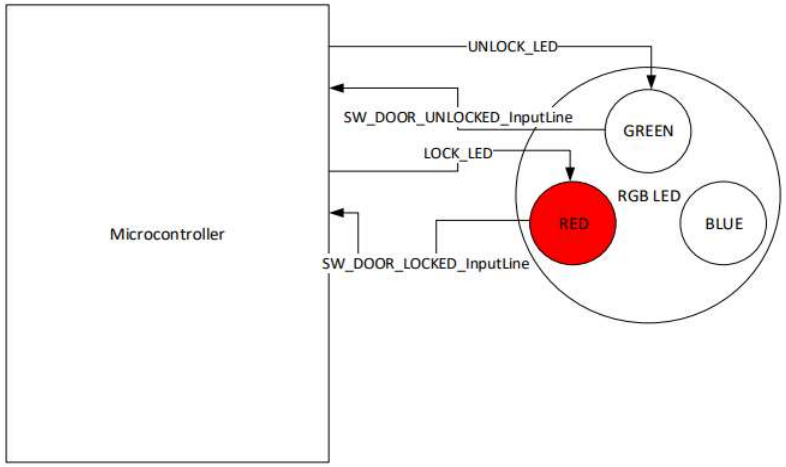
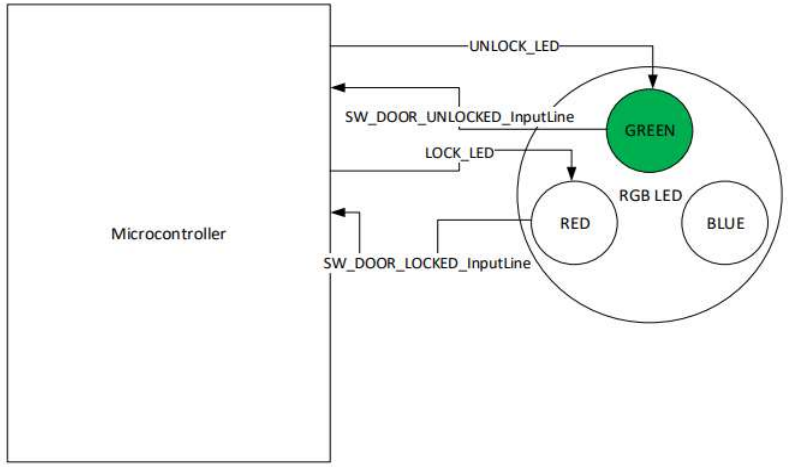
Door Locking

Door Locking function is used to control the corresponding Door **Solenoid** that is used to lock and unlock the corresponding Door.
Hardware - Software Requirements
Solenoid Behavior will be model with and RGB LED that will be used to indicate when the Door is **LOCK** or **UNLOCK**. When RGB become **Green** (0,100,0), then it shall indicate Door is **UNLOCK**. When RGB become **Red** (100,0,0), then it shall indicate Door is **LOCK**. When RGB become any other value different from Green or Red, then it shall be considered invalid.
Door Locking Configuration
The Data Identifier $0180 (DID_0180) has the purpose to configurate the duration of the LED on time for Lock and Unlock actuations for Door Locking functionality. DID_0180. UnlockBlinkTime resolution is 10 ms. DID_0180. UnlockBlinkTime default value is 10 (100ms) DID_0180. LockBlinkTime resolution is 10 ms. DID_0180. LockBlinkTime default value is 10 (100ms)
Software Requirements
LOCK_DOOR_ACTUATION
LOCK_DOOR_ ACTUATION function is executed when the operation to lock the door is executed. The sequence for **LOCK_DOOR_ ACTUATION** shall be first to turn off **UNLOCK_LED**, then to hold states during 100 milliseconds, and finally to turn on **LOCK_LED**.
UNLOCK_DOOR_ACTUATION
UNLOCK_DOOR_ ACTUATION function is executed when the operation to unlock the door is executed. The sequence for **LOCK_DOOR_ ACTUATION** shall be first to turn off **LOCK_LED**, then to hold states during 100 milliseconds, and finally to turn on **UNLOCK_LED**.





Manual Mode

Driver Door and **Passenger Door** are allowed to execute **Door Locking Manual Mode.**
LOCK_DOOR_ACTUATION
Driver Door
If **LOCK_BTN** of the **Driver Door** DCU (CAN frame DCU_1) transitions from **BTN_NOT_PRESSED** to **BTN_PRESSED**, then **LOCK_DOOR_ ACTUATION** shall be executed on **Driver Door.**
If **LOCK_BTN** and **UNLOCK_BTN** of the **Driver Door** DCU (CAN frame DCU_1) are **pressed**, then **LOCK_DOOR_ ACTUATION** shall not be executed on **Driver Door.**
If **LOCK_BTN** and **UNLOCK_BTN** of the **Driver Door** DCU (CAN frame DCU_1) are **not pressed**, then **LOCK_DOOR_ ACTUATION** shall not be executed on **Driver Door.**
Passenger Door
If **LOCK_BTN** of the **Passenger Door** DCU (CAN frame DCU_2) transitions from **BTN_NOT_PRESSED** to **BTN_PRESSED**, then **LOCK_DOOR_ ACTUATION** shall be executed on **Passenger Door.**
If **LOCK_BTN** and **UNLOCK_BTN** of the **Passenger Door** DCU (CAN frame DCU_2) are **pressed**, then **LOCK_DOOR_ ACTUATION** shall not be executed on **Passenger Door.**
If **LOCK_BTN** and **UNLOCK_BTN** of the **Passenger Door** DCU (CAN frame DCU_2) are **not pressed**, then **LOCK_DOOR_ ACTUATION** shall not be executed on **Passenger Door.**
UNLOCK_DOOR_ACTUATION
Driver Door
If **UNLOCK_BTN** of the **Driver Door** DCU (CAN frame DCU_1) transitions from **BTN_NOT_PRESSED** to **BTN_PRESSED**, then **UNLOCK_DOOR_ ACTUATION** shall be executed on **Driver Door.**
If **LOCK_BTN** and **UNLOCK_BTN** of the **Driver Door** DCU (CAN frame DCU_1) are **pressed**, then **UNLOCK_DOOR_ ACTUATION** shall not be executed on **Driver Door.**
If **LOCK_BTN** and **UNLOCK_BTN** of the **Driver Door** DCU (CAN frame DCU_1) are **not pressed**, then **UNLOCK_DOOR_ ACTUATION** shall not be executed on **Driver Door.**
Passenger Door
If **UNLOCK_BTN** of the **Passenger Door** DCU (CAN frame DCU_2) transitions from **BTN_NOT_PRESSED** to **BTN_PRESSED**, then **UNLOCK_DOOR_ ACTUATION** shall be executed on **Passenger Door.**
If **LOCK_BTN** and **UNLOCK_BTN** of the **Passenger Door** DCU (CAN frame DCU_2) are **pressed**, then **UNLOCK_DOOR_ ACTUATION** shall not be executed on **Passenger Door.**
If **LOCK_BTN** and **UNLOCK_BTN** of the **Passenger Door** DCU (CAN frame DCU_2) are **not pressed**, then **UNLOCK_DOOR_ ACTUATION** shall not be executed on **Passenger Door.**

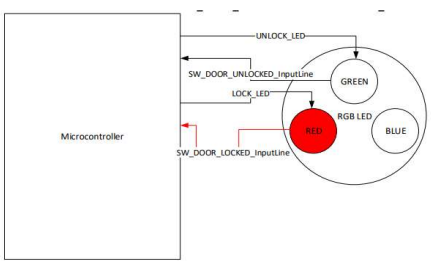
Remote Operation

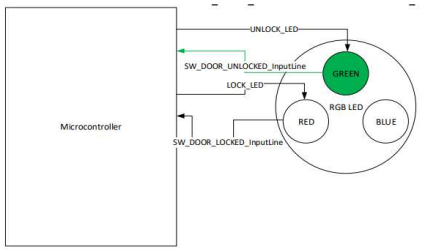
If the message for **Remote Operation** from **BCM** request via **CAN network** is authenticated from the corresponding source, then **All Doors** shall execute Door Locking.
Depending on the **Door Locking** operations shall be executed from requests from **Driver Door** or **Passenger Door**.
LockingReq is a signal that reports to the network the Lock or Unlock request to Body Control Module.
NO_LOCKING_REQ is the value that indicates that there is no Lock or Unlock command request.
LOCK_REQ is the value that indicates that user has requested a LOCK request operation.
UNLOCK_REQ is the value that indicates that user has requested a UNLOCK request operation.
ConfortCmd is a signal that contains the Confort Command, and it represents the Confort Operation for the vehicle.
NoCmd represents No Command to execute.
LockCmd represents Lock Command.
UnlockAllCmd represents Unlock Command for all Doors.
UnlockDrvrCmd represents Unlock Command for Driver Door only.

LOCK_DOOR_ACTUATION
Driver Door
If **DCU_2.LockingReq** (CAN frame DCU_2 Byte 1) signal transitions from **NO_LOCKING_REQ** value (0x00) to **LOCK_REQ** value (0x01), then **Driver Door** shall execute **LOCK_DOOR_ ACTUATION.**
Passenger Door
If signal **DCU_1.LockingReq** (CAN frame DCU_1 Byte 1) transitions from **NO_LOCKING_REQ** value (0x00) to **LOCK_REQ** value (0x01), then **Passenger Door** shall execute **LOCK_DOOR_ ACTUATION.**
RearLeft Door
If signal **DCU_1.LockingReq** (CAN frame DCU_1 Byte 1) transitions from **NO_LOCKING_REQ** value (0x00) to **LOCK_REQ** value (0x01), then **RearLeft Door** shall execute **LOCK_DOOR_ ACTUATION.**
If signal **DCU_2.LockingReq** (CAN frame DCU_2 Byte 1) transitions from **NO_LOCKING_REQ** value (0x00) to **LOCK_REQ** value (0x01), then **RearLeft Door** shall execute **LOCK_DOOR_ ACTUATION.**
RearRight Door
If signal **DCU_1.LockingReq** (CAN frame DCU_1 Byte 1) transitions from **NO_LOCKING_REQ** value (0x00) to **LOCK_REQ** value (0x01), then **RearRight Door** shall execute **LOCK_DOOR_ ACTUATION.**
When signal **DCU_2.LockingReq** (CAN frame DCU_1 Byte 1) transitions from **NO_LOCKING_REQ** value (0x00) to **LOCK_REQ** value (0x01), then **RearRight Door** shall execute **LOCK_DOOR_ ACTUATION.**
UNLOCK_DOOR_ACTUATION
Driver Door
If signal **DCU_2.LockingReq** (CAN frame DCU_2 Byte 1) transitions from **NO_LOCKING_REQ** value (0x00) to **UNLOCK_REQ** value (0x02), then **Driver Door** shall execute **UNLOCK_DOOR_ ACTUATION .**
If signal **BCM_2.ConfortCmd** (CAN frame BCM_2 Byte 2) transitions from **No Cmd** value (0x00) to **UnlockDrvrCmd** value (0x03), then **UNLOCK_DOOR_ ACTUATION** shall be executed only for **Driver Door**.
Passenger Door
If signal **DCU_1.LockingReq** (CAN frame DCU_1Byte 1) transitions from **NO_LOCKING_REQ** value (0x00) to **UNLOCK_REQ** value (0x02), then **Passenger Door** shall execute **UNLOCK_DOOR_ ACTUATION.**
RearLeft Door
If signal **DCU_1.LockingReq** (CAN frame DCU_1 Byte 1) transitions from **NO_LOCKING_REQ** value (0x00) to **UNLOCK_REQ** value (0x02), then **RearLeft Door** shall execute **UNLOCK_DOOR_ ACTUATION.**
If signal **DCU_2.LockingReq** (CAN frame DCU_2 Byte 1) transitions from **NO_LOCKING_REQ** value (0x00) to **UNLOCK_REQ** value (0x02), then **RearLeft Door** shall execute **UNLOCK_DOOR_ ACTUATION.**
RearRight Door
If signal **DCU_1.LockingReq** (CAN frame DCU_1 Byte 1) transitions from **NO_LOCKING_REQ** value (0x00) to **UNLOCK_REQ** value (0x02), then **RearRight Door** shall execute **UNLOCK_DOOR_ ACTUATION.**
If signal **DCU_2.LockingReq** (CAN frame DCU_2 Byte 1) transitions from **NO_LOCKING_REQ** value (0x00) to **UNLOCK_REQ** value (0x02), then **RearRight Door** shall execute **UNLOCK_DOOR_ ACTUATION.**

Door Lock Status

Door Lock Status Report
LOCK_DOOR_ACTUATION
If **LOCK_BTN** of the DCU transitions from **BTN_NOT_PRESSED** to **BTN_PRESSED**, then **Driver Door** shall transmit signal **DCU_1.LockingReq** (CAN frame DCU_1 Byte 1) with a value equal to 0x01 (LOCK_REQ) **only once**. After that, **Driver Door** shall set **DCU_1.LockingReq** (CAN frame DCU_1 Byte 1) back to 0x00 (NO_LOCKING_REQ) and keep it like this as long as there is no lock/unlock request from LOCK_BTN/UNLOCK_BTN. If **LOCK_BTN** of the DCU transitions from **BTN_NOT_PRESSED** to **BTN_PRESSED**, then **Passenger Door** shall transmit signal **DCU_2.LockingReq** (CAN frame DCU_2 Byte 1) with a value equal to 0x01 (LOCK_REQ) **only once**. After that, **Passenger Door** shall set **DCU_2.LockingReq** (CAN frame DCU_2 Byte 1) back to 0x00 (NO_LOCKING_REQ) and keep it like this as long as there is no lock/unlock request from LOCK_BTN/UNLOCK_BTN.
UNLOCK_DOOR_ACTUATION
If **UNLOCK_BTN** of the DCU transitions from **BTN_NOT_PRESSED** to **BTN_PRESSED**, then **Driver Door** shall transmit signal **DCU_1.LockingReq** (CAN frame DCU_1 Byte 1) with a value equal to 0x02 (UNLOCK_REQ) **only once**. After that, **Driver Door** shall set **DCU_1.LockingReq** (CAN frame DCU_1 Byte 1) back to 0x00 (NO_LOCKING_REQ) and keep it like this as long as there is no lock/unlock request from LOCK_BTN/UNLOCK_BTN. If **UNLOCK_BTN** of the DCU transitions from **BTN_NOT_PRESSED** to **BTN_PRESSED**, then **Passenger Door** shall transmit signal **DCU_2.LockingReq** (CAN frame DCU_2 Byte 1) with a value equal to 0x02 (UNLOCK_REQ) **only once**. After that, **Passenger Door** shall set **DCU_2.LockingReq** (CAN frame DCU_2 Byte 1) back to 0x00 (NO_LOCKING_REQ) and keep it like this as long as there is no lock/unlock request from LOCK_BTN/UNLOCK_BTN.
Door Lock Status Determination
DOOR_LOCKED
If **SW_DOOR_LOCKED** is determined as **SW_ACTIVE**, then Door shall be considered as **Locked**.
DOOR_UNLOCKED
If **SW_DOOR_UNLOCKED** is determined as **SW_ACTIVE**, then Door shall be considered as **Unlocked**.
DOOR_ERROR
If **SW_DOOR_LOCKED** and **SW_DOOR_UNLOCKED** are determined as **SW_ACTIVE** during 10 samples of 10 ms consecutively each, then **Door Lock** shall be considered as **DOOR_ERROR.** If **DOOR_LOCKED** or **DOOR_UNLOCKED** status has been determined without a **DOOR_ERROR** after **Door Locking** functionality is re-enable after a power cycle (HW Diagnostics), then **DOOR_ERROR** shall not be considered present.
HW Diagnostics
If Door Lock is considered as **DOOR_ERROR**, then Door Locking functionality shall be disable until next power cycle. (Transition OFF -> RUN). If Door is considered as **DOOR_ERROR**, the corresponding DTC for **Driver Door Position Error Detected** shall be set as DTC: 0xA00100. If Door is considered as **DOOR_ERROR**, the corresponding DTC for **Passenger Door Position Error Detected** shall be set as DTC: 0xA00200. If Door is considered as **DOOR_ERROR**, the corresponding DTC for **RearRight Door Position Error Detected** shall be set as DTC: 0xA00300. If Door is considered as **DOOR_ERROR**, the corresponding DTC for **RearLeft Door Position Error Detected** shall be set as DTC: 0xA00400. If **DOOR_ERROR** conditions are no longer detected, the corresponding DTC shall be clear.





Auto Lock While Driving

DoorLockSts is a signal that reports the Door Lock Status.
DOOR_LOCK is the value that indicates Door is currently Locked.
DOOR_UNLOCK is the value that indicates Door is currently Unlocked.
DOOR_UNKNOWN is the value that indicates Door is in an Unknown State.
ERROR is the value that indicates Door Position is on an Error State.
SysPwrMode is the signal that contains the System Power Mode that will be used for the ECU to know the Ignition Status of the system.
Run is the value that indicates Vehicle is in Run Mode.
If at least one of the DoorLockSts, from DCU_1, DCU_2, DCU_3 and DCU_4, (CAN frame DCU Byte 1) is equal to**DOOR_UNLOCK** value (0x01) and **Vehicle Speed** indicates equal or greater than 20km/h and **System Power Mode** (CAN frame BCM_5 Byte 3) is equal to **Run** value (0x03), then **Driver Door** shall report command **LockingReq** (CAN frame DCU_1 Byte 1) as **LOCK_REQ** value (0x01).
Auto Lock While Driving request will be received by **BCM** and reported as **ConfortCmd** (CAN frame BCM_2 Byte 2) with value **LockCmd** (0x01).
Auto Lock While Driving sequence shall not take more than 500ms since **Vehicle Speed** and **System Power Mode** are met.

Frame BCM_5	SysPwrMode

Message Layout									
ID	BYTE 0	BYTE 1	BYTE 2	BYTE 3	BYTE 4	BYTE 5	BYTE 6	BYTE 7	PERIOD
0x110	BCM_5_MC	X	X	SysPwrMode	BCM_5_CMAC				500 ms

Signals Description			
BCM_5_MC	This is the Message Counter of the BCM_5 frame. This signal shall go between 0 and 255 with increments of 1. This signal shall be updated every time the telegram is transmitted. If the signal reaches its limit (255), then the counter value shall initialize as 0.		
SysPwrMode	This Signal contains the System Power Mode that will be used for the ECU to know the Ignition Status of the system.		
	0x00	(SNA)	Signal Not Available.
	0x01	(OFF)	Vehicle is OFF.
	0x02	(ACC)	Vehicle is in Accessory.
	0x03	(RUN)	Vehicle is in Run Mode.
	0x04	(CRANK)	Vehicle is doing Ignition.
	Values different than this shall be considered as INVALID Data.		
BCM_5_CMAC:	This is a 32 bytes CMAC used to authenticate the source of this message. If the CMAC is not valid then the content of the whole message shall be ignored.		

Frame BCM_2

Message Layout									
ID	**BYTE 0**	**BYTE 1**	**BYTE 2**	**BYTE 3**	**BYTE 4**	**BYTE 5**	**BYTE 6**	**BYTE 7**	**PERIOD**
0x252	**BCM_2_MC**	**X**	**ConfortCmd**	**X**	**BCM_2_CMAC**				**500 ms**

Signals Description			
BCM_2_MC	This is theMessage Counter of the BCM_frame. This signal shall go between 0 and 255 with increments of 1. This signal shall be updated every time the telegram is transmitted. If the signal reaches its limit (255), then the counter value shall initialize as 0.		
ConfortCmd	This Signal contains the Confort Command and it represents the confort operation for the vehicle.		
	0x00	(No Cmd)	No command to excuse
	0x01	(LockCmd)	Represents Lock Command
	0x02	(UnlockAllCmd)	Represents Unlock Command for all Doors.
	0x03	(UnlockDrvrCmd)	Represents Unlock Command for Driver Door only.
	Values different than this shall be considered as INVALID Data.		
BCM_2_CMAC:	This is a 32 bytes CMAC used to aauthenticat the source of this message. If the CMAC is not valid then the content of the whole message shall be ignored.		

Frame DCU_1

Message Layout									
ID	**BYTE 0**	**BYTE 1**	**BYTE 2**	**BYTE 3**	**BYTE 4**	**BYTE 5**	**BYTE 6**	**BYTE 7**	**PERIOD**
0x201	**WindowPos**	**LockingReq**	**WindowOp**	**RearWindowLock**	**DoorLockSts**	**WindowControl**	**DCU_1_MC**	**DCU_1_CRC**	**100 ms**

Signals Description			
WindowPos:	This Signal reports to the network the Driver Window Position.		
	0x00	(IN_BETWEEN)	Window is in between, this means the window is not completely OPEN neither completely CLOSE.
	0x01	(COMPLETELY_OPEN)	Window is Completely Open.
	0x02	(COMPLETELY_CLOSE)	Window is Completely Close.
	0x03	(ERROR)	Window is on an ERROR state.
	Values different than this shall be considered as INVALID Data.		
LockingReq:	This signal reports to the network the Lock or Unlock Request to Body Control Module.		
	0x00	(NO_LOCKING_REQ)	There is no Lock or Unlock command requested.
	0x01	(LOCK_REQ)	User has request a LOCK request operation.
	0x02	(UNLOCK_REQ)	User has request a UNLOCK request operation.
	Values different than this shall be considered as INVALID Data.		
WindowOp:	This signal reports the windows current Operation.		
	0x00	(WINDOW_IDLE)	Window is not moving.
	0x01	(WINDOW_UP)	Window is doing a Close Operation.
	0x02	(WINDOW_DOWN)	Window is doing a Open Operation.
	Values different than this shall be considered as INVALID Data.		
RearWindowLock:	This signal reports the status to block the Window Control operation for Rear Windows.		
	0x00	(REAR_WINDOW_UNBLOCK)	Rear Windows are allowed to operate.
	0x01	(REAR_WINDOW_BLOCK)	Rear Windows shall not operate.
	Values different than this shall be considered as INVALID Data.		
DoorLockSts:	This signal reports the Door Lock Status.		
	0x00	(DOOR_LOCK)	Door is currently Locked.
	0x01	(DOOR_UNLOCK)	Door is currently Unlocked.
	0x02	(DOOR_UNKNOWN)	Door is in an Unknown State.
	0x03	(ERROR)	Door Position is on an ERROR state.
	Values different than this shall be considered as INVALID Data.		
WindowControl:	This signal is used to control other Door Control Modules on the network.		

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X	X	WindowControl_Passenger		WindowControl_RearLeft		WindowControl_RearRight	

Commands:		
0x00	(WINDOW_NO_REQ)	No Window Request.
0x01	(WINDOW_UP_REQ)	Indicated Window Close.
0x02	(WINDOW_DOWN_REQ)	Indicated Window Open.
Values different than this shall be considered as INVALID Data.		

DCU_1_MC:	This is the Message counter of the DCU_1 frame. This signal shall go between 0 and 255 with increments of 1. This signal shall be updated every time the telegram is transmitted. If the signal reaches its limit (255), then the counter value shall initialize as 0.
DCU_1_CRC:	This is the Cyclic Redundancy Check for DCU_1 frame. This signal shall be calculated with an CRC8 algorithm from Byte0 to Byte5.

Frame DCU_2

Message Layout									
ID	BYTE 0	BYTE 1	BYTE 2	BYTE 3	BYTE 4	BYTE 5	BYTE 6	BYTE 7	PERIOD
0x202	WindowPos	LockingReq	WindowOp	RESERVED	DoorLockSts	RESERVED	DCU_2_MC	DCU_2_CRC	100 ms

Signals Description			
WindowPos:	This Signal reports to the network the Passenger Window Position.		
	0x00	(IN_BETWEEN)	Window is in between, this means the window is not completely OPEN neither completely CLOSE.
	0x01	(COMPLETELY_OPEN)	Window is Completely Open.
	0x02	(COMPLETELY_CLOSE)	Window is Completely Close.
	0x03	(ERROR)	Window is on an ERROR state.
	Values different than this shall be considered as INVALID Data.		
LockingReq:	This signal reports to the network the Lock or Unlock Request to Body Control Module.		
	0x00	(NO_LOCKING_REQ)	There is no Lock or Unlock command requested.
	0x01	(LOCK_REQ)	User has request a LOCK request operation.
	0x02	(UNLOCK_REQ)	User has request a UNLOCK request operation.
	Values different than this shall be considered as INVALID Data.		
WindowOp:	This signal reports the windows current Operation.		
	0x00	(WINDOW_IDLE)	Window is not moving.
	0x01	(WINDOW_UP)	Window is doing a Close Operation.
	0x02	(WINDOW_DOWN)	Window is doing a Open Operation.
	Values different than this shall be considered as INVALID Data.		
RESERVED	Reserved bytes shall be transmitted ad 0xFF.		
DoorLockSts:	This signal reports the Door Lock Status.		
	0x00	(DOOR_LOCK)	Door is currently Locked.
	0x01	(DOOR_UNLOCK)	Door is currently Unlocked.
	0x02	(DOOR_UNKNOWN)	Door is in an Unknown State.
	0x03	(ERROR)	Door Position is on an ERROR state.
	Values different than this shall be considered as INVALID Data.		
RESERVED	Reserved bytes shall be transmitted ad 0xFF.		
DCU_2_MC:	This is the Message counter of the DCU_2 frame. This signal shall go between 0 and 255 with increments of 1. This signal shall be updated every time the telegram is transmitted. If the signal reaches its limit (255), then the counter value shall initialize as 0.		
DCU_2_CRC:	This is the Cyclic Redundancy Check for DCU_2 fame. This signal shall be calculated with an CRC8 algorithm from Byte0 to Byte5.		

Data Identifier Lists

DID $0180:	Door Locking Configuration		
This DID has the purpose to configurate the duration of the LED on time for Lock and Unlock actuations for Door Locking functionality. This DID shall support service Read DID $22. This DID shall support service Write DID $2E. Write DID operation shall be protected by Extended session. Write DID operation shall be protected by security level $01.			
ID	**Byte 0**	**Byte 1**	**Byte 2**
0x0180	UnlockBlinkTime		LockBlinkTime
DID_0180.UnlockBlinkTime:			
	Resolution:	**10 ms**	
	Default value:	**10 (100 ms)**	
DID_0180.LockBlinkTime:			
	Resolution:	**10 ms**	
	Default value:	**10 (100 ms)**	