Title: Door control module SW Architecture

Repositorio en git hub: ArielGP / DiplomadoSistemasEmbebidos_Modulo1

History				
Issue status (Index)	**Maturity/Date** (draft/invalid/valid) (dd-mmm-yyyy)	**Author** Department	**Check/Release** Department	**Description**
1.0	Draft 04-0ct-20	Miguel Garcia	Miguel Garcia	Creation of the document

Table of Contents

1 Purpose

Crate the first release of the architecture to the project "Door control module" based on the requirements described and reviewed in the traceability matrix template and the requirements check list.

2 Definitions and abbreviations

Definitions

Special Byte	*Special byte received to change the baud rate 55h*

Abbreviations

Only SW Architecture specific abbreviations.

References

N°	Document name	Reference
1	*Traceability_matrix_template_v2*	
2	*Requirement _check _list*	

3 Realization constraints and targets

Correct operation of a system which controls, supervises and administrates the door-locking, window position and the associated HW diagnostics of a 4-door vehicle.

A system that has no trouble in resolving operations for 4 doors independently

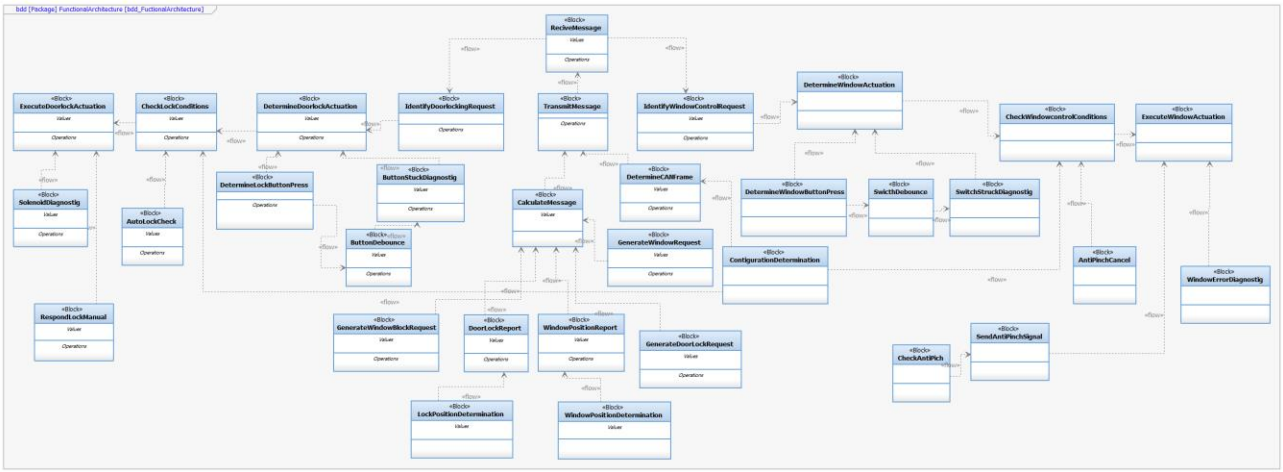
A system that has clear modularity to reuse parts in each of the 4 door configurations

A system that can respond efficiently to both remote and manual inputs

A system that can operate with a set of dynamic restrictions, for example limiting the window control based on a window lock functionality as well as the physical limitations of the window state.

A system that has always enough information about itself to interact with other car systems.

4 SW Functional Architecture



4.1 Table of functions

Function Name	Description
DetermineLockButtonPress	Function that checks for lock button presses and identifies the time pressed to execute the correct window actuation or choose the correct element for the request
DetermineWindowButtonPress	Function that checks for window control switch presses and identifies the time pressed to execute the correct window actuation or choose the correct element for the request
CheckLockConditions	Function that receives crucial information and a lock request that determines if a lock actuation should be executed
CheckWindowcontrolConditions	Function that receives crucial information and a lock request that determines if a lock window actuation should be executed
ExecuteWindowActuation	Function that executes the window actuation
DetermineWindowActuation	Function that determines the appropriated window action based on the window state, the time of the button pressed and other conditions
IdentifyWindowControlRequest	Function that identifies if a window control operation has been requested in a message from the driver door
ExecuteDoorlockActuation	Function that executes the door locking actuation
DetermineDoorlockActuation	Function that determines the appropriated window action based on the lock state and other conditions
IdentifyDoorlockingRequest	Function that identifies if a door locking operation has been requested in a message from a front door
RespondLockManual	Function that blocks software lock operations and gives priority to the lock manual mode, works per individual front door
CalculateMessage	Function that formulates massages based on the CAN frame registered in the configuration.
TransmitMessage	Function that Transmits the calculated message thought the CAN network
ReciveMessage	Function that receives the calculated message thought the CAN

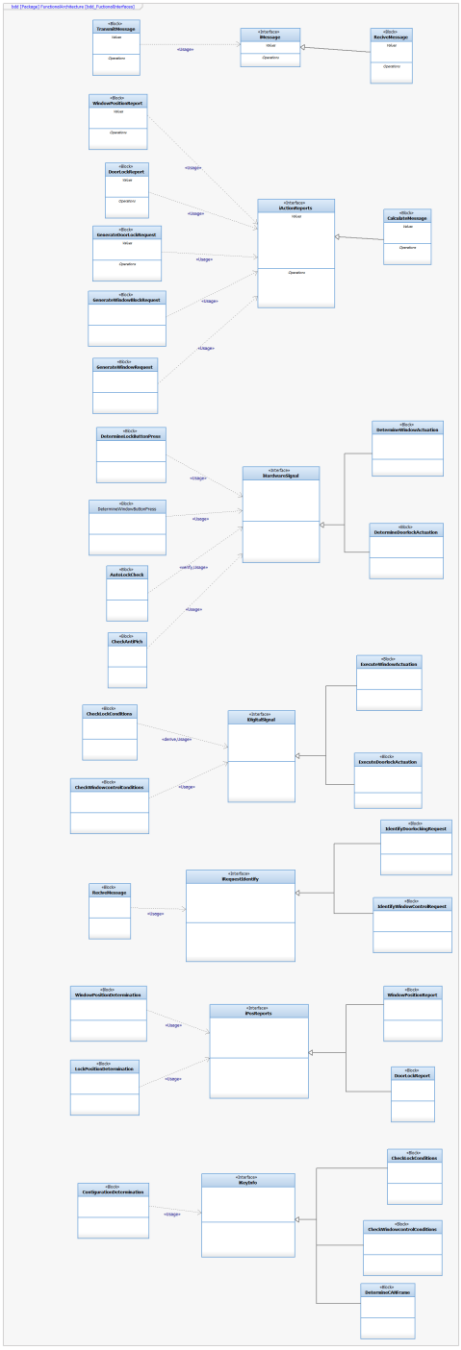
	network
DoorLockReport	Function that identifies and reports the door locking state to send it when the lock button changes positions
ContigurationDetermination	Function that identifies the door based on the original configuration
DetermineCANFrame	Function that identifies the CAN frame for the messages based on the ContigurationDetermination's results
CheckAntiPich	Fuction that checks for window interference during window control open actuations
SendAntiPinchSignal	Function that generates and sends a emergency cancel window actuation request, followed by a global open actuation
AntiPinchCancel	Function that sends a signal to CheckWindowcontrolConditions to block all operations for 15 seconds
WindowPositionDetermination	Function that determines the window position based on past locations and actuations
WindowPositionReport	Function that reports the window position to construct the message
ButtonDebounce	Function that checks for the button debounce conditions and acts upon it
SwicthDebounce	Function that checks for the switch debounce conditions and acts upon it
WindowErrorDiagnostig	Function that stops window control actuations for a power cycle when detecting a window error state.
ButtonStuckDiagnostig	Function that inhibits the button functionality for a power cycle when a button stuck state is reported
SwitchStuckDiagnostig	Function that inhibits the switch functionality for a power cycle when a button stuck state is reported
SolenoidDiagnostig	Function that inhibits the door locking functionality for a power cycle when a door look state is reported
AutoLockCheck	Function that checks externally for auto lock conditions and sends a signal to generate a lock request per unlock door
GenerateWindowRequest	Function that detects and generates a remote window control request and reports the details to construct the CAN message
GenerateDoorLockRequest	Function that detects and generates a remote door lock request and reports the details to construct the CAN message
GenerateWindowBlockRequest	Function that detects and generates a remote window block request and reports the details to construct the CAN message

4.2 Table of functional Interfaces

	Description
iMessage	Format that describes a CAN message following a special framework sent through the network
iActionReports	Information associated with the building of the message, reported in strings with 2 possible contents
iHardwareSignal	Digitalized hardware signal from peripherals
iDigitalSignal	Digital signal indicating special events
iRequestsIdentify	Information associated with a special message that can only be sent through the front doors CAN frames.

iPosReports	Information associated with building a message that describes a position or a state
iKeyInfo	Information associated with the exchange of data related to the validation of conditions

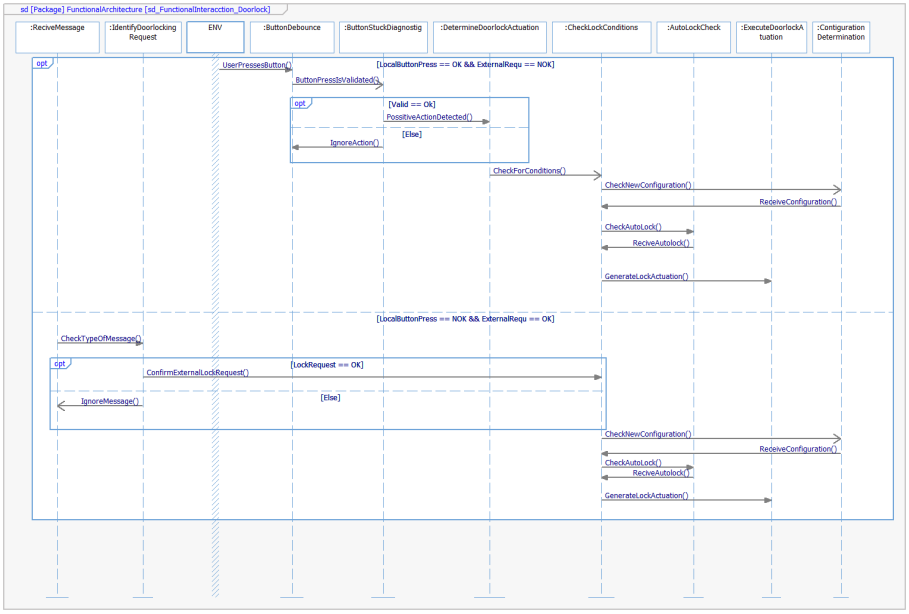
4.3 Functional Interface



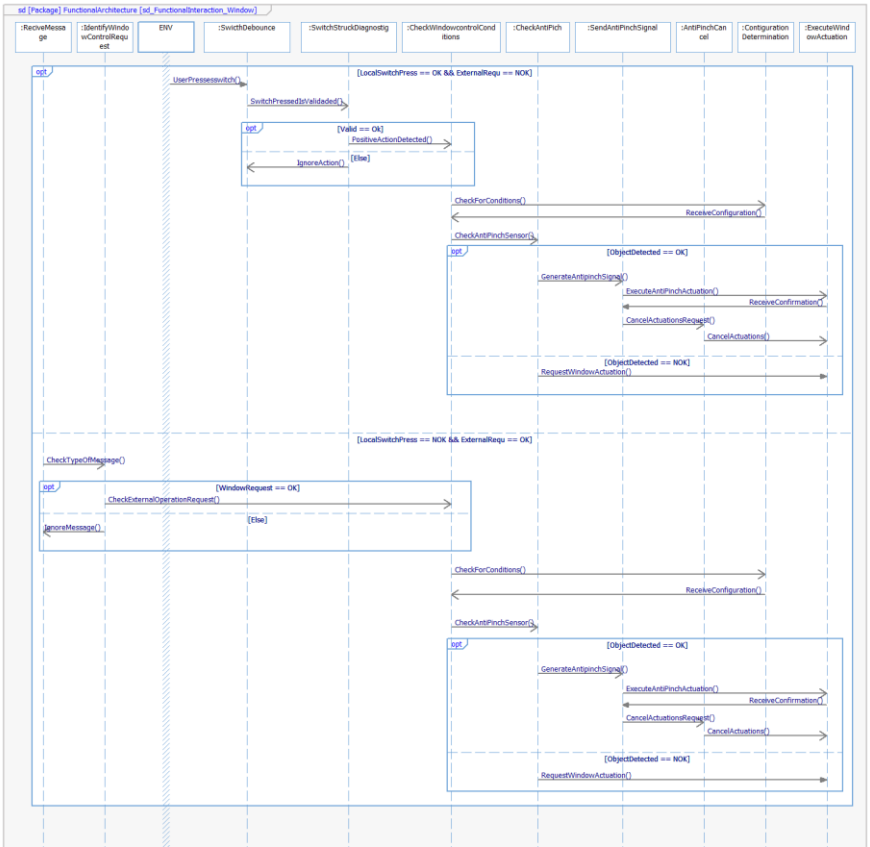
Interface still lacks detail and should be reviewed before publishing

4.4 Functional Interaction

A door lock functional interaction:



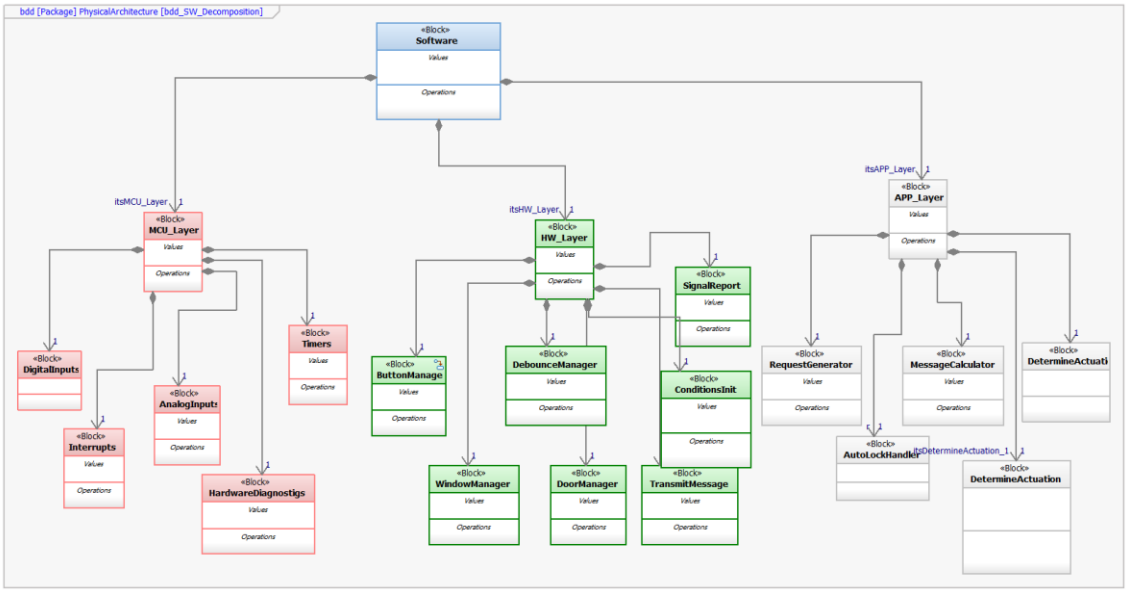
A window control functional interaction:



5 SW Physical Architecture

Define Physical architecture that will be implemented on Software with its corresponding SW decomposition.
Identify different SW Components with its corresponding interaction among them.
Identify and describe the signals and events used to share information among SW Components.

5.1 Physical Decomposition



5.2 Table of SW components

SWC Name	Description	Estimated Size (KB)
DebouceManager	Administrates switch and button debounce protocols	1
AnalogInputs	Reads analog inputs	1
DigitalInputs	Reads digital inputs	1
Timers	Administrates the controller's timers for synchronic applications	2
SignalReport	Reports and identifies the states of relevant actions	3
HardwareDiagnostigs	Handles important hardware diagnostics of the system	2
ConditionsInit	Initialize conditions and configuration	3
RequestGenerator	Generates the signal requests	4
AutoLockHandler	Checks for the auto lock conditions actively and generates auto lock signals	5
MessageCalculator	Calculates messages for the transmission through the CAN network	2
TransmitMessage	Sends formatted messages through the CAN network	3
DetermineActuation	Determines the correct window actuation based on a remote request or a button pressed	2
AntiPinchSystem	Checks for the anti-pinch conditions actively and generates anti pinch signals	3

ButtonManager	Manages button and switches states and functionalities	4
WindowManager	Manages window actuations	3
DoorManager	Manages Door locking actuations	1
DetectRequest	Detects messages with request traits	1
ConditionsHandler	Manages the dynamics of the conditions	2
Interrupts	Manages interrupts	3

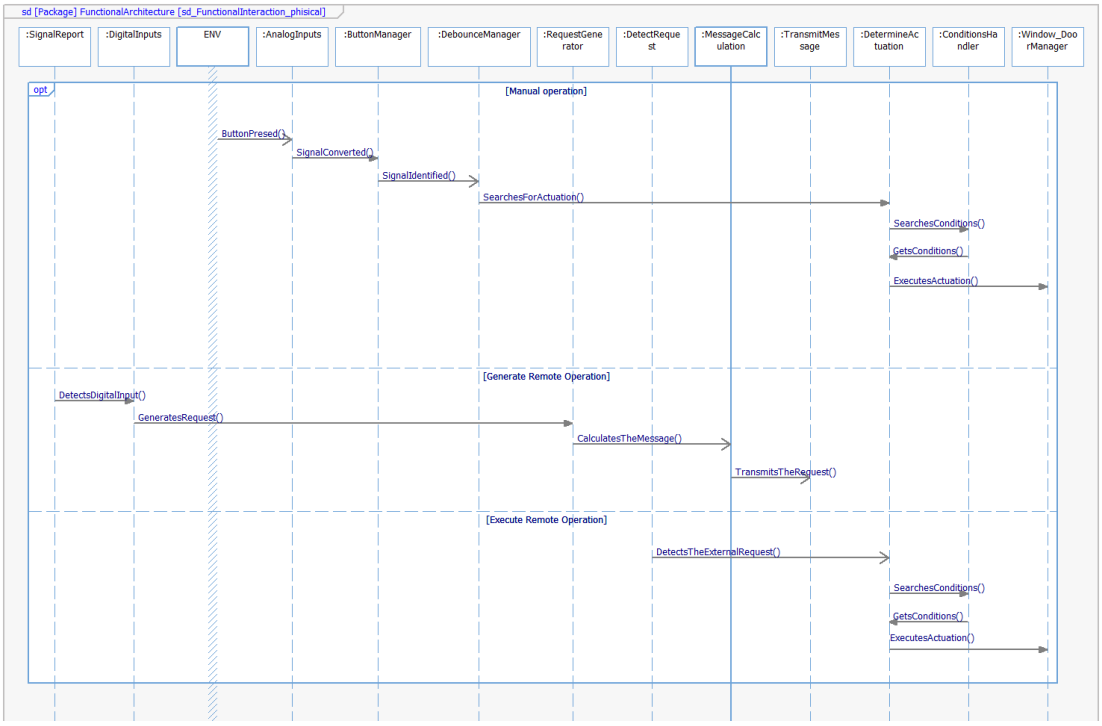
5.3 Table of Physical Interfaces

Operation	Description
iAnalogSignal	Interface designed to exchange digitalized analog signals
iDigitalSignal	Interface designed to exchange info about a request post CAN detection
iMessage	Interface designed to exchange information following the corresponding CAN Frame
iSignalReports	Interface designed to report request
iPositionReport	Interface designed to report information about window and door states calculations
iActuations	Interface designed to exchange information about solicited window or door actuations
iKeyInfo	Interface designed to exchange information about conditions and conditions changes

5.4 Physical Interfaces

Fiscal interface diagram is not yet ready to be documented, isn't mature enough

5.5 Physical Interaction



More details should be added in the future

5.6 Describe Dynamic Behavior

This chapter shall be used to describe at physical level the runtime behavior of the SW.

5.6.1 OS tasks

OS task	Description	Periodicity	Period	Priority	CPU Load
Check for CAN Messages	Continuously check for operation request solicited by other doors	25 ms	1	1	2%
Administrate Button debounce	Executes button/switch debounce during button/switch presses	100 ms	1	2	3%
Identify CAN Requests	Continuously evaluate every message when detected to search for remote requests	50 ms	1	3	2%
Control window Actuations	Execute the all the actions related to the determined window actuation	50 ms	1	2	4%
Control door actuations	Execute the all the actions related to the determined door actuation	50 ms	1	1	4%
Check operation conditions	Continuously checks for actualizations for the operating conditions	100 ms	1	1	2%
Read sensors	Reads external sensors	25 ms	1	1	1%
Auto lock Check	Reads the auto lock conditions on a moving car	100 ms	1	2	1%
Anti-pinch check	Reads the window sensor and the anti-pinch conditions during window actuations	15 ms	1	3	5%
Read states	Calculates states based on sensor reads	25 ms	1	2	3%
Listen to CAN Messages	Periodically listens for CAN message updates	50 ms	1	2	3%

5.6.2 Interrupt Handling

ISR task	Description	Priority	CPU Load
Button Press	Button interrupt to brake idle state and respond to the manual actuation request	1	2%
Anti-pinch total	Request to immediately cancel all window actuations and start the anti-pinch procedure	2	1%

interrupt			
Auto lock actuation priority	Request to immediately cancel the lock idle state and start the Auto lock procedure	2	1%
Request received	Interrupt to immediately respond to remote door lock instructions or window operations request	2	2%

5.6.3 Power Modes

5.6.3.1.1 Low Power Mode

The system does not operate in low power mode.

5.6.3.1.2 High Power Mode

All the system's functions are declared in a high-power mode state.

5.7 Synchronization Mechanisms

To synchronize the system, we used an atom code approach

6 Function to Physical Allocation

Perform the allocation of function to a SW component. One function shall be allocated to one SW component (if this cannot be achieved indicate a clarification wich part of the function shall be allocated to every SW component). One SW component might have

Function	SW component	Clarification
DetermineLockButtonPress	ButtonManager	It should operate with both button presses
DetermineWindowButtonPress	ButtonManager	It should operate with both button presses
CheckLockConditions	ConditionsHandler	
CheckWindowcontrolConditions	ConditionsHandler	
ExecuteWindowActuation	WindowManager	
DetermineWindowActuation	DetermineActuation	
IdentifyWindowControlRequest	DetectRequest	The receive and identify request have been unified
ExecuteDoorlockActuation	DoorManager	
DetermineDoorlockActuation	DetermineActuation	
IdentifyDoorlockingRequest	DetectRequest	The receive and identify request have been unified
RespondLockManual	DoorManager	
CalculateMessage	MessageCalculator	
TransmitMessage	TransmitMessage	
ReciveMessage	DetectRequest	The receive and identify request have been unified
DoorLockReport	SignalReport	
ContigurationDetermination	ConditionsInit	

DetermineCANFrame	ConditionsHandler	
CheckAntiPich	AntiPinchSystem	
SendAntiPinchSignal	AntiPinchSystem	
AntiPinchCancel	AntiPinchSystem	
WindowPositionDetermination	SignalReport	
WindowPositionReport	SignalReport	
ButtonDebounce	DebouceManager	
SwicthDebounce	DebouceManager	
WindowErrorDiagnostig	HardwareDiagnostigs	
ButtonStuckDiagnostig	HardwareDiagnostigs	
SwitchStuckDiagnostig	HardwareDiagnostigs	
SolenoidDiagnostig	HardwareDiagnostigs	
AutoLockCheck	AutoLockHandler	
GenerateWindowRequest	RequestGenerator	
GenerateDoorLockRequest	RequestGenerator	
GenerateWindowBlockRequest	RequestGenerator	

Low level components like interrupts, digital, analog and timers where not concider on the functionality sector, however they will be implemented

7 SW Requirements Allocation

Elements located in the traceability matrix template

8 SW Integration Plan

Identify the dependencies for SW construction. Identify hierarchy for implementation. Indicate the integration steps for SW construction.

Integration Step	Description	SW components	Comments
1.-Sensors and signals	Integrate all related to the detection of signals	AnalogInputs, DigitalInputs, Timers, SignalReport,	
2.-Button interactions	Complement all related to the button interaction and debounce	HardwareDiagnostigs, ButtonManager, Interrupts, DebounceMannager	
3.-Actuation identifications	Integrate steps related to the request identification and validation	DetermineActuation, ConditionsInit, DetectRequest, ConditionsHandler	
4.-Actuation executions	Integrate all function related to the execution of actuations	AutoLockHandler, AntiPinchSystem, WindowManager, DoorManager	
5.-Request generation	Add the components related	RequestGenerator, MessageCalculator,	

	to the request generation	TransmitMessage	

9 Functional Safety

This chapter will be fill during Module 5