Window Control

Window Control function is used to control the corresponding Window DC motor that is used to Open and Close the corresponding Window.

Hardware - Software Requirements

Window behavior will be model with a 10 RED LED BAR that will be used to indicate window position and window operations.
Window behavior will be controlled using 10 digital outputs.
Window Actuation will be displayed as an animation with 500ms delay between transitions on the LED arrangement.
Window Animation to open the Window from CompletelyClose to CompletelyOpen is demonstrated when the LED arrangement transitions from **ON** to **OFF** in descendant order, from LED 10 to LED 1.
Window Animation to close the Window from CompletelyOpen to CompletelyClose is demonstrated when the LED arrangement transitions from OFF to ON in ascendant order, from LED 1 to LED 10.

Software Requirements

WindowOp is a signal that reports the Windows current Operation.
WINDOW_IDLE is the value that reports that the Window is not moving.
If there is not a **Window Control** Actuation on the **Driver Door,** then ECU shall report on the CAN frame **DCU_1.WindowOp** (CAN frame DCU_1 Byte 2) as **WINDOW_IDLE** value (0x00) .
If there is not a **Window Control** Actuation on the **Passenger Door,** then ECU shall report on the CAN frame **DCU_2.WindowOp** (CAN frame DCU_2 Byte 2) as **WINDOW_IDLE** value (0x00) .
If there is not a **Window Control** Actuation on the **RearLeft Door,** then ECU shall report on the CAN frame **DCU_3.WindowOp** (CAN frame DCU_3 Byte 2) as **WINDOW_IDLE** value (0x00) .
If there is not a **Window Control** Actuation on the **RearRight Door,** then ECU shall report on the CAN frame **DCU_4.WindowOp** (CAN frame DCU_4 Byte 2) as **WINDOW_IDLE** value (0x00) .

OPEN_WINDOW_ACTUATION

OPEN_WINDOW_ ACTUATION function is executed when the LED arrangement transitions from **ON** to **OFF** in descendant order, from LED 10 to LED 1.
Between every transition of the LED arrangement from ON to OFF, there shall be a 500ms delay.

Driver Door

If Driver Door **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is different than **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is different than **ERROR** value (0x03), then **OPEN_WINDOW_ACTUATION** function shall be executed.
If there is an **OPEN_WINDOW_ACTUATION** on the **Driver Door,** then ECU shall report on the CAN frame **DCU_1.WindowOp** (CAN frame DCU_1 BYTE 2) as **WINDOW_DOWN** value (0x02).
If Driver Door **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is considered as **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is considered as **ERROR** value (0x03), then **OPEN_WINDOW_ACTUATION** function shall stop.

Passenger Door

If Passenger Door **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is different than **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is different than **ERROR** value (0x03), then **OPEN_WINDOW_ACTUATION** function shall be executed.
If there is an **OPEN_WINDOW_ACTUATION** on the **Passenger Door,** then ECU shall report on the CAN frame **DCU_2.WindowOp** (CAN frame DCU_2 BYTE 2) as **WINDOW_DOWN** value (0x02).
If Passenger Door **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is considered as **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is considered as **ERROR** value (0x03), then **OPEN_WINDOW_ACTUATION** function shall stop.

RearLeft Door

If RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is different than **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is different than **ERROR** value (0x03), then **OPEN_WINDOW_ACTUATION** function shall be executed.
If there is an **OPEN_WINDOW_ACTUATION** on the **RearLeft Door,** then ECU shall report on the CAN frame **DCU_3.WindowOp** (CAN frame DCU_3 BYTE 2) as **WINDOW_DOWN** value (0x002).
If RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is considered as **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is considered as **ERROR** value (0x03), then **OPEN_WINDOW_ACTUATION** function shall stop.

RearRight Door

If RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is different than **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is different than **ERROR** value (0x03), then **OPEN_WINDOW_ACTUATION** function shall be executed.
If there is an **OPEN_WINDOW_ACTUATION** on the **RearRight Door,** then ECU shall report on the CAN frame **DCU_4.WindowOp** (CAN frame DCU_4 BYTE 2) as **WINDOW_DOWN** value (0x02).
If RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is considered as **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is considered as **ERROR** value (0x03), then **OPEN_WINDOW_ACTUATION** function shall stop.

GLOBAL_OPEN_WINDOW_ACTUATION

GLOBAL_OPEN_WINDOW_ACTUATION function is executed when the LED arrangement transitions from **ON** to **OFF** in descendant order, from LED 10 to LED 1.
Between every transition of the LED arrangement from ON to OFF, there shall be a 500ms delay.

Driver Door

If Driver Door **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is different than **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is different than **ERROR** value (0x03), then **GLOBAL_OPEN_WINDOW_ACTUATION** function shall be executed.
If there is a **GLOBAL_OPEN_WINDOW_ACTUATION** on the **Driver Door,** then ECU shall report on the CAN frame **DCU_1.WindowOp** (CAN frame DCU_1 BYTE 2) as **WINDOW_DOWN** value (0x02).
If Driver Door **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is considered as **COMPLETELY_OPEN** value (0x01), then **GLOBAL_OPEN_WINDOW_ACTUATION** function shall stop.

Passenger Door

If Passenger Door **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is different than **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is different than **ERROR** value (0x03), then **GLOBAL_OPEN_WINDOW_ACTUATION** function shall be executed.
If there is a **GLOBAL_OPEN_WINDOW_ACTUATION** on the **Passenger Door,** then ECU shall report on the CAN frame **DCU_2.WindowOp** (CAN frame DCU_2 BYTE 2) as **WINDOW_DOWN** value (0x02).
If Passenger Door **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is considered as **COMPLETELY_OPEN** value (0x01), then **GLOBAL_OPEN_WINDOW_ACTUATION** function shall stop.

RearLeft Door

If RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is different than **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is different than **ERROR** value (0x03), then **GLOBAL_OPEN_WINDOW_ACTUATION** function shall be executed.
If there is a **GLOBAL_OPEN_WINDOW_ACTUATION** on the **RearLeft Door,** then ECU shall report on the CAN frame **DCU_3.WindowOp** (CAN frame DCU_3 BYTE 2) as **WINDOW_DOWN** value (0x02).
If RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is considered as **COMPLETELY_OPEN** value (0x01), then **GLOBAL_OPEN_WINDOW_ACTUATION** function shall stop.

RearRight Door

If RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is different than **COMPLETELY_OPEN** value (0x01) and **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is different than **ERROR** value (0x03), then **GLOBAL_OPEN_WINDOW_ACTUATION** function shall be executed.
If there is a **GLOBAL_OPEN_WINDOW_ACTUATION** on the **RearRight Door,** then ECU shall report on the CAN frame **DCU_4.WindowOp** (CAN frame DCU_4 BYTE 2) as **WINDOW_DOWN** value (0x02).
If RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is considered as **COMPLETELY_OPEN** value (0x01), then **GLOBAL_OPEN_WINDOW_ACTUATION** function shall stop.

CLOSE_WINDOW_ACTUATION

CLOSE_WINDOW_ACTUATION function is executed when the LED arrangement transitions from **OFF** to **ON** in ascendant order, from LED 1 to LED 10.
Between every transition of the LED arrangement from ON to OFF, there shall be a 500ms delay.

Driver Door

If Driver Door **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is different than **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is different than **ERROR** value (0x03), then **CLOSE_WINDOW_ACTUATION** function shall be executed.
If there is a **CLOSE_WINDOW_ACTUATION** on the **Driver Door,** then ECU shall report on the CAN frame **DCU_1.WindowOp** (CAN frame DCU_1 BYTE 2) as **WINDOW_UP** value (0x01).
If Driver Door **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is considered as **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is considered as **ERROR** value (0x03), then **CLOSE_WINDOW_ACTUATION** function shall stop.

Passenger Door

If Passenger Door **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is different than **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is different than **ERROR** value (0x03), then **CLOSE_WINDOW_ACTUATION** function shall be executed.
If there is a **CLOSE_WINDOW_ACTUATION** on the **Passenger Door,** then ECU shall report on the CAN frame **DCU_2.WindowOp** (CAN frame DCU_2 BYTE 2) as **WINDOW_UP** value (0x01).
If Passenger Door **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is considered as **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is considered as **ERROR** value (0x03), then **CLOSE_WINDOW_ACTUATION** function shall stop.

RearLeft Door

If RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is different than **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is different than **ERROR** value (0x03), then **CLOSE_WINDOW_ACTUATION** function shall be executed.
If there is a **CLOSE_WINDOW_ACTUATION** on the **RearLeft Door,** then ECU shall report on the CAN frame **DCU_3.WindowOp** (CAN frame DCU_3 BYTE 2) as **WINDOW_UP** value (0x01).
If RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is considered as **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is considered as **ERROR** value (0x03), then **CLOSE_WINDOW_ACTUATION** function shall stop.

RearRight Door

If RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is different than **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is different than **ERROR** value (0x03), then **CLOSE_WINDOW_ACTUATION** function shall be executed.
If there is a **CLOSE_WINDOW_ACTUATION** on the **RearRight Door,** then ECU shall report on the CAN frame **DCU_4.WindowOp** (CAN frame DCU_4 BYTE 2) as **WINDOW_UP** value (0x01).
If RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is considered as **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is considered as **ERROR** value (0x03), then **CLOSE_WINDOW_ACTUATION** function shall stop.

GLOBAL_CLOSE_WINDOW_ACTUATION

GLOBAL_CLOSE_WINDOW_ACTUATION function is executed when the LED arrangement transitions from **OFF** to **ON** in ascendant order, from LED 1 to LED 10.
Between every transition of the LED arrangement from ON to OFF, there shall be a 500ms delay.

Driver Door

If Driver Door **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is different than **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is different than **ERROR** value (0x03), then **GLOBAL_CLOSE_WINDOW_ACTUATION** function shall be executed.
If there is a **GLOBAL_CLOSE_WINDOW_ACTUATION** on the **Driver Door,** then ECU shall report on the CAN frame **DCU_1.WindowOp** (CAN frame DCU_1 BYTE 2) as **WINDOW_UP** value (0x01).
If Driver Door **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is considered as **COMPLETELY_CLOSE** value (0x02), then **GLOBAL_CLOSE_WINDOW_ACTUATION** function shall stop.

Passenger Door

If Passenger Door **WINDOW_POSITION** (frame DCU_2 Byte 0) is different than **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is different than **ERROR** value (0x03), then **GLOBAL_CLOSE_WINDOW_ACTUATION** function shall be executed.
If there is a **GLOBAL_CLOSE_WINDOW_ACTUATION** on the **Passenger Door,** then ECU shall report on the CAN frame **DCU_2.WindowOp** (CAN frame DCU_2 BYTE 2) as **WINDOW_UP** value (0x01).
If Passenger Door **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is considered as **COMPLETELY_CLOSE** value (0x02), then **GLOBAL_CLOSE_WINDOW_ACTUATION** function shall stop.

RearLeft Door

If RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is different than **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is different than **ERROR** value (0x03), then **GLOBAL_CLOSE_WINDOW_ACTUATION** function shall be executed.
If there is a **GLOBAL_CLOSE_WINDOW_ACTUATION** on the **RearLeft Door,** then ECU shall report on the CAN frame **DCU_3.WindowOp** (CAN frame DCU_3 BYTE 2) as **WINDOW_UP** value (0x01).
If RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is considered as **COMPLETELY_CLOSE** value (0x02), then **GLOBAL_CLOSE_WINDOW_ACTUATION** function shall stop.

RearRight Door

If RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is different than **COMPLETELY_CLOSE** value (0x02) and **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is different than **ERROR** value (0x03), then **GLOBAL_CLOSE_WINDOW_ACTUATION** function shall be executed.
If there is a **GLOBAL_CLOSE_WINDOW_ACTUATION** on the **RearRight Door,** then ECU shall report on the CAN frame **DCU_4.WindowOp** (CAN frame DCU_4 BYTE 2) as **WINDOW_UP** value (0x01).
If RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is considered as **COMPLETELY_CLOSE** value (0x02), then **GLOBAL_CLOSE_WINDOW_ACTUATION** function shall stop.

CANCEL_WINDOW_ACTUATION

CANCEL_WINDOW_ACTUATION function is executed when all the LED stop transitioning from ON to OFF or vice versa, showing the current position of the Window.

Driver Door

Only when there is a Window Control Actuation On Going on the Driver Door, then **CANCEL_WINDOW_ACTUATION** can be executed to abort current actuation.
If there is a **CANCEL_WINDOW_ACTUATION** on the **Driver Door,** then ECU shall report on the CAN frame **DCU_1.WindowOp** (CAN frame DCU_1 BYTE 2) as **WINDOW_IDLE** value (0x00).

Passenger Door

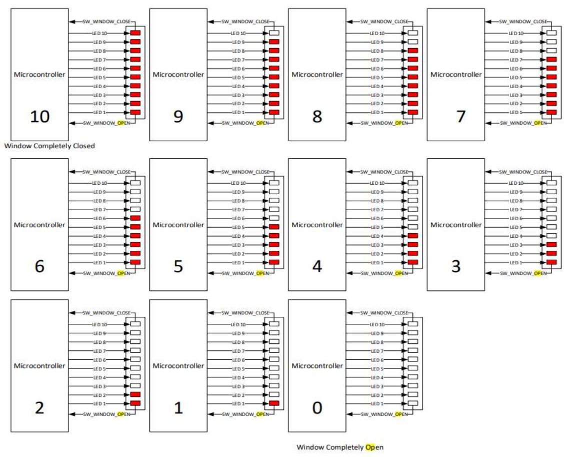
Only when there is a Window Control Actuation On Going on the Passenger Door, then **CANCEL_WINDOW_ACTUATION** can be executed to abort current actuation.
If there is a **CANCEL_WINDOW_ACTUATION** n the **Passenger Door,** then ECU shall report on the CAN frame **DCU_2.WindowOp** (CAN frame DCU_2 BYTE 2) as **WINDOW_IDLE** value (0x00).

RearLeft Door

Only when there is a Window Control Actuation On Going on the RearLeft Door, then **CANCEL_WINDOW_ACTUATION** can be executed to abort current actuation.
If there is a **CANCEL_WINDOW_ACTUATION** on the **RearLeft Door,** then ECU shall report on the CAN frame **DCU_3.WindowOp** (CAN frame DCU_3 BYTE 2) as **WINDOW_IDLE** value (0x00).

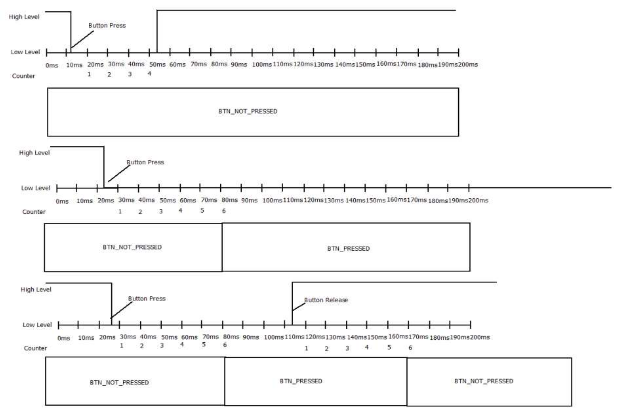
RearRight Door

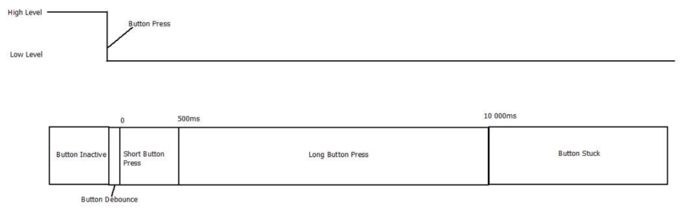
Only when there is a Window Control Actuation On Going on the RearRight Door, then **CANCEL_WINDOW_ACTUATION** can be executed to abort current actuation.
If there is a **CANCEL_WINDOW_ACTUATION** on the **RearRight Door,** then ECU shall report on the CAN frame **DCU_4.WindowOp** (CAN frame DCU_4 BYTE 2) as **WINDOW_IDLE** value (0x00).



Manual Mode

All Doors are allowed to execute manual **Window Control** from its corresponding **OPEN_BTN** and **CLOSE_BTN** states.
WindowControl is signal used to control other Door Control Modules on the network.
WINDOW_NO_REQ is the value that indicates No Window Request.
WINDOW_UP_REQ is the value that indicates Window Close.
WINDOW_DOWN_REQ is the value that indicates Window Open.
RearWindowLock is a signal that reports the status to block the Window Control operation for Rear Windows.
REAR_WINDOW_UNBLOCK is the value that indicates that Rear Windows are allowed to operate.
REAR_WINDOW_BLOCK is the value that indicates that Rear Windows shall not operate.
Button Configuration
The Data Identifier $0111 (DID_0111) has the purpose to indicate the value for Stuck Button Detection.
DID_0111. StuckBtnCfg resolution is 10 ms.
DID_0111. StuckBtnCfg default value is 1000 (10 000ms)
DID_0111. LongBtnCfg resolution is 10 ms.
DID_0111. LongBtnCfg default value is 50 (500 ms)
Button HW Diagnostics
If Button is considered as **STUCK**, then that Button shall be ignored until next power cycle. (Transition OFF -> RUN).
Window Control Buttons Error
If Button is considered as **STUCK**, the corresponding DTC for **Driver Door – Window Control Buttons Error** shall be set as **DTC: 0x901100.**
If Button is considered as **STUCK**, the corresponding DTC for **Passenger Door – Window Control Buttons Error Detected** shall be set as **DTC: 0x902100.**
If Button is considered as **STUCK**, the corresponding DTC for **RearRight Door – Window Control Buttons Error Detected** shall be set as **DTC: 0x903100.**
If Button is considered as **STUCK**, the corresponding DTC for **RearLeft Door – Window Control Buttons Error Detected** shall be set as **DTC: 0x904100.**
Door Locking Buttons Error
If Button is considered as **STUCK**, the corresponding DTC for **Driver Door – Door Locking Buttons Error** shall be set as **DTC: 0x901200.**
If Button is considered as **STUCK**, the corresponding DTC for **Passenger Door – Door Locking Buttons Error** shall be set as DTC: **0x902200.**
If **BUTTONS** are no longer considered as Stuck, then the corresponding DTC shall be clear.
Short Button Press
If a Button transitions from **BTN_NOT_PRESSED** to **BTN_PRESSED** and **BTN_NOT_PRESSED** within a time **<= 500** milliseconds, then it shall be considered as a **SHORT_BTN_PRESS**.
OPEN_WINDOW_ACTUATION
If the **OPEN_BTN** state on **Driver Door** is equal to **SHORT_BTN_PRESS,** then **Driver Door** shall execute **OPEN_WINDOW_ACTUATION**.
If the **OPEN_BTN** state on **Passenger Door** is equal to **SHORT_BTN_PRESS,** then **Passenger Door** shall execute **OPEN_WINDOW_ACTUATION**.
If the **OPEN_BTN** state on **RearLeft Door** is equal to **SHORT_BTN_PRESS** and **DCU_1.RearWindowLock** (CAN frame DCU_1 Byte 3) is equal to **REAR_WINDOW_UNBLOCK** value (0x00), then **RearLeft Door** shall execute **OPEN_WINDOW_ACTUATION**.
If the **OPEN_BTN** state on **RearRight Door** is equal to **SHORT_BTN_PRESS** and **DCU_1.RearWindowLock** (CAN frame DCU_1 Byte 3) is equal to **REAR_WINDOW_UNBLOCK** value (0x00), then **RearRight Door** shall execute **OPEN_WINDOW_ACTUATION.**
CLOSE_WINDOW_ACTUATION
If the **CLOSE_BTN** state on **Driver Door** is equal to **SHORT_BTN_PRESS,** then **Driver Door** shall execute **CLOSE_WINDOW_ACTUATION**.
If the **CLOSE_BTN** state on **Passenger Door** is equal to **SHORT_BTN_PRESS,** then **Passenger Door** shall execute **CLOSE_WINDOW_ACTUATION**.
If the **CLOSE_BTN** state on **RearLeft Door** is equal to **SHORT_BTN_PRESS** and **DCU_1.RearWindowLock** (CAN frame DCU_1 Byte 3) is equal to **REAR_WINDOW_UNBLOCK** value (0x00), then **RearLeft Door** shall execute **CLOSE_WINDOW_ACTUATION**.
If the **CLOSE_BTN** state on **RearRight Door** is equal to **SHORT_BTN_PRESS** and **DCU_1.RearWindowLock** (CAN frame DCU_1 Byte 3) is equal to **REAR_WINDOW_UNBLOCK** value (0x00), then **RearRight Door** shall execute **CLOSE_WINDOW_ACTUATION**.
Long Button Press
If a Button transitions from **BTN_NOT_PRESSED** to **BTN_PRESSED** and **BTN_NOT_PRESSED** within a time **> 500** milliseconds and **<= 10 000** milliseconds, then it shall be considered as a **LONG_BTN_PRESS**.
GLOBAL_OPEN_WINDOW_ACTUATION
If the **OPEN_BTN** state on **Driver Door** is equal to **LONG_BTN_PRESS,** then **Driver Door** shall execute **GLOBAL_OPEN_WINDOW_ACTUATION**.
If the **OPEN_BTN** state on **Passenger Door** is equal to **LONG_BTN_PRESS,** then **Passenger Door** shall execute **GLOBAL_OPEN_WINDOW_ACTUATION**.
If the **OPEN_BTN** state on **RearLeft Door** is equal to **LONG_BTN_PRESS** and **DCU_1.RearWindowLock** (CAN frame DCU_1 Byte 3) is equal to **REAR_WINDOW_UNBLOCK** value (0x00), then **RearLeft Door** shall execute **GLOBAL_OPEN_WINDOW_ACTUATION**.
If the **OPEN_BTN** state on **RearRight Door** is equal to **LONG_BTN_PRESS** and **DCU_1.RearWindowLock** (CAN frame DCU_1 Byte 3) is equal to **REAR_WINDOW_UNBLOCK** value (0x00), then **RearRightDoor** shall execute **GLOBAL_OPEN_WINDOW_ACTUATION**.
GLOBAL_CLOSE_WINDOW_ACTUATION
If the **CLOSE_BTN** state on **Driver Door** is equal to **LONG_BTN_PRESS,** then **Driver Door** shall execute **GLOBAL_CLOSE_WINDOW_ACTUATION**.
If the **CLOSE_BTN** state on **Passenger Door** is equal to **LONG_BTN_PRESS,** then **Passenger Door** shall execute **GLOBAL_CLOSE_WINDOW_ACTUATION**.
If the **CLOSE_BTN** state on **RearLeft Door** is equal to **LONG_BTN_PRESS** and **DCU_1.RearWindowLock** (CAN frame DCU_1 Byte 3) is equal to **REAR_WINDOW_UNBLOCK** value (0x00), then **RearLeft Door** shall execute **GLOBAL_CLOSE_WINDOW_ACTUATION**.
If the **CLOSE_BTN** state on **RearRight Door** is equal to **LONG_BTN_PRESS** and **DCU_1.RearWindowLock** (CAN frame DCU_1 Byte 3) is equal to **REAR_WINDOW_UNBLOCK** value (0x00), then **RearRight Door** shall execute **GLOBAL_CLOSE_WINDOW_ACTUATION**.
Button Stuck
If a Button is equal to **BTN_PRESSED > 10 000** milliseconds, then it shall be considered as a **BTN_STUCK**.
Window Control Report
WINDOW_UP_REQ
If **PASSENGER_CLOSE_BTN** state is equal to **SHORT_BTN_PRESS** or **LONG_BTN_PRESS**, then **Driver Door** shall report **WINDOW_UP_REQ** value (0x01) on **WindowControl_Passenger** bits position (Bit 4 y Bit 5) on the CAN frame **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5).
If **REARLEFT_CLOSE_BTN** state is equal to **SHORT_BTN_PRESS** or **LONG_BTN_PRESS**, then **Driver Door** shall report **WINDOW_UP_REQ** value (0x01) on **WindowControl_RearLeft** bits position (Bit 2 y Bit 3) on the CAN frame **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5).
If **REARRIGHT_CLOSE_BTN** state is equal to **SHORT_BTN_PRESS** or **LONG_BTN_PRESS**, then **Driver Door** shall report **WINDOW_UP_REQ** value (0x01) on **WindowControl_RearRight** bits position (Bit 0 y Bit 1) on the CAN frame **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5).
WINDOW_DOWN_REQ
If **PASSENGER_OPEN_BTN** state is equal to **SHORT_BTN_PRESS** or **LONG_BTN_PRESS**, then **Driver Door** shall report **WINDOW_DOWN_REQ** value (0x02) on **WindowControl_Passenger** bits position (Bit 4 y Bit 5) on the CAN frame **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5).
If **REARLEFT_OPEN_BTN** state is equal to **SHORT_BTN_PRESS** or **LONG_BTN_PRESS**, then **Driver Door** shall report **WINDOW_DOWN_REQ** value (0x02) on **WindowControl_RearLeft** bits position (Bit 2 y Bit 3) on the CAN frame **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5).
If **REARRIGHT_OPEN_BTN** state is equal to **SHORT_BTN_PRESS** or **LONG_BTN_PRESS**, then **Driver Door** shall report **WINDOW_DOWN_REQ** value (0x02) on **WindowControl_RearRight** bits position (Bit 0 y Bit 1) on the CAN frame **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5).
Rear Window Lock Report
REAR_WINDOW_UNBLOCK
While **REAR_WINDOW_LOCK_BTN** state is equal to **BTN_NOT_PRESSED**, then **Driver Door** shall report **REAR_WINDOW_UNBLOCK** value (0x00) on **DCU_1.RearWindowLock** bits position (CAN frame DCU_1 Byte 3).
REAR_WINDOW_BLOCK
While **REAR_WINDOW_LOCK_BTN** state is equal to **BTN_PRESSED**, then **Driver Door** shall report **REAR_WINDOW_BLOCK** value (0x01) on **DCU_1.RearWindowLock** bits position (CAN frame DCU_1 Byte 3).





Remote Operation

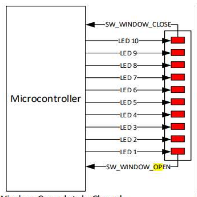
All Doors are allowed to execute Door Locking for Remote Operation from **BCM** request via **CAN network**.
Passenger Door is allowed to execute Door Locking for Remote Operation from **DCU 1** request via **CAN network**.
RearLeft Door is allowed to execute Door Locking for Remote Operation from **DCU 1** request via **CAN network**.
RearRight Door is allowed to execute Door Locking for Remote Operation from **DCU 1** request via **CAN network**.
OPEN_WINDOW_ACTUATION
If **BCM_2.ConfortCmd** signal (CAN frame BCM_2 Byte 2) is received with a **UnlockAllCmd** value (0x02) consecutively at least during 500 milliseconds, then **OPEN_WINDOW_ACTUATION** shall be executed.
If **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_DOWN_REQ** value (0x02) on **WindowControl_Passenger** bits position (Bit 4 y Bit 5), then **OPEN_WINDOW_ACTUATION** shall be executed on **Passenger Door**.
If **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_DOWN_REQ** value (0x02) on **WindowControl_RearLeft** bits position (Bit 2 y Bit 3), then **OPEN_WINDOW_ACTUATION** shall be executed on **RearLeft Door**.
If **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_DOWN_REQ** value (0x02) on **WindowControl_RearRight** bits position (Bit 0 y Bit 1), then **OPEN_WINDOW_ACTUATION** shall be executed on **RearRight Door**.
CLOSE_WINDOW_ACTUATION
If **BCM_2.ConfortCmd** signal (CAN frame BCM_2 Byte 2) is received with a **LockCmd** value (0x01) consecutively at least during 500 milliseconds, then **CLOSE_WINDOW_ACTUATION** shall be executed.
If DCU_1.WindowControl (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_UP_REQ** value (0x01) on **WindowControl_Passenger** bits position (Bit 4 y Bit 5), then **CLOSE_WINDOW_ACTUATION** shall be executed on **Passenger Door.**
If DCU_1.WindowControl (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_UP_REQ** value (0x01) on **WindowControl_RearLeft** bits position (Bit 2 y Bit 3), then **CLOSE_WINDOW_ACTUATION** shall be executed on **RearLeft Door.**
If DCU_1.WindowControl (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_UP_REQ** value (0x01) on **WindowControl_RearRight** bits position (Bit 0 y Bit 1), then **CLOSE_WINDOW_ACTUATION** shall be executed on **RearRight Door.**
CANCEL_WINDOW_ACTUATION
Driver Door
If **BCM_2.ConfortCmd** signal (CAN frame BCM_2 Byte 2) transitions from **LockCmd** value (0x01) to **No Cmd** value (0x00) or Driver Door **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is equal to **COMPLETELY_CLOSE** value (0x02) during a **CLOSE_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed.
If **BCM_2.ConfortCmd** signal (CAN frame BCM_2 Byte 2) transitions from **UnlockAllCmd** value (0x02) to **No Cmd** value (0x00) or Driver Door **WINDOW_POSITION** (CAN frame DCU_1 Byte 0) is equal to **COMPLETELY_OPEN** value (0x01) during an **OPEN_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed.
Passenger Door
If **BCM_2.ConfortCmd** signal (CAN frame BCM_2 Byte 2) transitions from **LockCmd** value (0x01) to **No Cmd** value (0x00) or Passenger Door **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is equal to **COMPLETELY_CLOSE** value (0x02) during a **CLOSE_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed.
If **BCM_2.ConfortCmd** signal (CAN frame BCM_2 Byte 2) transitions from **UnlockAllCmd** value (0x02) to **No Cmd** value (0x00) or Passenger Door **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is equal to **COMPLETELY_OPEN** value (0x01) during an **OPEN_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed.
If **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_NO_REQ** value (0x00) on **WindowControl_Passenger** bits position (Bit 4 y Bit 5) or Passenger Door **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is equal to **COMPLETELY_OPEN** value (0x01) during an **OPEN_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed on **Passenger Door**.
If **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_NO_REQ** value (0x00) on **WindowControl_Passenger** bits position (Bit 4 y Bit 5) or Passenger Door **WINDOW_POSITION** (CAN frame DCU_2 Byte 0) is equal to **COMPLETELY_CLOSE** value (0x02) during a **CLOSE_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed on **Passenger Door**.
RearLeft Door
If **BCM_2.ConfortCmd** signal (CAN frame BCM_2 Byte 2) transitions from **LockCmd** value (0x01) to **No Cmd** value (0x00) or RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is equal to **COMPLETELY_CLOSE** value (0x02) during a **CLOSE_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed.
If **BCM_2.ConfortCmd** signal (CAN frame BCM_2 Byte 2) transitions from **UnlockAllCmd** value (0x02) to **No Cmd** value (0x00) or RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is equal to **COMPLETELY_OPEN** value (0x01) during an **OPEN_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed.
If **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_NO_REQ** value (0x00) on **WindowControl_RearLeft** bits position (Bit 2 y Bit 3) or RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is equal to **COMPLETELY_OPEN** value (0x01) during an **OPEN_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed on **RearLeft Door**.
If **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_NO_REQ** value (0x00) on **WindowControl_RearLeft** bits position (Bit 2 y Bit 3) or RearLeft Door **WINDOW_POSITION** (CAN frame DCU_3 Byte 0) is equal to **COMPLETELY_CLOSE** value (0x02) during a **CLOSE_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed on **RearLeft Door**.
RearRight Door
If **BCM_2.ConfortCmd** signal (CAN frame BCM_2 Byte 2) transitions from **LockCmd** value (0x01) to **No Cmd** value (0x00) or RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is equal to **COMPLETELY_CLOSE** value (0x02) during a **CLOSE_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed.
If **BCM_2.ConfortCmd** signal (CAN frame BCM_2 Byte 2) transitions from **UnlockAllCmd** value (0x02) to **No Cmd** value (0x00) or RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is equal to **COMPLETELY_OPEN** value (0x01) during an **OPEN_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed.
If **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_NO_REQ** value (0x00) on **WindowControl_RearRight** bits position (Bit 0 y Bit 1) or RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is equal to **COMPLETELY_OPEN** value (0x01) during an **OPEN_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed on **RearRight Door**.
If **DCU_1.WindowControl** (CAN frame DCU_1 Byte 5) signal is received with a **WINDOW_NO_REQ** value (0x00) on **WindowControl_RearRight** bits position (Bit 0 y Bit 1) or RearRight Door **WINDOW_POSITION** (CAN frame DCU_4 Byte 0) is equal to **COMPLETELY_CLOSE** value (0x02) during a **CLOSE_WINDOW_ACTUATION**, then **CANCEL_WINDOW_ACTUATION** shall be executed on **RearRight Door**.

AntiPinch Operation

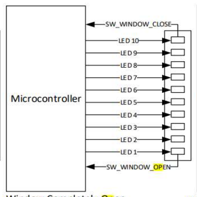
ANTIPINCH_SIGNAL is a digital Input on the system that reports when an Anti-pinch Event has occurred.
ANTIPINCH_SIGNAL has a dedicated instance per Door.
ANTIPINCH_SIGNAL will report the Anti-pinch Event using an Analog input 10 bits resolution.
ANTIPINCH_SIGNAL shall be detected if the Analog signal transitions from below the threshold to above the threshold. Use 820 ADC counts as threshold reference.
Only during **CLOSE_WINDOW_ACTUATION** or **GLOBAL_CLOSE_WINDOW_ACTUATION**, the **ANTIPINCH_SIGNAL** shall report the Anti-pinch Event.
If **ANTIPINCH_SIGNAL** is present, then an immediate **CANCEL_WINDOW_ACTUATION** shall be executed. Then a **GLOBAL_OPEN_ACTUATION** shall be executed.

Window Position

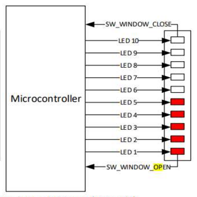
Window Position Report
WindowPos is a signal that reports to the network the Window Position.
IN_BETWEEN value (0x00) indicates that Window is in between, which means the Window is not completely OPEN neither Completely CLOSE.
COMPLETELY_OPEN value (0x01) indicates that Window is Completely OPEN.
COMPLETELY_CLOSE value (0x02) indicates that Window is Completely CLOSE.
ERROR value (0x03) indicates that Window is on an ERROR state.
All Doors are allowed to execute Door Locking for Remote Operation from **BCM** request via **CAN network**.
For **Driver Door**, it shall report the determined Lock Status Via **DCU_1.WindowPos** (CAN frame DCU_1 Byte 0).
For **Passenger Door**, it shall report the determined Lock Status Via **DCU_2.WindowPos** (CAN frame DCU_2 Byte 0).
For **RearLeft Door**, it shall report the determined Lock Status Via **DCU_3.WindowPos** (CAN frame DCU_3 Byte 0).
For **RearRight Door**, it shall report the determined Lock Status Via **DCU_4.WindowPos** (CAN frame DCU_4 Byte 0).
Window Position Determination
WINDOW_IN_BETWEEN
If **SW_WINDOW_OPEN** is determined as **SW_ACTIVE** and **SW_WINDOW_CLOSE** is determined as **SW_INACTIVE**, then Window shall be considered as **IN_BETWEEN**.
WINDOW_COMPLETELY_OPEN
If **SW_WINDOW_OPEN** is determined as **SW_INACTIVE** and **SW_WINDOW_CLOSE** is determined as **SW_INACTIVE**, then Window shall be considered as **COMPLETELY_OPEN**.
WINDOW_COMPLETELY_CLOSE
If **SW_WINDOW_OPEN** is determined as **SW_ACTIVE** and **SW_WINDOW_CLOSE** is determined as **SW_ACTIVE**, then Window shall be considered as **COMPLETELY_CLOSE**.
WINDOW_ERROR
If **SW_WINDOW_OPEN** is determined as **SW_INACTIVE** and **SW_WINDOW_CLOSE** is determined as **SW_ACTIVE** during 10 samples of 10 milliseconds consecutively each, then Window shall be considered as **WINDOW_ERROR**.
If **SW_WINDOW_OPEN** is determined as **SW_ACTIVE** and **SW_WINDOW_CLOSE** is determined as **SW_INACTIVE** after Window Control functionality is re-enable after power cycle (See HW Diagnostics) , then **WINDOW_ERROR** shall not be considered as present.
HW Diagnostics
If window is considered as **WINDOW_ERROR**, then Window Control functionality shall be disable until next power cycle. (Transition OFF -> RUN).
If window is considered as **WINDOW_ERROR**, the corresponding DTC for **Driver Window Position Error Detected** shall be set as DTC: 0xB00100.
If window is considered as **WINDOW_ERROR**, the corresponding DTC for **Passenger Window Position Error Detected** shall be set as DTC: 0xB00200.
If window is considered as **WINDOW_ERROR**, the corresponding DTC for **RearRight Window Position Error Detected** shall be set as DTC: 0xB00300.
If window is considered as **WINDOW_ERROR**, the corresponding DTC for **RearLeft Window Position Error Detected** shall be set as DTC: 0xB00400.
If **WINDOW_ERROR** conditions are no longer detected, the corresponding DTC shall be clear.



Window Completely Closed



Window Completely Open



Window In Between (Example)

Hardware - Software Requirements

Button Debounce

In order to use a mechanism to discard glitches on the buttons a debounce mechanism shall be used.
The debounce mechanism for buttons implies to monitor periodically a signal and increment counters to mature the state of a signal.
All the buttons used on the system shall use an inverted logic.
An inverted logic means that if the buttons are in Low State, then they will be considered as ACTIVE (Idle State shall be high).
Debounce mechanism for buttons consists to increment a counter if the Button State has not change from previous value.
A threshold shall be used to indicate if the Button can be considered as matured (**BTN_PRESSED**) or dematured (**BTN_NOT_PRESSED**).
For Buttons, the threshold value to determine a BTN_PRESSED will be 50ms (6 counts).
For Buttons, the threshold value to determine a BTN_NOT_PRESSED will be 50ms (6 counts).
The Button position will be evaluated periodically every 10ms.

Switch Position Debounce

It is required to use a mechanism to discard glitches on the switches used to determine position.
The debounce mechanism for switches implies to monitor periodically a signal and increment counters to mature the state of a signal.
All the switches used on the system shall use a positive logic.
A positive logic means that if the switches are in High State, then they will be considered as ACTIVE (Idle State shall be high).
Debounce mechanism for switches consists to increment a counter if the Switch State has not change from previous value.
A threshold shall be used to indicate if the Switch can be considered as matured or dematured.
For Switches, the threshold value to determine a BTN_PRESSED will be 30ms (4 counts).
For Switches, the threshold value to determine a BTN_NOT_PRESSED will be 30ms (4 counts).
The Switch position will be evaluated periodically every 10ms.

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.		

Frame DCU_3

Message Layout									
ID	BYTE 0	BYTE 1	BYTE 2	BYTE 3	BYTE 4	BYTE 5	BYTE 6	BYTE 7	PERIOD
0x203	WindowPos	RESERVED	WindowOp	RESERVED	DoorLockSts	RESERVED	DCU_3_MC	DCU_3_CRC	100 ms

Signals Description			
WindowPos:	This Signal reports to the network the Rear Left 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.		
RESERVED	Reserved bytes shall be transmitted ad 0xFF.		
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_3_MC:	This is the Message counter of the DCU_3 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_3_CRC:	This is the Cyclic Redundancy Check for DCU_3 fame. This signal shall be calculated with an CRC8 algorithm from Byte0 to Byte5.		

Frame DCU_4

Message Layout									
ID	BYTE 0	BYTE 1	BYTE 2	BYTE 3	BYTE 4	BYTE 5	BYTE 6	BYTE 7	PERIOD
0x203	WindowPos	RESERVED	WindowOp	RESERVED	DoorLockSts	RESERVED	DCU_4_MC	DCU_4_CRC	100 ms

Signals Description			
WindowPos:	This Signal reports to the network the Rear Right 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.		
RESERVED	Reserved bytes shall be transmitted ad 0xFF.		
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_4_MC:	This is the Message counter of the DCU_4 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_4_CRC:	This is the Cyclic Redundancy Check for DCU_4 fame. This signal shall be calculated with an CRC8 algorithm from Byte0 to Byte5.		

Data Identifier Lists

DID $0111:	Button Configuration		
This DID has the purpose to indicate the value for Stuck Button Detection. 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**
0x0111	StuckBtnCfg		LongBtnCfg
DID_0111. StuckBtnCfg:			
Resolution:	**10 ms**		
Default value:	**1000 (10 000 ms)**		
DID_0111. LongBtnCfg:			
Resolution:	**10 ms**		
Default value:	**50 (500 ms)**		