|  |  |
| --- | --- |
| **Title:** | **Door Control Module**  **Door Driver Component** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **History** | | | | |
| **Issue status**  (Index) | **Maturity/Date**  (draft/invalid/valid)  (dd-mmm-yyyy) | **Author**  Department | **Check/Release**  Department | **Description** |
| 1.0 | Draft  30/08/2021 | Marco Mares | Marco Mares | Creation of the document |

Table of Contents

[1 Purpose 3](#__RefHeading___Toc357768773)

[2 Definitions and abbreviations 3](#__RefHeading___Toc357768774)

[3 Realization constraints and targets 3](#__RefHeading___Toc357768775)

[4 SW Conceptual design 3](#__RefHeading___Toc357768776)

[5 SW Component internal breakdown 4](#__RefHeading___Toc357768777)

[5.1 Functional Decomposition 4](#__RefHeading___Toc357768778)

[*5.2* Function *<Type> <function name> (type par 1, .., type par n)* 5](#__RefHeading___Toc357768779)

[*5.3* Function <Type*> <function name> (type par 1, .., type par n)* 5](#__RefHeading___Toc357768780)

# Purpose

This document has been created as a demonstration the students to how to create the detailed design from requirements and SW architecture.

# Definitions and abbreviations

**Definitions**

|  |  |
| --- | --- |
| Special Byte | Special byte received to change the baud rate 55h |
|  |  |
|  |  |
|  |  |
|  |  |

**Abbreviations**

Only SW Component specific abbreviations.

References

|  |  |  |
| --- | --- | --- |
| **N°** | **Document name** | **Reference** |
| 1 | SWA\_DCU | SW Architecture description |
| 2 | Traceability Matrix Template.xlsm | SW Requirements description |
|  |  |  |
|  |  |  |

# Realization constraints and targets

The main purpose of the Door Physical Software Component of the ECUAbtraction Layer is to implement all the services to control Door Lock on the ECU. It is responsible to attend request from application to manipulate Door locking operations. It is also responsible to determine and provide lock status to Application.

# SW Conceptual design

Door Component is connected to the following components using the mentioned interfaces:

* Dio Component – iDio Interface
* HWConfig Component – iHWConfig Interface
* DoorApp Component – iDoor Interface

The parts not mentioned in the list above are outside the system boundary.

Door Component uses the following interfaces to get information from other components and send requests to DoorApp Component:

* Door\_Get\_Status():DOOR\_STATUS
* Door\_Init():void
* Door\_Run():void
* Door\_Set\_Request(request: DOOR\_REQUEST):void

Dio component provides the following interfaces and uses the mentioned data types:

* Dio\_Read\_DoorLock():PIN\_VALUE
* Dio\_Read\_DoorUnlock():PIN\_VALUE

HWConfig component provides the following interfaces and uses the mentioned data types:

* HWConfig\_IsAny(config: uint8, config\_mask: uint8):Boolean

A list of the used data types and their values is listed below:

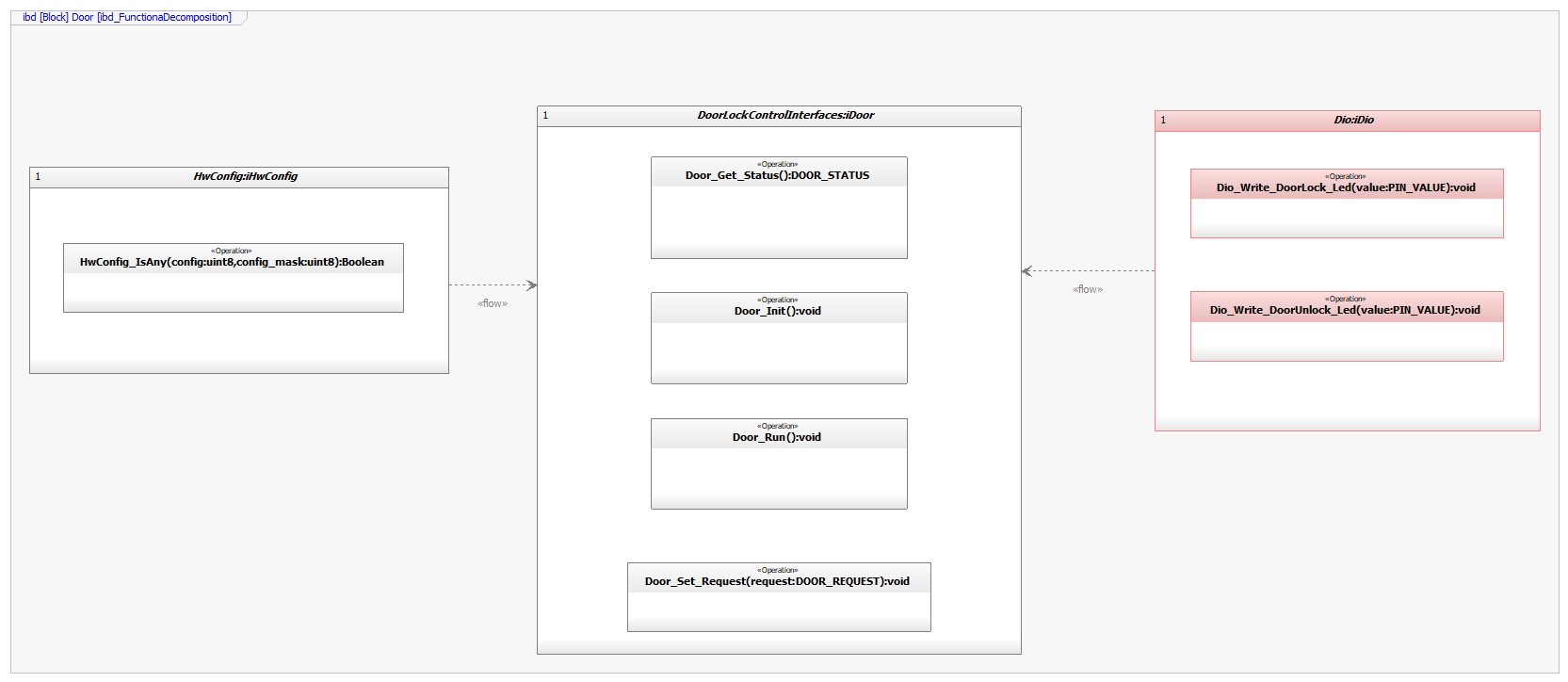
* DOOR\_REQUEST:
  + DOOR\_REQUEST\_NONE = 0
  + DOOR\_REQUEST\_LOCK = 1
  + DOOR\_REQUEST\_UNLOCK = 2
* DOOR\_STATUS:
  + DOOR\_STATUS\_LOCKED = 0
  + DOOR\_STATUS\_UNLOCKED = 1
  + DOOR\_STATUS\_UNKNOWN = 2
  + DOOR\_STATUS\_ERROR = 3
* PIN\_VALUE:
  + DIO\_LOW = 0
  + DIO\_HIGH = 1

# SW Component internal breakdown

Door SW Component will not be break into sub-components. The component is not complex enough to be split into sub-components.

## Functional Decomposition

Overview of functions and their dependencies shown by a Static Function Tree

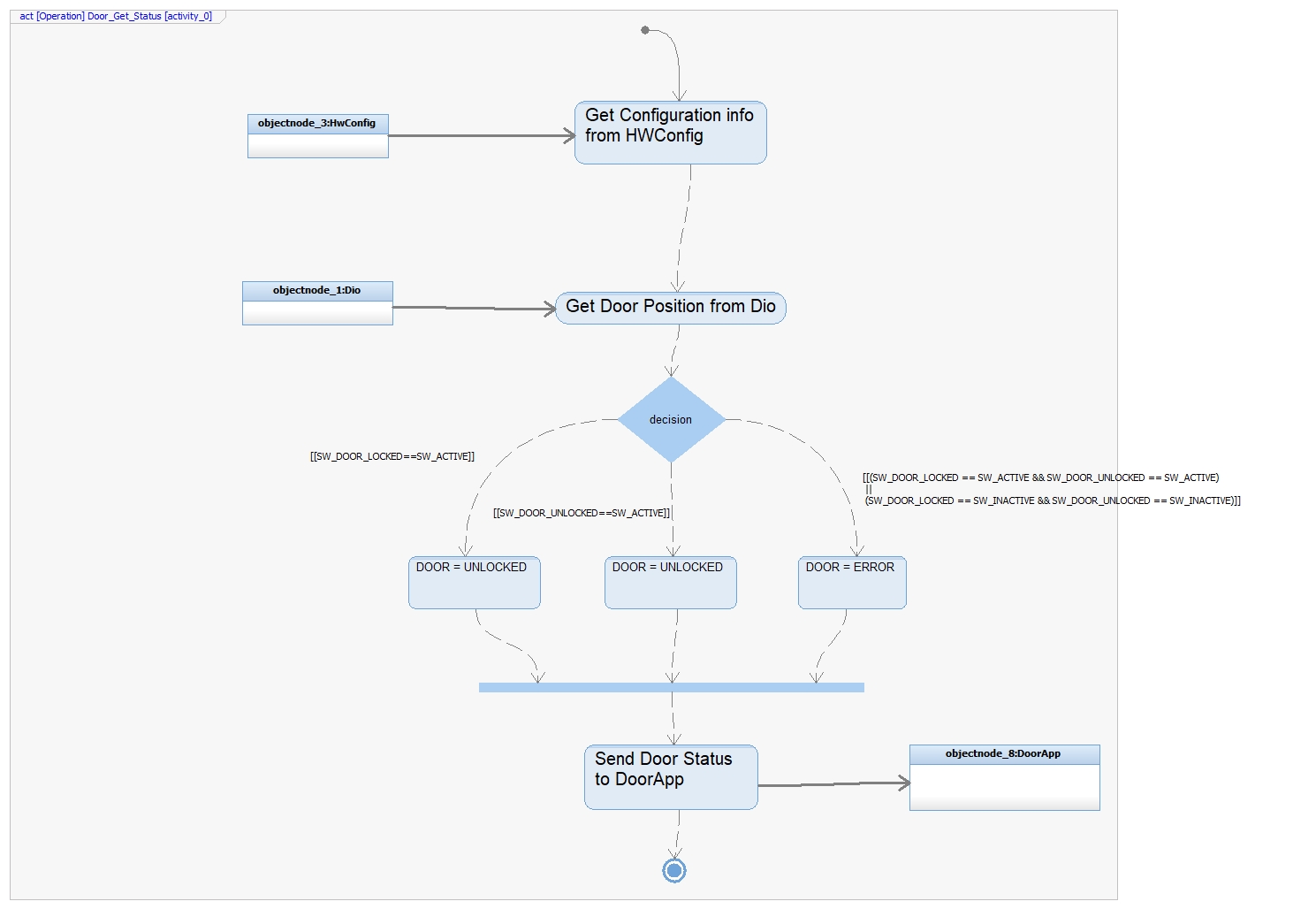


**Function Description and Dynamic Behavior**

## **Function** void Door\_Send\_Request(DOOR\_STATUS, PIN\_VALUE)

|  |  |
| --- | --- |
| **Description** | This function reads the locking status of the Door from the Dio Component, initializes and runs Door Component and sends Door Status information to DoorApp. |
| **Parameter 1** input | PIN\_VALUE |
| **Parameter 2**  output | DOOR\_STATUS |
| **Return Value** | No return value |
| **Precondition** | *Function will periodically check for changes in locking status value* |
| **Post condition** | Depending on the value of DOOR\_STAUS, Door shall be considered as Locked, Unlocked, Unknown or Error. |
| **Error Conditions** | If SW\_DOOR\_LOCKED and SW\_DOOR\_UNLOCKED are determined as SW\_ACTIVE or if SW\_DOOR\_LOCKED and SW\_DOOR\_UNLOCKED are determined as SW\_INACTIVE then Door Lock shall be considered as ERROR |
| **Requirements** | DCU\_SWR\_20  DCU\_SWR\_21  DCU\_SWR\_22  DCU\_SWR\_23  DCU\_SWR\_24 |

**Dynamic Behavior**

  
The activity diagram above describes the behavior of DOOR\_SEND\_REQUEST() operation. This is the only operation needed to achieve Door Component functionalities. The operation starts getting variant configuration from HWConfig component. This configuration defines which door is the ECU controlling. After reading configuration variant, Door component will read PIN\_VALUE of Door Lock Status from Dio Component. Thereafter, Door Component will assign a value to DOOR\_STATUS depending on the status read from Dio Component. Finally, Door Component will create and send a Manual Locking Request to DoorApp Component.