|  |  |  |
| --- | --- | --- |
| **Author** | **Review** | **Approval** |
| SW designer: F.Gilbert | SW team leader: C. Redon | SW line manager: Y. Erb |
|  |  |  |
| **Distribution** | | |
| Project leader: X. Christman  Design leader: S. Delcoy | SW designer: A. LeBeherec  SW designer: T. Demirel  SW designer: C. Sauvage | SW designer : H. Egnifi  SW designer: S. Claveau  SW designer: S. Fléchelle |
|  |  |  |

Detailed Design of HWA Software Component

# Table of content

1. General Information 3

1.1. Revision history \* 3

1.2. Purpose and Scope 3

1.3. Referenced documents 3

1.3.1. External documents 3

1.3.2. Internal Documents 3

1.4. Terminology and definitions 3

2. SW atomic architectural unit design 4

2.1. Overview 4

2.2. Traceability 4

2.3. Files structure 4

3. FEATURES 6

3.1. Services 6

3.1.1. Service Name 6

3.2. Types 7

3.2.1. Name Structure definition 7

3.3. Variables 7

3.4. Constants 7

4. EEPROM 8

5. Configuration 8

5.1. Haptic requests description 8

5.2. Haptic requests parameters 8

6. Compilation Options 8

# General Information

## Revision history \*

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Author(s)** | **Description/comment** |
| 001 | 28.03.2014 | F. Gilbert | First review of document |
|  |  |  |  |
|  |  |  |  |

*\* Template history is found in the CM tool used for templates*

## Purpose and Scope

The review of this document is done thanks to …

The purpose of this document is to describe the design of the PP4G HWA Software Component.

## Referenced documents

### External documents

|  |  |  |
| --- | --- | --- |
| **Id** | **Title** | **Reference** |
|  |  |  |
|  |  |  |
|  |  |  |

### Internal Documents

|  |  |  |
| --- | --- | --- |
| **Id** | **Title** | **Reference** |
|  | SW architecture design interface description | E1355904 in PTC |
|  | ECU parameters specification | DOORS/PP4G/DES/  TF-J |
|  |  |  |
|  |  |  |

## Terminology and definitions

|  |  |
| --- | --- |
| **Terminology** | **Meaning** |
| AAU | Atomic architectural unit |
| MBD | Model Based Design |
| HWA | Haptic Warning |
| SW | software |
|  |  |

# SW atomic architectural unit design

## Overview

The aim of the “HWA” component is to decide to trigger and to decide to abort the HWA cycles.

The design of the “HWA” component is handled by the MBD technology, with the support of tools such as MATLAB environment and a code generator tool such as Target Link for instance.

## Traceability

The traceability matrix is built from the system specification until the architecture document.

Then, refer to [Doc1] the get the traceability against the functional, design and safety requirements, related to this unit.

## Files structure

Below is the description of the files structure defined for this unit.

**HWA\_HapticWarning.h**

**HWA\_Config.h**

**HWA\_HapticWarning.c**

**HWA\_HapticWarning.c**

It will gather:

* The main function of HWA algorithm, called HWA\_runMainFunction(), which is the unique exported function in charge of providing:
  + the triggering decisions of HWA cycles
  + the abortion decisions of HWA cycles
  + the HWA-conditions informations of HWA cycles
  + the HWA-activation informations of HWA cycles
  + the triggering source informations of HWA cycles.
* All the internal functions which provide the steps to decide to trigger and to abort all the HWA cycles.

**HWA\_HapticWarning.h**

It will gather the definition of all exported constants and functions. Only the definition of the unique exported function is available: the HWA\_runMainFunction() runnable.

**HWA\_Config.h**

Parameters used for condition/request calculation are defined into a header file HWA\_Config.h.

This header file includes NVP\_Param and makes the link between HWA and NVP parameters.

# FEATURES

The purpose of this chapter is to only describe the internal implementation of the component.

For the description of the external implementation, please refer to [Doc1] (to get the list of services, types, variables and constants exported by this unit).

Actually, the description of the internal implementation is not necessary for this SW unit since its complexity is extremely low. Therefore, the current chapter shall not be treated. Then, refer to C-code implementation directly.

## Services

### Service Name

The list and the description of services are available in [Doc1].

Refer to this document for more details.

|  |  |  |
| --- | --- | --- |
| **Object** | | |
|  | | |
| **Prototype** | | |
|  | | |
| **Input Parameters** | | |
| Name | Type | Description |
|  |  |  |
| **Output Parameters :** | | |
| Name | Type | Description |
|  |  |  |
| **Return value** | | |
| Type | Description | |
|  |  | |
| **Dynamic aspect** | | |
| Who (callers) | Description | |
|  |  | |
| **Static aspect** | | |
|  | | |
| **Constraints** | | |
|  | | |

## Types

### Name Structure definition

The list and the description of types are available in [Doc1].

Refer to this document for more details.

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Field Type** | **Field description** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Variables

The list and the description of variables are available in [Doc1].

Refer to this document for more details.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Object** | **Type** | **Unit** | **Range** | **Dynamic Aspect** | **Constraints** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

## Constants

The list and the description of constants are available in [Doc1].

Refer to this document for more details.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Object** | **Type** | **Value** | **Unit** | **Constraints** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# EEPROM

The EEPROM parameters are all specified in [Doc2].

Refer to this document for more details.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Object** | **Type** | **Value** | **Unit** | **Constraints** |
| NVP\_u16BfdMaxTemperature |  | U16 | 445 | °C |  |
| NVP\_u16BfdMinTemperature |  | U16 | 262 | °C |  |
| NVP\_u16OverKL30QualTest |  | U16 | 16500 | mV |  |
| NVP\_u16UnderKL30QualTest |  | U16 | 8500 | mV |  |
| NVP\_u16UnderKL30TensQualTest |  | U16 | 7500 | mV |  |
| NVP\_u8HapticFirstCycleDuration |  | U8 | 1 | s |  |

# Configuration

## Haptic requests description

Haptic profile is played with 2 following cycles. The first is used to reduce the belt slack and the second to handle the vibration of the belt. Below the behavior of the request.

CIL Haptic request

Haptic Level

NVP\_HapticFirstCycleDuration

## Haptic requests parameters

Parameters are defined into Config header file. Then the model does not need to be regenerated each time calibration changes.

# Compilation Options