplifer	O	O	O
Comprehensive Safety	ADAS	X	O	◎
	Occupant detector	X	O	O
	DMS	X	O	O
	Meter	O	X	X
	Turn lamp switch	O	O	O
	Break light	O	O	O
	EPS	O	O	O
	Airbag	O	O	O
	ABS	O	O	O
	TCS	O	O	O
	Door sensor, Door latch	O	O	O
Comfort	HVAC	O	O	O
	Map lamp	O	O	O
	Romp lamp	O	O	O
	Door lamp	O	O	O
	Ambient light	O	O	O

Convenience	Seat control module	O	O	O
	Power window switch	O	O	O
	Control switch,	O	O	O
	Hands off detector	O	O	O
	Outside Mirror switch	O	O	O

O：Have ◎：Integrated X：None

TABLE 5- Hardware standard

7.4 Sanity Check

Sanity check whether N7 and Model 3 meet the smart cabin scope and Domain definition based on standard SDV structure.

Scope	Hardware Parts	Toyota Corolla Cross	Lexus NX	Luxgen N7	Telsa Model 3
Connectivity and Cloudservice/Long Life	TBox (External Connectivity)	X	X	O	O
	WiFI	X	X	O	O
	BT	O	O	O	O
Infotainment	CID,	X	O	O	O
	Cluster	X	O	O	X
	Equalizer,	O	O	O	O
	Amplifer	O	O	O	O
Comprehensive Safety	ADAS	L2	L2+	L2	L2+
	Occupant detector	X	X	O	O
	DMS	X	X	X	O
	Turn lamp switch	O	O	O	O
	Break light	O	O	O	O
	EPS	O	O	O	O
	Airbag	O	O	O	O
	ABS	O	O	O	O
	TCS	O	O	O	O
	Door sensor, Door latch	O	O	O	O
Comfort	HVAC	O	O	O	O
	Map lamp	O	O	O	O
	Romp lamp	O	O	O	O
	Door lamp	O	O	O	O
	Ambient light	O	O	O	O
Convenience	Seat control module	O	O	O	O

	Power window switch	O	O	O	O
	Control switch,	O	O	O	O
	Hands off detector	O	O	O	O
	Outside Mirror switch	O	O	O	O

O：Have　X：None

TABLE 6-Sanity Check

8.0 Smart Cabin System architecture

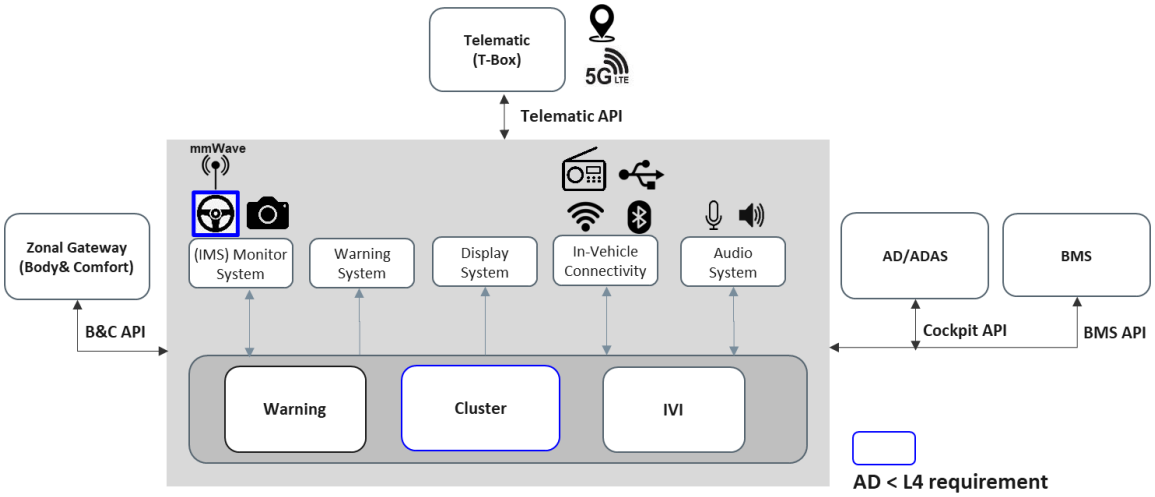


Figure 5 Smart Cabin System Architecture

Smart Cabin system architecture. The Smart Cabin controller includes Cluster, IVI, and Warning units, and the system is connected to Display system, Warning system, In-Vehicle Connectivity system, In-vehicle Monitor system (IMS), Audio System, and connected with ADAS/AD autonomous driving domain controller as autonomous driving, BMS battery management system, Zonal Gateway (Body and Control) display and control.

Some functional systems of the smart cabin involve different levels of autonomous driving. When level 4 and 5 autonomous driving, the driver's control of the vehicle may be ignored. There is no need to consider the driving function.

8.1 API Structure

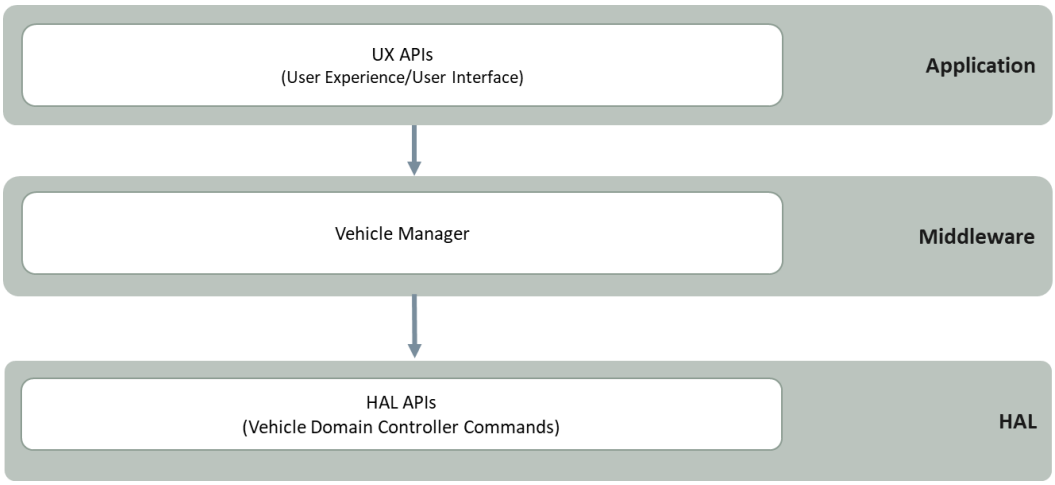


Figure 6 Smart Cabin APIs Structure

UX API : Through the GUI, Voice, Gesture, or manipulation of interface elements, communicate, control, and retrieve information from underlying devices.

HAL API : Abstraction of vehicle interior hardware and vehicle system functionalities enables applications to be developed across different vehicles and hardware platforms, for instance, through protocols like CAN Bus , LIN Bus, or Virtual switch in hypervisor or container structure.

8.1.1 User Experience / User Interface (UX APIs)

Using UX APIs refers to the utilization of application programming interfaces (APIs) to enhance the user experience. This can encompass various contexts and purposes. It can help improve the performance, functionality, and user experience of your application, providing added value to users. Depending on the specific requirements of the application, we can choose different UX APIs to achieve objectives.

8.1.2 Vehicle Domain Control (HAL APIs)

Integrating controls from different domains (such as seats, windows, HVAC systems, doors, etc.) into HAL APIs can further enhance the following aspects:

- Hardware abstraction: HAL APIs provide the hardware abstraction layer to different field devices on the vehicle so that these devices can be accessed and controlled more easily. It reduces direct interaction between developers and hardware, improving development efficiency and maintainability.
- Agnostic: Integrating control from different areas into a HAL API ensures compatibility across other models and manufacturers. This means developers can cross multiple platforms without having to modification.
- Simplify the development process: Integrating control from different domains can significantly simplify the development process. Developers can use the same API

interface to access seats, windows, HVAC systems, doors, and more, rather than learning and managing multiple APIs or interfaces.

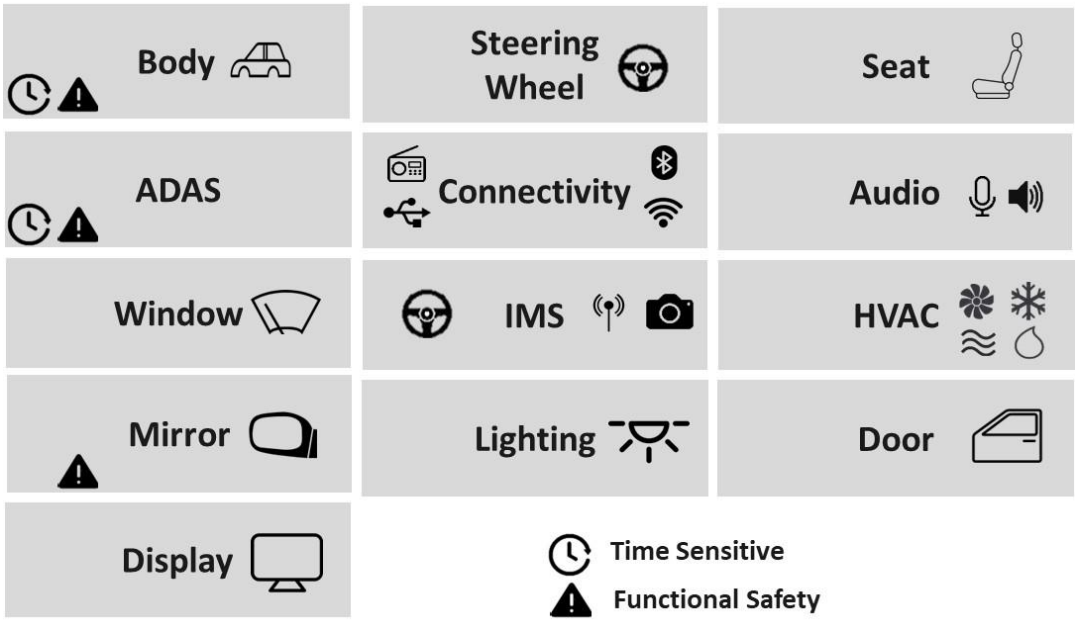


Figure 7 Domain Controller

- Body domain

The body refers to the outer shell of a vehicle, including the front, rear, sides as well as the components within these parts, such as turn signals, and speed meters.

- ADAS domain

ADAS is a vehicle technology designed to improve driving safety. It includes a series of electronic systems and sensors that can help drivers avoid collisions. ADAS technology is becoming increasingly popular in modern cars, helping to improve the driving experience for drivers and reduce the risk of traffic accidents.

- Steering Wheel domain

The steering wheel system integrates a variety of advanced technologies and functions, including integrated controls and behavior monitoring, to provide drivers with a more convenient and intelligent driving experience.

- Seat domain

The purpose of adjusting a vehicle seat is to ensure that the driver or passenger is sitting comfortably and safely while operating or riding in the vehicle. Adjusting the seat allows the person to achieve a comfortable driving or riding position that provides adequate support to their body while also ensuring they have a clear view of the road or surroundings.

- HVAC domain

HVAC control system is designed to maintain a comfortable temperature inside the vehicle in different weather conditions, providing better comfort and driving experience for passengers.

- Door domain

The doors provide passengers in and out of the vehicle and are equipped with many advanced features. To enhance the safety and convenience of the car, making it easier for passengers to enter and exit.

- Display domain

Vehicle display system provides information to the driver and passengers about vehicle performance, settings, and infotainment, and provides access to advanced features such as entertainment, navigation and personalized control interface, etc.

- Audio domain

Vehicle audio systems can be a part of the overall entertainment experience, enhancing in-car comfort and making each driving experience more enjoyable.

- Mirror domain

Mirrors in a car include rear-view mirrors and side-view mirrors that allow the driver to observe the traffic behind. It enables the driver to see other vehicles, pedestrians, and bicycles, facilitating lane changes, overtaking, and parking. The rearview mirror plays a crucial role in driving safety, helping drivers make informed decisions and reducing the risk of accidents.

- Window domain

Car windows are an important component of a vehicle that provides visibility and ventilation, as well as soundproofing, insulation, and water resistance. Nowadays, car windows also come equipped with advanced features such as intelligent lifting and displaying, providing passengers with better comfort and convenience.

- Connectivity domain

Vehicle connectivity is a device that can connect a vehicle to an external and enable data transmission. It allows drivers to use mobile applications or other services, such as media streaming and OTA, etc.

- IMS domain

In-vehicle monitor system. IMS is a safety system. It can monitor the driver's status in real-time and alert the driver to situations such as lack of attention or fatigue, thereby improving driving safety.

- Light domain

The lighting installed on vehicles not only improves night-time driving safety but also enhances vehicle aesthetics and personalization.

8.1.3 Time Sensitive

The system must be able to quickly and reliably perceive the environment and make decisions and reactions in milliseconds or microseconds to protect the safety of the driver and other road users, especially in ADAS systems, therefore time sensitivity must be considered.

For drivers, considering time sensitivity and real-time performance is crucial for their driving experience. It will affect functional safety. If the system's response time is too slow, the driver may feel confused, disappointed, or insecure. Conversely, if the system's response time is fast, the driver can use the vehicle's functions more easily and enjoy a better driving experience.

8.1.4 Functional Safety

Functional safety requirements in automotive refer to a set of measures taken during the vehicle design and development process to prevent or mitigate hazards and injuries caused by system malfunction. Considering functional safety requirements improves the reliability of vehicle systems, ensures the proper functioning of critical functions, reduces accidents and unexpected incidents, and protects the lives and property of passengers and other road users.

8.2 Definition of Command Items

Type	Description
Request	Request to Smart Cabin Domain controller
Response	Response from Smart Cabin Domain controller

TABLE 7-Item Type

Data Type	Description
BOOLEAN	Represent true or false, used for logical operations and conditional statements.
FLOAT	It is a 4 bytes FLOAT number with positive or negative numbers and decimals
int 8	It is an 8-bit signed integer with a range from -128 to 127.
uint 8	It is an 8-bit unsigned integer with a range from 0 to 255.
uint 16	It is a 16-bit unsigned integer with a range from 0 to 65535 or indicates 2 bytes of data.
uint 32	It is a 32-bit unsigned integer with a range from 0 to 4,294,967,295 or indicates 4 bytes of data.
enum	A special data type that enables a variable to be a set of predefined constants. The variable must be equal to one of the values that have been predefined for it.

TABLE 8-Value Type

Unit	Description
mm	Distance measured in millimeters
cm	Distance measured in centimeters
m	Distance measured in meters
km	Distance measured in kilometers
inch	Distance measured in inches
km/h	Speed measured in kilometers per hour
m/s	Speed measured in meters per second
m/s^2	Acceleration measured in meters per second squared
cm/s^2	Acceleration measured in centimeters per second squared
ml	Volume measured in milliliters
l	Volume measured in liters
cm^3	Volume measured in cubic centimeters
Celsius	Temperature measured in degree Celsius
degrees	Angle measured in degrees
degrees/s	Angular speed measured in degrees per second
W	Power measured in watts
kW	Power measured in kilowatts
PS	Power measured in horsepower
kWh	Energy consumption measured in kilowatt-hours
g	Mass measured in grams
kg	Mass measured in kilograms
lbs	Mass measured in pounds
V	Electric potential measured in volts
A	Electric current measured in amperes
Ah	Electric charge measured in ampere hours
ms	Time measured in milliseconds
s	Time measured in seconds
min	Time measured in minutes
h	Time measured in hours
day	Time measured in days
weeks	Time measured in weeks
months	Time measured in months
years	Time measured in years

UNIX Timestamp	Unix time is a system for describing a point in time
mbar	Pressure measured in millibars
Pa	Pressure measured in pascal
kPa	Pressure measured in kilopascal
stars	Rating measured in stars
g/s	Mass per time measured in grams per second
g/km	Grams per kilometer
kWh/100km	Kilowatt hours per 100 kilometers
ml/100km	Milliliter per 100 kilometers
l/100km	Liter per 100 kilometers
l/h	Liter per hour
mpg	Miles per gallon
N	Newton
Nm	Newton meter
rpm	Revolutions per minute
Hz	Frequency
ratio	Ratio
percent	Percent
nm/km	Nano meter per kilometer
kN	Kilo newton
dBm	Decibel milliwatt

TABLE 9-Unit

8.3 Command Workflow

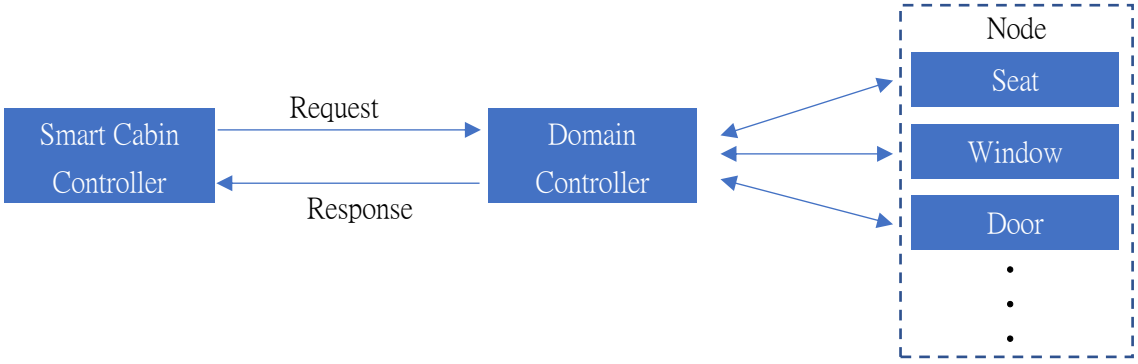


Figure 8 Command Workflow

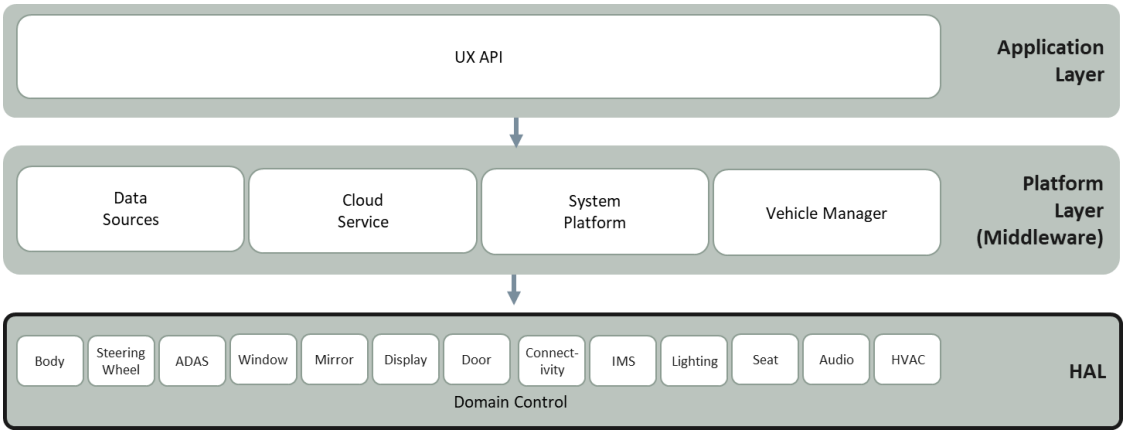
Command requests to the Smart Cabin domain controller or get the response command from the Smart Cabin domain controller.

The data content of the command request or get the data content of the response.

9.0 Vehicle Domain Control (HAL APIs)

Integrating controls from different domains (such as seats, windows, HVAC systems, doors, etc.) into HAL APIs can further enhance the following aspects:

- Hardware abstraction: HAL APIs provide the hardware abstraction layer to different field devices on the vehicle so that these devices can be accessed and controlled more easily. It reduces direct interaction between developers and hardware, improving development efficiency and maintainability.
- Agnostic: Integrating control from different areas into a HAL API ensures compatibility across other models and manufacturers. This means developers can cross multiple platforms without having to modification.
- Simplify the development process: Integrating control from different domains can significantly simplify the development process. Developers can use the same API interface to access seats, windows, HVAC systems, doors, and more, rather than learning and managing multiple APIs or interfaces.



9.1 Steering Wheel Domain

- Steering wheel control Commands

Steering wheel control functions allow the driver to operate various features of the car while keeping their hands on the wheel, minimizing distractions and helping to ensure safe driving.

- KeyEventHome
  - Description: Key Event (Home)
  - Data Type: enum
  - Type: Request
    - ✓ Message: Press, Click, Double Click, Hold, Hold Repeat
- KeyEventBack
  - Description: Key Event (Back)
  - Data Type: enum

- Type: Request
  - ✓ Message: Press, Click, Double Click, Hold, Hold Repeat

- KeyEventUp
  - Description: Key Event (Up)
  - Data Type: enum
  - Type: Request
    - ✓ Message: Press, Click, Double Click, Hold, Hold Repeat
- KeyEventDown
  - Description: Key Event (Down)
  - Data Type: enum
  - Type: Request
    - ✓ Message: Press, Click, Double Click, Hold, Hold Repeat
- KeyEventLeft
  - Description: Key Event (Left)
  - Data Type: enum
  - Type: Request
    - ✓ Message: Press, Click, Double Click, Hold, Hold Repeat
- KeyEventRight
  - Description: Key Event (Right)
  - Data Type: enum
  - Type: Request
    - ✓ Message: Press, Click, Double Click, Hold, Hold Repeat
- KeyEventForward
  - Description: Key Event (Forward)
  - Data Type: enum
  - Type: Request
    - ✓ Message: Press, Click, Double Click, Hold, Hold Repeat
- KeyEventBackward
  - Description: Key Event (Backward)
  - Data Type: enum
  - Type: Request
    - ✓ Message: Press, Click, Double Click, Hold, Hold Repeat
- KeyEventPlay
  - Description: Key Event (Play)
  - Data Type: enum

- Type: Request
  - ✓ Message: Press, Click, Double Click, Hold, Hold Repeat
- KeyEventMode
  - Description: Key Event (Mode)
  - Data Type: enum
  - Type: Request
    - ✓ Message: Press, Click, Double Click, Hold, Hold Repeat

- Steering wheel hands off detection Commands

  The purpose of steering wheel hands-off detection is to ensure that the driver maintains control of the vehicle at all times while it is in motion.

  - HandOffDetectStatus
    - Description: Ensure that the driver maintains control of the vehicle
    - Data Type: enum
    - Type: Request
      - ✓ Message: Hand Off, Hand On, Warning
  - HandOffDetectTime
    - Description: Hand Off Detect Time
    - Unit = s
    - Data Type: unit8
    - Type: Response
      - ✓ Message: Resolution = 1s per bit
  - HandOffDetectCalbration
    - Description: Hand Off Detect Calbration
    - Data Type: BOOLEAN
    - Type: Request
      - ✓ Message: Off, On
    - Data Type: enum
    - Type: Response
      - ✓ Message: Off, Process, Pass, Failed
  - SteeringWheelHeaterEnable
    - Description: Warms up the surface of the steering wheel
    - Data Type: BOOLEAN
    - Type: Request
      - ✓ Message: Off, On
    - Data Type: enum
    - Type: Response

- ✓ Message: Off, On, Overheating protection

9.2 Seat Domain

Vehicle seating system integrates functions such as adjustment, heating and so on that provide a comfortable seating experience for driver and passengers. Adjusting the seat allows the person to achieve a comfortable position.

The Seat Command message is used by the system to control the seat function and indicate the status.

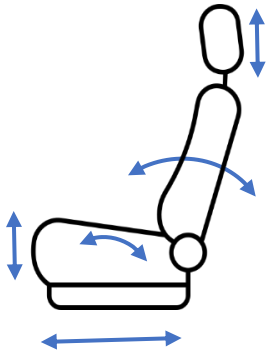


Figure 9 Seat Adjust Direction

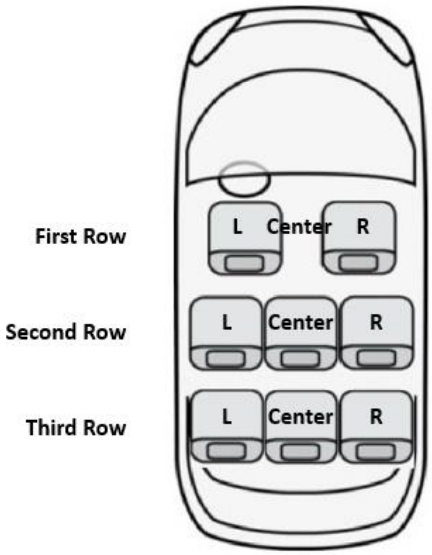


Figure 10 Seat Location

- SeatForwardBackwardPositionFL
  - Description: Adjust First Row-Left seat position
  - Unit: mm
  - Data Type: uint16
  - Type: Request
    - ✓ Message: Resolution = 1mm per bit
  - Data Type: uint16

- Type: Response
  - ✓ Message: Resolution = 1mm per bit

- SeatUpDownPositionFL
  - Description: Adjust First Row-Left seat position
  - Unit: mm
  - Data Type: uint16
  - Type: Request
    - ✓ Message: Resolution = 1mm per bit
  - Data Type: uint16
  - Type: Response
    - ✓ Message: Resolution = 1mm per bit
- SeatCushionTiltPositionFL
  - Description: Adjust First Row-Left cushion tilt position
  - Unit: degree
  - Data Type: uint16
  - Type: Request
    - ✓ Message: Resolution = 0.5° per bit
  - Data Type: uint16
  - Type: Response
    - ✓ Message: Resolution = 0.5° per bit
- SeatBackrestTiltPositionFL
  - Description: Adjust First Row-Left backrest tilt position
  - Unit: degree
  - Data Type: uint16
  - Type: Request
    - ✓ Message: Resolution = 0.5° per bit
  - Data Type: uint16
  - Type: Response
    - ✓ Message: Resolution = 0.5° per bit
- HeadrestUpDownPositionFL
  - Description: Adjust First Row-Left headrest position
  - Unit: mm
  - Data Type: uint16
  - Type: Request
    - ✓ Message: Resolution = 1mm per bit

- Data Type: uint16
- Type: Response
  - ✓ Message: Resolution = 1mm per bit

- SeatBeltStatusFL
  - Description: Show the First Row-Left seat belt status
  - Data Type: BOOLEAN
  - Type: Response
    - ✓ Message: Fasten, Not fasten
- SeatForwardBackwardPositionFR
  - Description: Adjust First Row-Right seat position
  - Unit: mm
  - Data Type: uint16
  - Type: Request
    - ✓ Message: Resolution = 1mm per bit
  - Data Type: uint16
  - Type: Response
    - ✓ Message: Resolution = 1mm per bit
- SeatUpDownPositionFR
  - Description: Adjust First Row-Right seat position
  - Unit: mm
  - Data Type: uint16
  - Type: Request
    - ✓ Message: Resolution = 1mm per bit
  - Data Type: uint16
  - Type: Response
    - ✓ Message: Resolution = 1mm per bit
- SeatCushionTiltPositionFR
  - Description: Adjust First Row-Right cushion tilt position
  - Unit: degree
  - Data Type: uint16
  - Type: Request
    - ✓ Message: Resolution = 0.5° per bit
  - Data Type: uint16
  - Type: Response
    - ✓ Message: Resolution = 0.5° per bit

- SeatBackrestTiltPositionFR
  - Description: Adjust First Row-Right backrest tilt position
  - Unit: degree
  - Data Type: uint16
  - Type: Request
    - ✓ Message: Resolution = 0.5° per bit
  - Data Type: uint16
  - Type: Response
    - ✓ Message: Resolution = 0.5° per bit
- HeadrestUpDownPositionFR
  - Description: Adjust First Row-Right headrest position
  - Unit: mm
  - Data Type: uint16
  - Type: Request
    - ✓ Message: Resolution = 1mm per bit
  - Data Type: uint16
  - Type: Response
    - ✓ Message: Resolution = 1mm per bit
- SeatBeltStatusFR
  - Description: Show the First Row-Right seat belt status
  - Data Type: BOOLEAN
  - Type: Response
    - ✓ Message: Fasten, Not fasten

9.3 Body Domain

Body usually refers to the entire body of a vehicle, including the front and rear end as well as the chassis system. Provide meter, turn light, and other functions. These functions can provide stability and control to the driver, ensuring that the vehicle is more stable and reliable during maneuvering, thereby enhancing driver safety and confidence.

The Body Command message is used to control some functions and indicate the body status.

- Meter 

A meter is a device that measures the amount of something that is used. Meter Speed is a meter which measures the vehicle speed. Meter gear indicates the power output or driving direction according to the driving conditions.

- MeterSpeed
  - Unit: km/h
  - Data Type: uint16
  - Type: Response
    - ✓ Message: Resolution=0.125 km/h per bit
- MeterGear
  - Data Type: enum
  - Type: Response
    - ✓ Message: Park, Low, D2, D3, D, Reverse, Neutral, Failure

- Turn Lamp 

  Turn Lamp is a lighting device indicating driving direction.
  - TurningSwitch
    - Data Type: enum
    - Type: Request
      - ✓ Message: Off, Left, Right
  - TurnLampLeftStatus
    - Data Type: BOOLEAN
    - Type: Response
      - ✓ Message: Inactive, Active
  - TurnLampRightStatus
    - Data Type: BOOLEAN
    - Type: Response
      - ✓ Message: Inactive, Active
- Brake Light Switch

  Brake light switch is a device that activates the brake lights to signal when the brakes are applied. And indicate the Break Switch status.
  - BrakeLightSwitchStatus
    - Data Type: BOOLEAN
    - Type: Response
      - ✓ Message: Inactive, Active
- Steering Wheel Angle 

  Steering Wheel Angle Validity is used to verify if the angle of the steering wheel rotation is effective, measuring the angle of the steering wheel rotation.

  - SteeringWheelAngelValidity

- Data Type: BOOLEAN
- Type: Response
  - ✓ Message: Invalid, Valid

- SteeringWheelAngel
  - Unit: deg
  - Type: int 16
  - Type: Response
    - ✓ Message: Resolution=0.1 deg, Positive value is clockwise

- ECO Mode 

  ECO Mode is used to reduce the output power consumption.
  - ECOMode
    - Data Type: BOOLEAN
    - Type: Request
      - ✓ Message: Disabled, Enabled
    - Data Type: enum
    - Type: Response
      - ✓ Message: Inactive, Active, Failed

- Hill Descent Control (HDC) 

  Hill Descent Control Status helps the vehicle to safely navigate downhill sections. And indicate the Hill Descent Control status.
  - HDCStatus
    - Time Sensitive
    - Functional Safety
    - Data Type: enum
    - Type: Response
      - ✓ Message: Inactive, Active, Standby, Failed

- Hill Start Assist 

  Hill Start Assist helps the driver to control the vehicle more easily and safely when starting uphill. And indicate the Hill Start Assist status.
  - HillStartAssistActive
    - Time Sensitive
    - Functional Safety
    - Data Type: BOOLEAN
    - Type: Request

        - ✓ Message: Disabled, Enabled
    - Data Type: enum
    - Type: Response
        - ✓ Message: Inactive, Active, Failed

- Auto Vehicle Hold (AVH) 

  Auto Vehicle Hold is a vehicle safety feature that automatically maintains the vehicle in a stationary state during parking without requiring the driver to step on the brake pedal. And indicate the Auto Vehicle Hold Status.

    - AVHStatus
        - Time Sensitive
        - Data Type: BOOLEAN
        - Type: Request
            - ✓ Message: Disabled, Enabled
        - Data Type: enum
        - Type: Response
            - ✓ Message: Inactive, Active, Failed

- Anti-lock Braking System (ABS) 

  Anti-lock Braking System is a vehicle safety device that prevents the wheels from locking up due to excessive braking force during emergency braking, thereby maintaining vehicle stability and preventing the vehicle from skidding or losing control. And indicate the Antilock Brake System status.

    - ABSActive
        - Time Sensitive
        - Functional Safety
        - Data Type: enum
        - Type: Response
            - ✓ Message: Inactive, Active, Failed

- Electronic Stability Control (ESC) 

  Electronic Stability Control is a vehicle dynamic control system that helps drivers avoid loss of control and maintain vehicle stability. And indicate the Electronic Stability Control status.

    - ESCStatus
        - Time Sensitive
        - Functional Safety
        - Data Type: BOOLEAN

  - Type: Request
    - ✓ Message: Disabled, Enabled
  - Data Type: enum
  - Type: Response
    - ✓ Message: Inactive, Active, Failed

- Traction Control System (TCS)

  Traction Control System prevents the vehicle from slipping or losing control during start-up or driving. And indicate the Traction Control System status.

  - TCSStatus
    - Time Sensitive
    - Functional Safety
    - Data Type: BOOLEAN
    - Type: Request
      - ✓ Message: Disable , Enable
    - Data Type: enum
    - Type: Response
      - ✓ Message: Inactive, Active, Failed

- Electronic Brake Force Distribution System (EBD)

  Electronic Brake Force Distribution automatically adjusts the brake force of the vehicle through the electronic controller to improve braking performance and safety. And indicate the Electronic Brake Force distribution status.

  - EBDSystemStatus
    - Time Sensitive
    - Functional Safety
    - Data Type: BOOLEAN
    - Type: Response
      - ✓ Message: Normal, Failed

- Hydraulic Brake Boost (HBB)

  Hydraulic Brake Boost helps the driver to control vehicle braking more easily, improve braking performance, and safety. And indicate the Hydraulic Brake Boost status.

  - HBBStatus
    - Functional Safety
    - Data Type: BOOLEAN
    - Type: Response
      - ✓ Message: Normal, Failed

- Airbag System Lamp 

  Airbag system lamp indicates whether the vehicle's airbag system is functioning properly.

  - AirbagSystemLampStatus
    - Data Type: enum
    - Type: Response
      - ✓ Message: Off, On, Failed
- Antitheft Status

  Antitheft Status indicate whether the vehicle's antitheft system is activated or has been attacked.

  - AntitheftStatus
    - Data Type: enum
    - Type: Response
      - ✓ Message: Disarm, Arm, Wait for Rearm, Alarm, Failed
- Front Wiper Switch Status 

  Front Wiper Switch Status indicates the current status of the front wipers.

  - FrontWiperSwitchStatus
    - Data Type: enum
    - Type: Response
      - ✓ Message: Off, Int (Interim), Low Speed, High Speed, Mist
- Front Washer Switch Status 

  Front Washer Switch Status indicates the current status of the washer.

  - FrontWasherSwitchStatus
    - Data Type: enum
    - Type: Response
      - ✓ Message: Off, On, Failed
- Electric Power Assisted Steering (EPAS)

  Electric Power Assisted Steering uses an electric motor to assist the driver in steering the vehicle. And indicate the Electric Power Assisted Steering status.

  - EPASStatus
    - Data Type: BOOLEAN
    - Type: Response
      - ✓ Message: Normal, Failed

- Fog Lamp Status 

  Fog Lamp Status indicates the current status of the fog lamp.

  - FrontFogLampDriverStatus
    - Data Type: BOOLEAN
    - Type: Request
      - ✓ Message: Disable , Enable
    - Data Type: enum
    - Type: Response
      - ✓ Message: Inactive, Active, Failed
  - RearFogLampStatus
    - Data Type: BOOLEAN
    - Type: Request
      - ✓ Message: Disable , Enable
    - Data Type: enum
    - Type: Response
      - ✓ Message: Inactive, Active, Failed

9.4 ADAS Domain

ADAS helps to improve the driving experience for drivers and reduce the risk of traffic accidents. Provide lane keeping, automatic braking, and other functions. These functions can be enabled or disabled according to needs, enhancing driving safety and convenience.

The ADAS Command message is used by the system to control the ADAS function and indicate the status.

- Autonomous Emergency Braking (AEB) 

  Autonomous Emergency Braking is a type of automotive safety technology that aims to assist drivers in reducing collision risk and even automatically slow down or stop the vehicle to avoid collision when the driver is unable to react in time. And indicate the Autonomous Emergency Braking status.

  - AEBONOFF
    - Description: Switch AEB open
    - Time Sensitive
    - Functional Safety
    - Data Type: BOOLEAN
    - Type: Request
      - ✓ Message: Disable, Enable

- Data Type: enum
- Type: Response
  - ✓ Message: Inactive, Active, Standby, Failed

- Forward Collision Warning (FCW) 

Forward Collision Warning is a type of automotive safety technology that aims to assist drivers in reducing collision risk by sensing traffic conditions ahead and issuing warnings to remind drivers to pay attention to the road. And indicate the Forward Collision Warning status.

- FCWONOFF
  - Description: Switch FCW open
  - Time Sensitive
  - Functional Safety
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Disable , Enable
  - Data Type: enum
  - Type: Response
    - ✓ Message: Inactive, Active, Standby, Failed

- Lane Departure Warning (LDWS) 

Lane Departure Warning is a type of automotive safety technology that aims to assist drivers in staying within their lane by issuing warnings when the driver deviates from the lane, reminding the driver to return to the correct lane. And indicate the Lane Departure Warning status.

- LDWSONOFF
  - Description: Switch LDWS open
  - Time Sensitive
  - Functional Safety
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Disable , Enable
  - Data Type: enum
  - Type: Response
    - ✓ Message: Inactive, Active, Standby, Failed

- Pedestrian Detection System (PDS) 

Pedestrian Detection System is a vehicle safety technology that uses advanced image processing and machine learning techniques to identify pedestrians ahead and issue warnings, enhancing driver awareness of pedestrians and reducing pedestrian accidents.

- PDSONOFF
  - Description: Switch PDS open
  - Time Sensitive
  - Functional Safety
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Disable , Enable
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, Standby, Active, Error

- Around View Monitoring (AVM)

  Around View Monitoring is a type of in-vehicle surveillance system that uses multiple cameras to capture images of the vehicle's surroundings, and displays a composite panoramic image on the vehicle display screen to assist the driver in performing parking and reversing maneuvers . And indicate the Around View Monitoring status .

  - AVMONOFF
    - Description: Switch AVM open
    - Time Sensitive
    - Functional Safety
    - Data Type: BOOLEAN
    - Type: Request
      - ✓ Message: Disable , Enable
    - Data Type: enum
    - Type: Response
      - ✓ Message: Inactive, Active, Failed

- Lane Keep Assist (LKA) 

  Lane Keep Assist is a type of in-vehicle safety assistance system that uses technologies such as cameras or radar sensors to monitor whether the vehicle is deviating from the centerline of the lane while driving. When necessary, the electronic control system can automatically control the vehicle to keep it driving on the centerline of the lane, avoiding lane departure and reducing the occurrence of traffic accidents. And indicate the Lane Keep Assist status .

- LKAONOFF
  - Description: Switch LKA open
  - Time Sensitive
  - Functional Safety
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Disable , Enable
  - Data Type: enum
  - Type: Response
    - ✓ Message: Inactive, Standby, Left Active, Right Active, Error, Failed

- Moving Object Detection (MOD) 

Moving Object Detection is an in-vehicle safety assistance system that uses technologies such as cameras or radar sensors to monitor the vehicle's surrounding environment, detecting and identifying moving objects such as pedestrians, vehicles, and other objects, and warning the driver to pay attention.

- MODONOFF
  - Description: Switch MOD open
  - Time Sensitive
  - Functional Safety
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Disable , Enable
  - Data Type: enum
  - Type: Response
    - ✓ Message: Inactive, Active, Standby, Error, Failed

- Blind Spot Detection (BSD) 

Blind Spot Detection is a vehicle safety feature that uses radar, ultrasonic, or camera sensor technology to monitor the environment around the vehicle, helping drivers better identify the surrounding environment while driving, and thereby reducing the risk of traffic accidents.

- BSDONOFF
  - Description: Switch BSD open
  - Time Sensitive

- Functional Safety
- Data Type: BOOLEAN
- Type: Request
  - ✓ Message: Disable , Enable

- BSDConfigurationMLOnly
  - Description: Switch BSD Configuration (Mechine Learning Only)
  - Time Sensitive
  - Functional Safety
  - Data Type: BOOLEAN
  - Type: Response
    - ✓ Message: No BSD, BSD On Board
- BSDStatus
  - Description: Show the outside of temperature
  - Time Sensitive
  - Functional Safety
  - Data Type: enum
  - Type: Response
    - ✓ Message: Inactive, Active, Failed

9.5 HVAC Domain

HVAC control system is designed to maintain a comfortable temperature inside the vehicle in different weather conditions. Typically, HVAC control system includes functions such as heating, ventilation and air conditioning.

The HAVC Command message is used by the system to control the HAVC function and indicate the status.

- OutsideTemperature
  - Description: Show the outside of temperature
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Resolution = 1.0 Celsius per bit
- ACMode **A/C**
  - Description: Switch AC open
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: BOOLEAN

- Type: Response
  - ✓ Message: Off, On

- ACTemperature
  - Description: Adjust AC temperature
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Resolution = 0.5 Celsius per bit
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Resolution = 0.5 Celsius per bit, Error

- ACFanSpeed 
  - Description: Adjust AC fan speed
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level = 0~7 Default = 2
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level = 0~7, Error

- WindDirection 
  - Description: Adjust wind direction
  - Data Type: enum
  - Type: Request
    - ✓ Message: Body, Body &Feet, Feet, Feet & Window
  - Data Type: enum
  - Type: Response
    - ✓ Message: Body, Body &Feet, Feet, Feet & Window, Error

- AirCirculationStatus 
  - Description: Switch air circulation mode
  - Data Type: enum
  - Type: Request
    - ✓ Message: Off, Inner circulation, Outer circulation
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, Inner circulation, Outer circulation, Error

- WindowDefoggingFront 
  - Description: Switch front window defogging mode
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- WindowDefoggingRear 
  - Description: Switch rear window defogging mode
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error

9.6 Door Domain

The doors provide passengers in and out of the vehicle and are equipped with many advanced features such as electric locking devices. This feature enhances the safety and convenience of the car, making it easier for passengers to enter and exit.

The Door Command message is used by the system to control the door function and indicate the status.

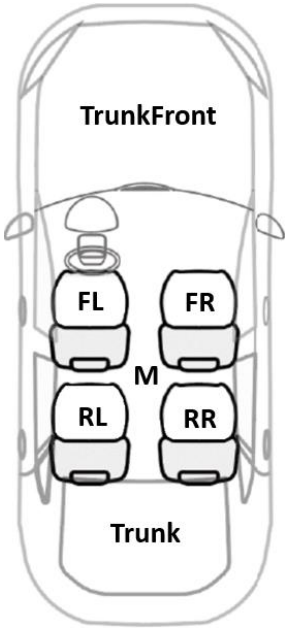


Figure 11 Door Locations

- DoorOpenFL
  - Description: Switch Front-Left door open
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Close, Open
  - Data Type: enum
  - Type: Response
    - ✓ Message: Closed, Open, Error
- DoorOpenFR
  - Description: Switch Front-Right door open
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Close, Open
  - Data Type: enum
  - Type: Response
    - ✓ Message: Closed, Open, Error
- DoorOpenRL
  - Description: Switch Rear-Left door open
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Close, Open

- Data Type: enum
- Type: Response
  - ✓ Message: Closed, Open, Error

- DoorOpenRR
  - Description: Switch Rear-Right door open
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Close, Open
  - Data Type: enum
  - Type: Response
    - ✓ Message: Closed, Open, Error
- DoorOpenTrunk
  - Description: Switch trunk open
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Close, Open
  - Data Type: enum
  - Type: Response
    - ✓ Message: Closed, Open, Error
- DoorOpenTrunkFront
  - Description: Switch front trunk open
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Close, Open
  - Data Type: enum
  - Type: Response
    - ✓ Message: Closed, Open, Error
- SlideDoorOpenLeft
  - Description: Switch left slide door open
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Close, Open
  - Data Type: enum
  - Type: Response
    - ✓ Message: Closed, Open, Error

- SlideDoorOpenRight
  - Description: Switch right slide door open
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Close, Open
  - Data Type: enum
  - Type: Response
    - ✓ Message: Closed, Open, Error
- DoorLockFL
  - Description: Switch Front-Left door lock
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Unlock, Lock
  - Data Type: enum
  - Type: Response
    - ✓ Message: Unlock, Lock, Error
- DoorLockFR
  - Description: Switch Front-Right door lock
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Unlock, Lock
  - Data Type: enum
  - Type: Response
    - ✓ Message: Unlock, Lock, Error
- DoorLockRL
  - Description: Switch Rear-Left door lock
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Unlock, Lock
  - Data Type: enum
  - Type: Response
    - ✓ Message: Unlock, Lock, Error
- DoorLockRR
  - Description: Switch Rear-Right door lock
  - Data Type: BOOLEAN

- Type: Request
  - ✓ Message: Unlock, Lock
- Data Type: enum
- Type: Response
  - ✓ Message: Unlock, Lock, Error

- DoorLockTrunk
  - Description: Switch trunk lock
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Unlock, Lock
  - Data Type: enum
  - Type: Response
    - ✓ Message: Unlock, Lock, Error
- DoorLockTrunkFront
  - Description: Switch front trunk lock
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Unlock, Lock
  - Data Type: enum
  - Type: Response
    - ✓ Message: Unlock, Lock, Error
- SlideDoorLockLeft
  - Description: Switch left slide door lock
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Unlock, Lock
  - Data Type: enum
  - Type: Response
    - ✓ Message: Unlock, Lock, Error
- SlideDoorLockRight
  - Description: Switch right slide door lock
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Unlock, Lock
  - Data Type: enum

- Type: Response
  - ✓ Message: Unlock, Lock, Error

9.7 Mirror Domain

Car mirrors consist of rear-view and side-view mirrors, adjustable for viewing angle. They utilize conventional mirrors, camera-monitors, or other devices to provide a clear view of the vehicle's rear, side, or front within the driver's field of vision.

The Mirror Command message is used by the system to control the Mirror function and indicate the status.Mirrors needs to follow regulations that UN R46 defines five views around the vehicle.

Class I: Rear-view device

The field of vision shall be such that the driver can see at least a 20 m wide, flat, horizontal portion of the road centred on the vertical longitudinal median plane of the vehicle and extending from 60 m behind the driver's ocular points to the horizon.

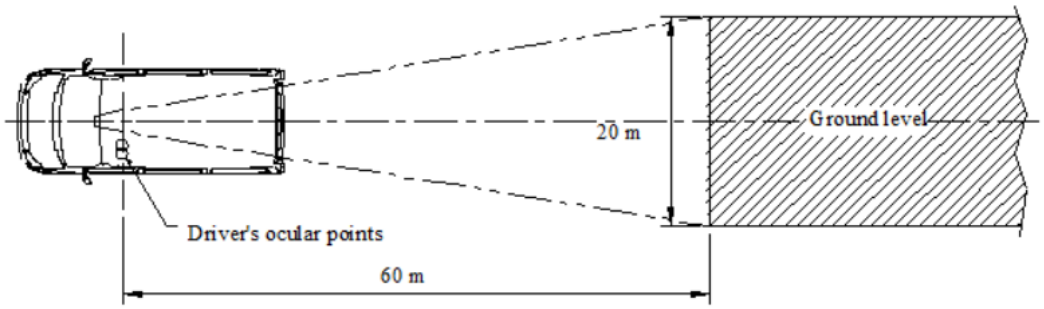


Figure 12 Class I fields of vision

- Class II: Main rear-view device

The field of vision shall be such that the driver can see at least a 5m wide, flat, horizontal portion of the road, which is bounded by a plane which is parallel to the median longitudinal vertical plane and passing through the outermost point of the vehicle on the driver's side of the vehicle and extends from 30 m behind the driver's ocular points to the horizon.

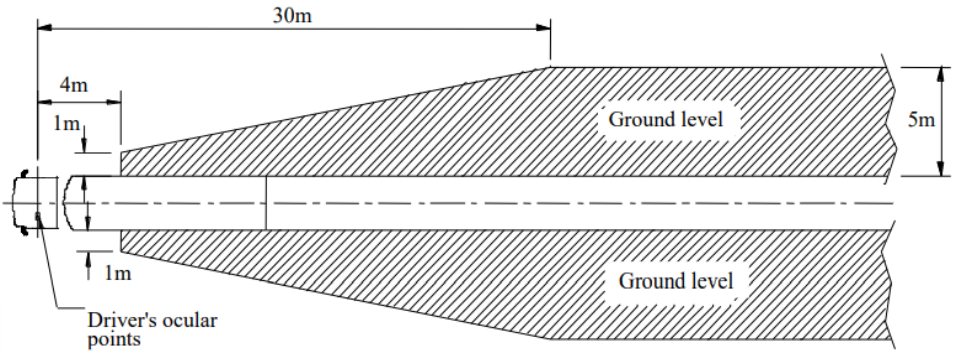


Figure 13 Class II fields of vision

- Class III: Main rear-view device

The field of vision shall be such that the driver can see at least a 4 m wide flat, horizontal portion of the road which is bounded by a plane parallel to the median longitudinal vertical plane passing through the outermost point of the vehicle on the passenger's side and which extends from 20 m behind the driver's ocular points to the horizon.

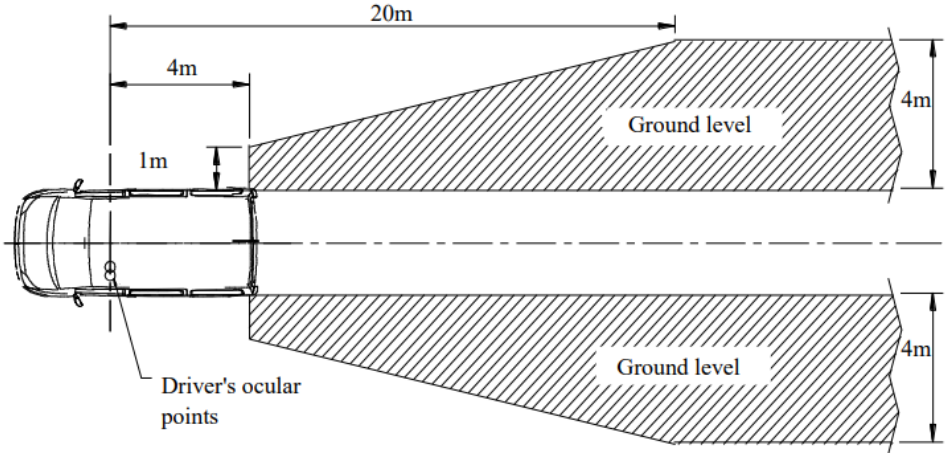


Figure 14 Class III fields of vision

- Class IV: Wide-angle view device

The field of vision shall be such that the driver can see at least a 15 m wide, flat, horizontal portion of the road, which is bounded by a plane parallel to the median longitudinal vertical plane of the vehicle and passing through the outermost point of the vehicle on the driver's side and which extends from at least 10 m to 25 m behind the driver's ocular points.

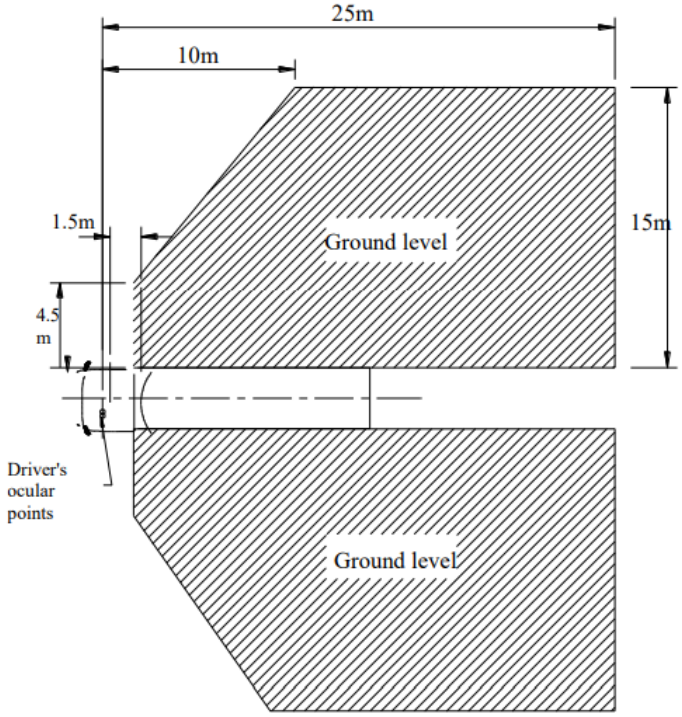


Figure 15 Class IV fields of vision

- Class V: Close-proximity view device

  The field of vision shall be such that the driver can see a flat horizontal portion of the road along the side of the vehicle, bounded by the following vertical planes.

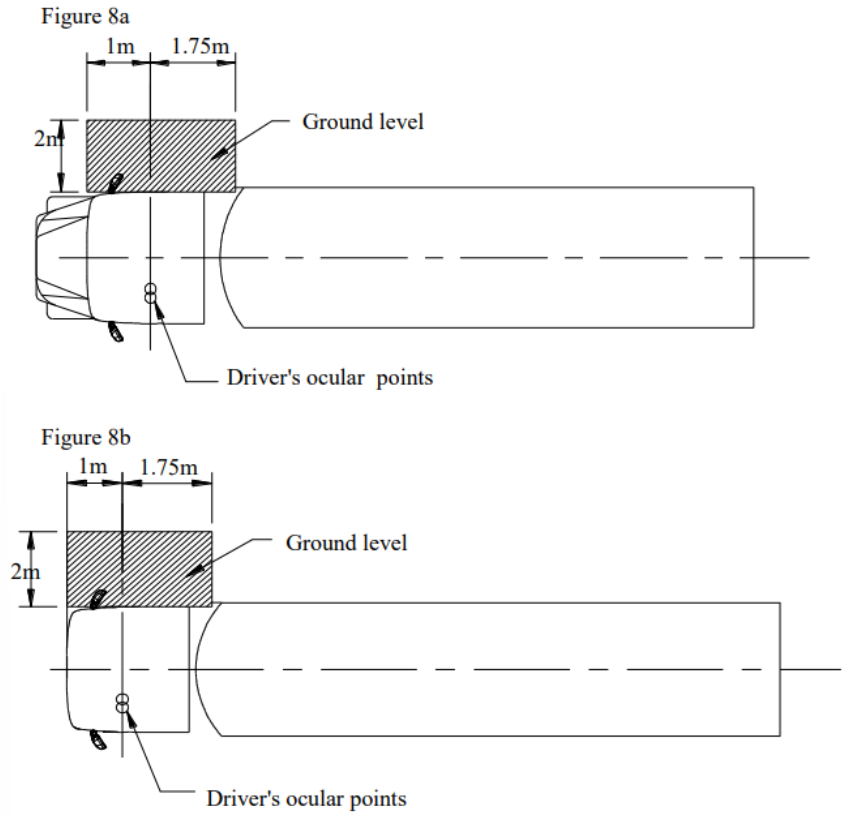


Figure 16 Class V fields of vision

- Class VI: front-view device

  The field of vision shall be such that the driver can see at least a flat horizontal portion of the road.

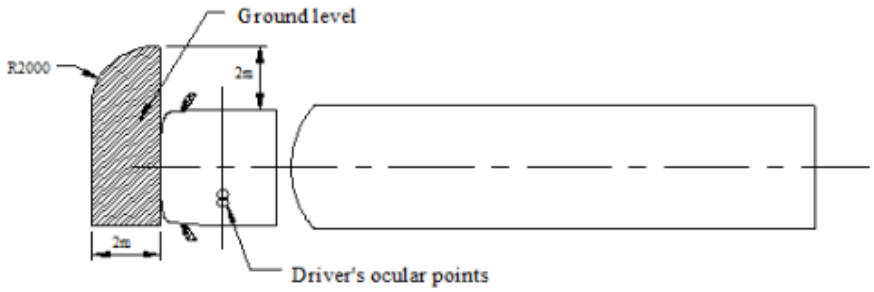


Figure 17 Class VI fields of vision

- List of Mirror Installation Quantities

Category	Transportation	Seats	Wheel	Total Weight
M1	Passenger	<9	<=4	
M2	Passenger	>9	<=4	<5t
M3	Passenger	>9	<=4	>=5t

N1	Goods		<=4	<3.5t
N2<=7.5t	Goods		<=4	>3.5t and <=7.5t
N2>7.5t	Goods		<=4	>7.5t and <12
N3	Goods		<=4	>=12

TABLE 10- Vehicle Category Definition

Category	Class I	Class II	Class III	Class IV	Class V	Class VI
M1	Compulsory	Optional	Compulsory	Optional	Optional	Optional
M2	Optional	Compulsory	Not permitted	Optional	Optional	Optional
M3	Optional	Compulsory	Not permitted	Optional	Optional	Optional
N1	Compulsory	Optional	Compulsory	Optional	Optional	Optional
N2<=7.5t	Optional	Compulsory	Not permitted	Compulsory	Compulsory	Optional
N2>=7.5t	Optional	Compulsory	Not permitted	Compulsory	Compulsory	Compulsory
N3	Optional	Compulsory	Not permitted	Compulsory	Compulsory	Compulsory

TABLE 11- List of Mirror Installation Quantities

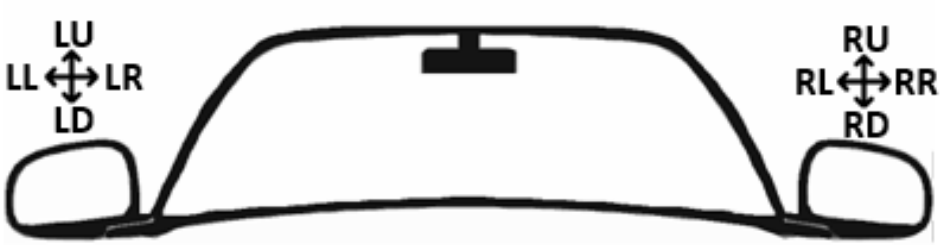


Figure 18 Side View Mirror Direction Definition

- SideviewMirrorRU
  - Description: Right sideview mirror up-side adjustment
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~9, Default=5
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~9, Error=255
- SideviewMirrorRD
  - Description: Right sideview mirror down-side adjustment
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~9, Default=5
  - Data Type: uint8
  - Type: Response

- ✓ Message: Level=0~9, Error=255

- SideviewMirrorRR
  - Description: Right sideview mirror right-side adjustment
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~9, Default=5
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~9, Error=255
- SideviewMirrorRL
  - Description: Right sideview mirror left-side adjustment
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~9, Default=5
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~9, Error=255
- SideviewMirrorLU
  - Description: Left sideview mirror up-side adjustment
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~9, Default=5
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~9, Error=255
- SideviewMirrorLD
  - Description: Left sideview mirror down-side adjustment
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~9, Default=5
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~9, Error=255
- SideviewMirrorLR
  - Description: Left sideview mirror right-side adjustment

- Data Type: uint8
- Type: Request
  - ✓ Message: Level=0~9, Default=5
- Data Type: uint8
- Type: Response
  - ✓ Message: Level=0~9, Error=255

- SideviewMirrorLL
  - Description: Left sideview mirror left-side adjustment
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~9, Default=5
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~9, Error=255
- SideviewMirrorDefoggingR
  - Description: Right sideview mirror defogging function
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: BOOLEAN
  - Type: Response
    - ✓ Message: Off, On
- SideviewMirrorDefoggingL
  - Description: Left sideview mirror defogging function
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: BOOLEAN
  - Type: Response
    - ✓ Message: Off, On
- SideviewMirrorBrightnessR 
  - Description: Right sideview display brightness
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~15, Default=8

- Data Type: uint8
- Type: Response
  - ✓ Message: Level=0~15, Error=255

- SideviewMirrorContrastR 
  - Description: Right sideview display contrast
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~15, Default=8
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~15, Error=255

- SideviewMirrorBrightnessL 
  - Description: Left sideview display brightness
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~15, Default=8
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~15, Error=255

- SideviewMirrorContrastL 
  - Description: Left sideview display contrast
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~15, Default=8
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~15, Default=8, Error=255

- SideviewCameraSelectionR
  - Description: Right sideview camera selection
  - Data Type: enum
  - Type: Request
    - ✓ Message: Class=I~VI
  - Data Type: enum
  - Type: Response
    - ✓ Message: Class=I~VI, Error

- SideviewCameraSelectionL
  - Description: Left sideview camera selection
  - Time Sensitive
  - Functional Safety
  - Data Type: enum
  - Type: Request
    - ✓ Message: Class=I~VI
  - Data Type: enum
  - Type: Response
    - ✓ Message: Class=I~VI, Error
- SideviewMirrorZoomR
  - Description: Right sideview mirror zoom in/out
  - Time Sensitive
  - Functional Safety
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~9, Default=5
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~9, Error=255
- SideviewMirrorZoomL
  - Description: Left sideview mirror zoom in/out
  - Time Sensitive
  - Functional Safety
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~9, Default=5
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~9, Error=255

9.8 Light Domain

Lighting plays a crucial role in providing safe driving and alerting other road users. Common car lighting decorations include interior map light, welcome Light, Ambience Light, etc.

The Light Command message is used to control some functions and indicate the light status.

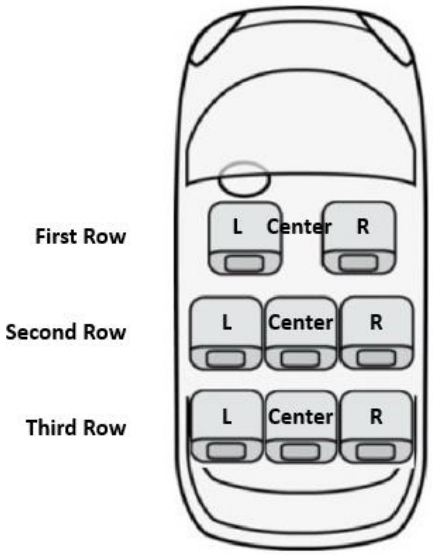


Figure 19 Interior Lighting Locations

- LightStatusFL
  - Description: Switch Front-Left map light
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- LightStatusFR
  - Description: Switch Front-Right map light
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- LightStatusRR
  - Description: Switch Rear-Right map light
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum

- Type: Response
  - ✓ Message: Off, On, Error
- LightStatusRL
  - Description: Switch Rear-Left map light
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- LightStatusM
  - Description: Switch Middle map light
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error

• Welcome Light

Welcome lights are installed on the vehicle's exterior or interior to provide illumination and visual effects when entering or exiting. They're commonly found at the bottom, beneath doors, on the roof, or inside the cabin.

Figure 20 Welcome Light

- WelcomeLightStatusFR
  - Description: Switch Rear-Right welcome light
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum

- Type: Response
  - ✓ Message: Off, On, Error
- WelcomeLightStatusFL
  - Description: Switch Front-Left welcome light
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- WelcomeLightStatusRR
  - Description: Switch Rear-Right welcome light
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- WelcomeLightStatusRL
  - Description: Switch Rear-Left welcome light
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error

• Ambience Light

Ambient lights create a customizable atmosphere inside the vehicle, adjusting colors, brightness, and patterns based on temperature, speed, and some conditions.

- AmbienceLightStatus
  - Description: Switch ambient light
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response

- ✓ Message: Off, On, Error

- AmbientLightModeSwitchStatusFixed
  - Description: Switch ambient light in fixed mode
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- AmbientLightModeSwitchStatusTemperature
  - Description: Switch ambient light by external temperature
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- AmbientLightModeSwitchStatusVolume
  - Description: Switch ambient light by audio volume level
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- AmbientLightModeSwitchStatusSpeed
  - Description: Switch ambient light by vehicle speed
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- AmbientLightModeSwitchStatusCall
  - Description: Switch ambient light by calling

- Data Type: BOOLEAN
- Type: Request
  - ✓ Message: Off, On
- Data Type: enum
- Type: Response
  - ✓ Message: Off, On, Error

- AmbientLightModeSwitchStatusDoor
  - Description: Switch ambient light by door open condition
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- AmbienceLightFixed
  - Description: Ambient light color on fixed mode
  - Data Type: uint8[4]
  - Type: Request
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)
  - Type: Response
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)
- AmbienceLightTemperature
  - Description: Ambient light color by external temperature
  - Data Type: uint8[4]
  - Type: Request
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)
  - Type: Response
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)
- AmbienceLightVolume
  - Description: Ambient light color by audio volume level
  - Data Type: uint8[4]
  - Type: Request
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)
  - Type: Response
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)

- AmbienceLightSpeed
  - Description: Ambient light color by vehicle speed
  - Data Type: uint8[4]
  - Type: Request
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)
  - Type: Response
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)
- AmbienceLightCall
  - Description: Ambient light color in call mode
  - Data Type: uint8[4]
  - Type: Request
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)
  - Type: Response
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)
- AmbienceLightDoor
  - Description: Ambient light color in door open condition
  - Data Type: uint8[4]
  - Type: Request
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)
  - Type: Response
    - ✓ Message: Red(0~255), Green(0~255), Blue(0~255), Reserved(0~1)

9.9 Audio Domain

Audio provides a high-quality music and sound experience, allowing you to enjoy premium audio performance while driving. You can adjust your audio experience to suit your personal preferences using features such as volume control and equalizer.

The Audio Command message is used to control some functions and indicate the audio status.

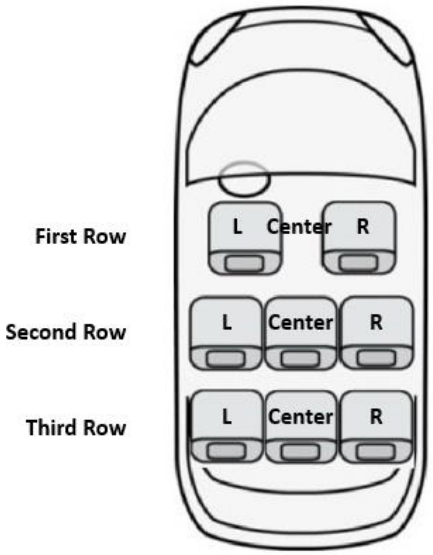


Figure 21 Audio Volume locations

- AudioVolumeFL
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=30
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255
- AudioVolumeFR
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=30
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255
- AudioVolumeRL
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=30
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255
- AudioVolumeRR

- Data Type: uint8
- Type: Request
  - ✓ Message: Level=0~100, Default=30
- Data Type: uint8
- Type: Response
  - ✓ Message: Level=0~100, Error=255

- AudioVolumeSW
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=30
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255
- EQMode 
  - Data Type: enum
  - Type: Request
    - ✓ Message: Manual, Jazz, Vocal, Popular, Rock, Classical, Dance, Default=Manual
  - Data Type: enum
  - Type: Response
    - ✓ Message: Manual, Jazz, Vocal, Popular, Rock, Classical, Dance,
- ManualEQ 
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~15, Default=7, Offset=-7
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~15, Offset=-7
- MultiMediaChannelVolume
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=30
  - Data Type: uint8
  - Type: Response

- ✓ Message: Level=0~100, Error=255

- NavigationChannelVolume
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=30
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255
- CallChannelVolume
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=30
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255
- AlertChannelVolume
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=30
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255
- HighSpeedLoudness
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On, Default=ON
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error

9.10 Display Domain

The car display serves multiple functions. It is used to display vehicle information, navigation maps, multimedia content, and driving assistance systems. The display screen typically features brightness and contrast adjustment capabilities to ensure clear visibility under different lighting conditions.

The Display Command message is used to control some functions and indicate the audio status.

- ClusterBrightness 
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=70
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255
- IVIBrightness 
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=70
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255
- ClusterContrast 
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=70
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255
- IVIContrast 
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=70
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255
- ClusterColor 
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=50
  - Data Type: uint8
  - Type: Response

- ✓ Message: Level=0~100, Error=255
- ✓

- IVIColor 
  - Data Type: uint8
  - Type: Request
    - ✓ Message: Level=0~100, Default=50
  - Data Type: uint8
  - Type: Response
    - ✓ Message: Level=0~100, Error=255

9.11 Connectivity Domain

Automotive connectivity applications encompass technologies such as WiFi and Bluetooth, providing wireless network connectivity and enabling wireless communication between devices within vehicles. This facilitates information transfer and interaction between vehicles, passengers, the external environment, and other connected entities.

The Connectivity Command message is used to control some functions and indicate the audio status.

- WiFiSwitchStatus
  - Description: Switch WiFi
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- WiFiSSID
  - Description: Input WiFi SSID
  - Data Type: string
  - Type: Request
    - ✓ Message: WiFi SSID
  - Data Type: string
  - Type: Response
    - ✓ Message: WiFi SSID
- WiFiPassword
  - Description: Input WiFi Password
  - Data Type: string

- Type: Request
  - ✓ Message: WiFi Password
- WiFiIPv4Adress
  - Description: Input WiFi IPv4 Address
  - Data Type: uint8[4]
  - Type: Request
    - ✓ Message: WiFi IPv4 Address
  - Data Type: uint8[4]
  - Type: Response
    - ✓ Message: WiFi IPv4 Address
- WiFiIPv6Address
  - Description: Input WiFi IPv6 Address
  - Data Type: array[6]
  - Type: Request
    - ✓ Message: WiFi IPv6 Address
  - Data Type: array[6]
  - Type: Response
    - ✓ Message: WiFi IPv6 Address
- WiFiSecurityType
  - Description: Input WiFi SecurityType
  - Data Type: string
  - Type: Request
    - ✓ Message: WEP, WPA, WPA2, WPA3
  - Data Type: string
  - Type: Response
    - ✓ Message: WEP, WPA, WPA2, WPA3
- BluetoothSwitchStatus ᛒ
  - Description: Switch Bluetooth
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- BluetoothID

- Description: Bluetooth ID
- Data Type: string
- Type: Request
  - ✓ Message: BT ID
- Data Type: string
- Type: Response
  - ✓ Message: BT ID

- BluetoothPassword
  - Description: Bluetooth Password
  - Data Type: string
  - Type: Request
    - ✓ Message: BT Password
- BluetoothMacAddress
  - Description: Bluetooth Mac Address
  - Data Type: uint8[6]
  - Type: Request
    - ✓ Message: Mac Address
  - Data Type: uint8[6]
  - Type: Response
    - ✓ Message: Mac Address
- BackgroundServiceStatus
  - Description: Background Service Status
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Off, On, Error
- BackgroundServiceSettings
  - Description: Background Service Status
  - Data Type: uint16
  - Type: Request
    - ✓ Message: Settings, Default
  - Data Type: BOOLEAN
  - Type: Response

- ✓ Message: Off, On, Error

9.12 IMS Domain

IMS Driver Monitoring System is a technology that monitors the driver's state in real-time using cameras and sensors, enhancing driving safety by assessing driver concentration and condition.

The IMS Command message is used to control some functions and indicate the IMS status.

- DriverMonitoringSystem
  - Description: Driver Monitoring System On/Off
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Inactive, Active, Standby, Error, Failed
- DriverStatus
  - Description: Driver Status
  - Data Type: unit8
  - Type: Response
    - ✓ Message: Concentration(0), Ohers(Non-concentration)
- OccupantSensingSystem
  - Description: Occupant Sensing System On/Off
  - Data Type: BOOLEAN
  - Type: Request
    - ✓ Message: Off, On
  - Data Type: enum
  - Type: Response
    - ✓ Message: Inactive, Active, Standby, Error, Failed

9.13 Window Commands

Passengers can easily open and close car windows using buttons or voice commands. This feature allows for ventilation, changes in visibility, and communication with the outside, providing a more comfortable driving and riding experience.

The Windows Command message is used to control some functions and indicate the windows status.

- DoorWindowFL
  - Description: Door Window (FL)
  - Data Type: unit8
  - Type: Request
    - ✓ Message: Up, Down, Stop, Default Position
- DoorWindowFL
  - Description: Door Window Position (FL)
  - Data Type: unit8
  - Type: Request
    - ✓ Message: Window Position(0~100)
  - Data Type: unit8
  - Type: Response
    - ✓ Message: Window Position(0~100), Error=255
- DoorWindowFR
  - Description: Door Window (FR)
  - Data Type: unit8
  - Type: Request
    - ✓ Message: Up, Down, Stop, Default Position
- DoorWindowFR
  - Description: Door Window Position (FR)
  - Data Type: unit8
  - Type: Request
    - ✓ Message: Window Position(0~100)
  - Data Type: unit8
  - Type: Response
    - ✓ Message: Window Position(0~100), Error=255
- DoorWindowRL
  - Description: Door Window (RL)
  - Data Type: unit8
  - Type: Request
    - ✓ Message: Up, Down, Stop, Default Position
- DoorWindowRL
  - Description: Door Window Position (RL)

- Data Type: unit8
- Type: Request
  - ✓ Message: Window Position(0~100)
- Data Type: unit8
- Type: Response
  - ✓ Message: Window Position(0~100), Error=255

- DoorWindowRR
  - Description: Door Window (RR)
  - Data Type: unit8
  - Type: Request
    - ✓ Message: Up, Down, Stop, Default Position
- DoorWindowRR
  - Description: Door Window Position (RR)
  - Data Type: unit8
  - Type: Request
    - ✓ Message: Window Position(0~100)
  - Data Type: unit8
  - Type: Response
    - ✓ Message: Window Position(0~100), Error=255

10.0 References

〔1〕 MIH and partner Internal materials

〔2〕 UN Regulation No. 46 Devices for indirect vision and motor vehicles with regard to the installation of these devices

〔3〕 Innovation : The Software Defined Vehicle, IBM https://www.ibm.com/blogs/digitale-perspektive/2023/06/the-software-defined-vehicle

〔4〕 Possibilities of Boosting Future E/E Architecture Development, Bosch Engineering 10/13/2021