**Project identification**

|  |  |
| --- | --- |
| Project Name | DHF # |
| eAUS | 0001 |

**Visas**

|  |  |  |  |
| --- | --- | --- | --- |
| **Signatures represent review and approval of the indicated revision of the below document** | | | |
| Function | Name | Signature | Date  (DD MMM YYYY) |
| Software Project Manager | Pascal HENCK |  |  |
|  |  |  |  |
| Author | Rafael WEHRMEISTER PADILHA |  |  |

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Description | Author | Effective Date  (DD MMM YYYY) |
| 1.0 | Creation | RWE | 13 JAN 2021 |
| 1.0.1 | Update following parts:   * Terms, abbreviations, references, … * Introduction/Overview * Bootloader FW: Maintenance and Safe * Production FW * Force Sensor calibration * Motor Stall * Error Management (partially) | RWE | 16 FEV 2021 |

Table of Contents

[2 Introduction 3](#_Toc64391840)

[2.1 Document overview 3](#_Toc64391841)

[3 Abbreviation and Terminology 3](#_Toc64391842)

[3.1 References 4](#_Toc64391843)

[3.1.1 Project References 4](#_Toc64391844)

[3.1.2 Standard and regulatory References 4](#_Toc64391849)

[3.2 Conventions 4](#_Toc64391850)

[3.2.1 Requirements conventions 4](#_Toc64391852)

[3.2.2 Writing conventions 5](#_Toc64391853)

[4 System Overview 6](#_Toc64391854)

[4.1 General description 6](#_Toc64391855)

[4.2 CU System Lifecycle 7](#_Toc64391857)

[5 REQUIREMENTS 8](#_Toc64391858)

[5.1 Lifecycle 8](#_Toc64391859)

[5.1.1 Bootloader Firmware 8](#_Toc64391860)

[5.1.2 Production Firmware 10](#_Toc64391861)

[5.1.3 Clinical Firmware 11](#_Toc64391862)

[5.2 Pressure & Pump Control 14](#_Toc64391863)

[5.2.1 Fluid Control 14](#_Toc64391864)

[5.2.2 Force Sensor/Pressure Calibration 15](#_Toc64391865)

[5.2.3 Pressure Sensor 16](#_Toc64391866)

[5.2.4 Motor 16](#_Toc64391867)

[5.3 Battery Management 17](#_Toc64391868)

[5.3.1 End Of Life and ERI 17](#_Toc64391869)

[5.4 Memory access 17](#_Toc64391870)

[5.4.1 File system 17](#_Toc64391871)

[5.4.2 Data Accessor 20](#_Toc64391872)

[5.5 Peripherals 21](#_Toc64391873)

[5.5.1 AlarmClock 21](#_Toc64391874)

[5.6 Safety 22](#_Toc64391875)

[5.6.1 UroTimer 22](#_Toc64391876)

[5.6.2 OverPressure 22](#_Toc64391877)

[5.6.3 Motor Stall 22](#_Toc64391878)

[5.6.4 Error Management 23](#_Toc64391879)

[5.7 Usability and human factors engineering 24](#_Toc64391880)

# Introduction

## Document overview

This document presents the software requirements specifications of eAUS software development project.

It describes:

* Requirements of functionalities, performances, interfaces, environment …
* Tests principles and definitions of validation methods of requirements,
* The compliance of requirements to customer needs,
* The relative importance and precedence of requirements

# Abbreviation and Terminology

| **Terms** | **Definitions** | **References** |
| --- | --- | --- |
| **ADC** | Analog to Digital Converter | N/A |
| **COTS** | Commercial off-the-Shelf Software | N/A |
| **CP** | Clinician Programmer | N/A |
| **CU** | Control Unit (the implant) | N/A |
| **eAUS** | electronic Artificial Urinary Sphincter | N/A |
| **FW** | Firmware, set of software and parameters stored into CU’s memory | N/A |
| **FS** | Force Sensor | N/A |
| **FS/P** | Force Sensor/Pressure | N/A |
| **Gateway** | When PRC is used as a gateway; to transfer command/data between CP and CU as a tunnel | N/A |
| **GRC** | Generic Remote Control | N/A |
| **MCU** | Microcontroller Unit, a computer chip designed for embedded applications | N/A |
| **OC** | Occlusive Cuff | N/A |
| **PRC** | Personal Remote Control | N/A |
| **PV** | Pression-Volume | N/A |
| **RF** | Radio Frequency | N/A |
| **SOUP** | Software of Unknown Provenance | N/A |
| **SRS** | Software Requirements Specification | N/A |
|  |  |  |

## References

### Project References

|  |  |  |
| --- | --- | --- |
| **File #** | **File Name** | **Rev.** |
| P01 | System requirement ?? |  |
| P02 | Risk analysis ?? | ? |
| P03 | GFR0401191630 Software Architecture Description CU | 1.0 |
| P04 | GFR2710171527 Software Requirements Specifications CMD | 1.0 |
| P05 | GFR1610171143Software Requirements Specification COM | 1.0 |
| P06 | MEMO THO2009071814 Driving Motor Without Hall Effect | 1.0 |
| P07 | TECHREQ ABO2007301120 Production SW needs | 1.0 |
| P08 | MEMO LDJ2011161600 Force Pressure Calibration Procedure | 1.0 |
| P09 | MEMO THO 0102211655 PV Calibration | 1.0 |
| P10 | MEMO GFR2008171143 eAUS Firmware Update Procedure | 1.0 |
|  |  |  |

### Standard and regulatory References

|  |  |  |
| --- | --- | --- |
| **File #** | **File Name** | **Rev.** |
| F01 | EN 62304 : Medical Device Software - Software Life Cycle Processes. (Software/Informatics) | 2006+A1:2015 |

## Conventions

### Requirements conventions

Requirements listed in this document are constructed according to the following structure:

|  |  |
| --- | --- |
| Requirement ID | Link to other requirements ID |
| Description | |

Reminder: The ID of SW specification shall be of type EAUS\_SW\_XYYY-ZZZZ

* + - X = 1 for this document
    - YYY representing a subsection shown in the following table:

|  |  |
| --- | --- |
| Subsection | YYY |
| General State | 000 |
| Bootloader | 001 |
| Surgery | 002 |
| Therapy | 003 |
| Follow-up | 004 |
| Implantation | 005 |
| Maintenance | 006 |
| Safe | 007 |
| Communication | 009 |
| Fluid Control | 010 |
| Pressure calibration | 011 |
| Protocol | 012 |
| Battery Management | 013 |
| Data Accessor | 014 |
| Pressure Sensor | 015 |
| Motor | 016 |
| Zarlink | 017 |
| Alarm CLK | 018 |
| Pgc basic | 019 |
| Logs | 020 |
| End Of Life | 023 |

* + - ZZZZ being the increment within the subsection.

### Writing conventions

~~To ease firmware mode’s identification inside the text, all referenced modes are written between square brackets and in bold.~~

~~Example: “[~~**~~MyMode~~**~~]”.~~

~~To ease firmware memory area’s identification inside the text, all referenced memory areas are written between parenthesis and in bold.~~

~~Example: “(~~**~~MyMemoryArea~~**~~)”.~~

~~To ease firmware parameter’s identification inside the text, all referenced parameters are written between simple quotes and in bold.~~

~~Example: “’~~**~~MyParam~~**~~’”.~~

~~To ease communication command’s identification inside the text, all referenced commands are written between braces and in bold.~~

~~Example: “{~~**~~MyCmd~~**~~}”.~~

~~To ease firmware error’s identification inside the text, all referenced errors are written between square brackets and in italic.~~

~~Example: “[~~*~~My\_Error~~*~~]”.~~

~~Example:~~

|  |  |
| --- | --- |
| ~~EAUS\_SRS\_1610171143\_0024~~ | ~~EAUS-SYS-0000-0097~~ |
| ~~In [~~**~~Activated~~**~~] mode, UroCath conditions shall be:~~  ~~Activity is below~~ **~~DecubitusAccelerationThresholdValue~~** ~~and the pressure is rising above~~ **~~UroCathPressureThresholdValue~~** ~~within a delay shorter than~~ **~~UroCathRisingTimeValue~~** ~~and maintained above~~ **~~UroCathPressureThresholdValue~~** ~~during~~ **~~UroCathHoldingTimeValue~~**~~.~~ | |

**The verbs shall and will:**

We adopt the commonly used formalism to distinguish between a requirement of the software and a statement of fact using the verbs ‘shall’ and ‘will’. Here *shall* is used to dictate the provision of a functional capability whereas w*ill* is used to cite things that the operational or development environment are to provide to the capability being specified. For example, "The hospital's network infrastructure will be used to connect the XXX SOFTWARE system to the PACS."

# System Overview

## General description

The electronic Artificial Urinary Sphincter (eAUS) is composed of 3 sub-systems, the Clinician Programmer (CP), the Control Unit (CU) and the Patient Remote Control (PRC). The CP is a tool for the Surgical Team to communicate with the implanted CU through a PRC.

The PRC has several purposes in the eAUS. It can be used as a standalone controlling device (standalone mode): its purpose is then to enable the patient to open the cuff on demand, to report alarms of the CU and to activate/deactivate the CU if required. Used in combination with the CP, it plays the role of communication gateway between CP and CU (managed mode). It communicates with Bluetooth for PRC and CP link and with a specific medical radio protocol for PRC and CU link.

For eAUS-Gen1, four variations of the PRC are possible (all have a different “main purpose”):

* 3 types of PRC for the patient:
  + Voiding PRC (main purpose: urinate, presented above)
  + Baseline PRC (main purpose: adapt urethral compression when getting up)
  + Laying down (main purpose: adapt urethral compression when lying down)
* 1 type of remote control for the clinical team:
  + Generic Remote Control (GRC) (main purpose: used for implantation and patient follow-ups)

Finally, the CU is the part implanted into the patient. It is mainly composed of two distinct parts, a casing with the control components (radio, electronic board, pump, pressure etc.) and an occlusive cuff (OC) with isotonic fluid inside. The overall system purpose is to adjust the fluid pressure inside the cuff in order to compress the patient urethra thus acting as an artificial sphincter. Radio is used for external communication purposes.

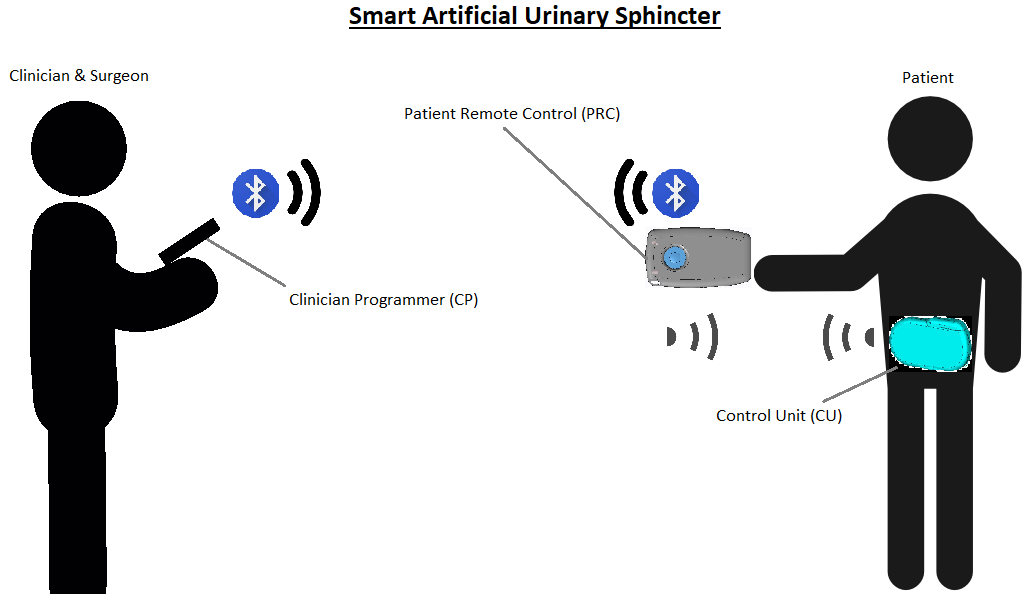


Figure 1: eAUS-Gen1 active devices overview

## CU System Lifecycle

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_SYS\_0000\_0282,  EAUS\_SYS\_0000\_0233,  EAUS\_SYS\_0000\_0273,  EAUS\_SYS\_0000\_0307 |
| The CU shall have a state diagram as:    Figure 2 - CU device lifecycle | |

# REQUIREMENTS

The Control unit firmware has been separated into three standalone software in order to segregate their features: Bootloader FW, Production FW and Clinical FW. Each software will configure peripherals to better fit the intended use and test software and hardware upon specific test case. During operation, each of them gives access to a restricted set of features and a reboot will be needed to enter other sequence. The Bootloader firmware is in charge of launching one of Applicative software’s, Production or Clinical, with the right sequence.

## Lifecycle

### Bootloader Firmware

Control Unit Bootloader firmware is responsible for the following features:

* Retrieve some information such as firmware or hardware identification.
* Read and adjust technical settings.
* Update (upgrade or downgrade) device Applicative software.
* Enter in a safe state in case of error.

Once Power-Up has occurred, the Bootloader firmware starts automatically in autotest mode, a necessary step to ensure that the firmware and hardware are in the expected state before executing sensitive code. Upon autotest success, the Bootloader firmware checks the reason of reset and either enters in **Maintenance** mode, **Safe** mode or launches the Applicative firmware with the appropriate sequence.

In **Maintenance** mode an authorized user is able through the CP to update the Applicative software, read technical parameters, retrieve firmware identification, algorithm tuning parameters and read debug logs. It allows entering other modes if no error is present.

In **Safe** mode the firmware is stalled, and user is able through CP to read error code and switch to **Maintenance**. Whenever rebooting into **Safe** mode, the device is first set in a configuration that is safe for the patient (the Occlusive Cuff is opened if possible, using a last chance opening sequence [P06]).

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0003 | EAUS\_RSK\_0000\_0071,  EAUS\_RSK\_0000\_0084,  EAUS\_RSK\_0000\_0297,  EAUS\_RSK\_0000\_0315 |
| The CU shall check its memory integrity using CRC for the following cases at the start-up (autotests):   * MCU ROM manufacturing parameters * Motor position * Technical and clinical parameters * Applicative firmware | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0004 | EAUS\_RSK\_0000\_0051,  EAUS\_RSK\_0000\_0297,  EAUS\_RSK\_0000\_0392 |
| The CU shall perform an SPI autotest at the bootloader start-up to verify that the radio chip is accessible. This will be performed by reading the Zarlink module ID. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0005 | ~~EAUS\_RSK\_0000\_0392~~ |
| ~~The CU shall execute an autotest on ADC reading a stable signal and verify returned value .~~ | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0002 | EAUS\_RSK\_0000\_0040 |
| Upon reset, the CU shall go to Bootloader firmware first and check the reset reason to decide if the CU shall go to Safe mode. | |

#### Maintenance Mode

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_RSK\_0000\_0150,  EAUS\_RSK\_0000\_0291,  EAUS\_RSK\_0000\_0300 |
| The CU shall implement the following sequence for firmware update:  - Reboot in bootloader maintenance mode  - New firmware upload to a temporary memory location  - CRC check of uploaded firmware using a proprietary algorithm for obfuscating integrity-checksum computation  - Retries download if CRC is wrong  - Erase the medical application  - Write the new firmware from temporary location into medical application memory space  - Check CRC of new medical application  - Retries copy if CRC is wrong  - Check CRC of all other non-volatile memory area | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0002 | EAUS\_RSK\_0000\_0158,  EAUS\_RSK\_0000\_0189,  EAUS\_RSK\_0000\_0291,  EAUS\_RSK\_0000\_0206 |
| The CU shall deflate the cuff when entering Maintenance mode, except when the CU turn from Production to Maintenance mode. | |

The list of commands available in Maintenance mode can be found in [P04].

#### Safe Mode

The main role of the Safe Mode is to open the cuff at all costs in order to prevent damage to the patient.

After securing the implant, the software only waits for a technician to wirelessly connect to it, to perform maintenance operations.

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_ | EAUS\_RSK\_0000\_0189 |
| The CU shall deflate the cuff when entering Safe mode by executing a last cuff opening order. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_ | EAUS\_RSK\_0000\_0282 |
| If the hall-effect sensors are damaged, the CU shall execute a last cuff opening order without using hall-effect sensors. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_ | EAUS\_RSK\_0000\_0282 |
| Safe mode operations shall be as represented on the following diagram:     * The software shall move the pump to its lowest position. * If a hall-effect sensors error is detected while driving the pump, the software shall try to drive the pump to its lowest position without using the hall-effect sensors. * After moving the pump, the software shall only wait for a wireless connection. | |

|  |  |
| --- | --- |
|  |  |
| Upon entering Safe mode, if the CU detects a motor stall and it shall try a second time. For this, it shall not execute a last cuff opening order without using hall-effect sensors as it can damage the pump. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_ |  |
| In Safe mode, the only authorized go to mode is Maintenance mode. | |

The list of commands available in Safe mode can be found in [P04].

### Production Firmware

Control Unit Production firmware allows the Manufacturing and Verification team in charge of Control Unit production to execute tests and set parameters. It shall only be used during the manufacturing process as otherwise the patient could be put in danger.

#### Production Mode

Control Unit Production firmware has only one mode and it is responsible for the following features:

* Retrieve some information such as firmware or hardware identification.
* Read and adjust technical settings.
* Configure motor parameters.
* Execute force sensor/pressure calibration.

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_ | TECHREQ\_Req1 |
| Production firmware shall not be used in already implanted Control Unit. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_ | TECHREQ\_Req2 |
| Production firmware shall be replaced by Clinical firmware after manufacturing process. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_ | TECHREQ\_Req3 |
| Technical settings generated by Production firmware shall be stored in flash. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_ | EAUS\_SYS\_0000\_0281\_2 |
| The CU shall be able to perform a calibration of its internal force sensor in Production mode (see 5.2.2). | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_ | Req6 |
| The CU shall provide a way to execute pump cycling (operate a requested number of complete strokes) and store its context. If the memory storage is full, the CU shall keep cycling till the number of requested cycles has been reached. | |

The list of commands available in Production mode can be found in [P04].

### Clinical Firmware

#### Implantation Mode

Control unit Surgery sequence shall be entered only once and is responsible for the following features:

* Wait for connection requests and connect with CP when available
* Drive the pump according to motion sensor output or PRC commands

In [**Implantation**] mode it waits for following commands : {**SavePatientChartCmd**} (then do the appropriate actions in (**PatientChartArea**)), {**PrimeAirCmd**} (to do the appropriate action on pump) and {**SetMotorPositionCmd**} & {**GetPressureCmd**} commands (to do the appropriate action on pump and pressure sensor including writing to (**PressureVolumeArea**)). It is also possible to reboots for [**Therapy**] mode. During all its execution, it can write useful information into (**DataLogArea**) and (**DebugLogArea**) logging tables.

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_SYS\_0000\_0050 |
| From Implementation mode the only mode change command authorized is "goToDeactivated".  (see state diagram)  Test:  go to implantation mode and test all "goto another mode" commands, just "goToDeactivated" command shall pass, others are unauthorized | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_RSK\_0000\_0153 |
| No permanent link to implementation mode (just one time). Link to state diagram | |

|  |  |
| --- | --- |
|  |  |
|  | |

#### Therapy Mode

Control Unit Therapy sequence is the day-to-day execution at-home and is responsible for the following features:

* Wait for connection requests and connect with PRC and CP when available
* Drive the pump according to PRC commands
* Detect safety-related conditions and operate a complete deactivation
* Temporarily deactivate and activate therapy features

Once the first post-surgery follow-up has been performed or after a complete deactivation, the Therapy sequence can be launched. It starts in [**Autotest**] mode in order to verify that conditions are met for proper operation of Therapy features. Upon autotest failure it reboots in [**Safe**] mode. Upon autotest success it enters in [**Deactivated**] mode.

In [**Deactivated**] mode the software waits for {**ActivateCmd**} or {**GoToFollowUpCmd**} commands. Upon {**GoToFollowUpCmd**} command reception it switches off the appropriate modules, saves any useful status then reboots in [**Follow-up**] mode. Upon {**ActivateCmd**} command reception and after a successful autotest, it orders a cuff inflation to ‘**BaselinePressureValue**’ and changes mode for [**Activated**] mode. Helped by parameters in (**TechnicalSettingsArea**) it can interpret low battery level and wait for a cuff deflation before rebooting in [**Safe**] mode.

In [**Activated**] mode the software operates the fluidic circuit, monitors safety-related inputs, listens on communication link for incoming connection initialization. Helped by the parameters in (**TechnicalSettingsArea**), the software can command the fluidic circuit to increase or decrease pressure inside the cuff helped by driving motor position. Upon reception of {**VoidingCmd**} command the software can command fluidic system to decrease pressure inside the cuff helped by parameters in (**TechnicalSettingsArea**). Upon reception of {**ApplyBaselinePositionCmd**} or {**ApplyLyingDownPositionCmd**} command the software can command fluidic system to increase or to decrease pressure inside the cuff helped by parameters in (**TechnicalSettingsArea**). Helped by parameters in (**TechnicalSettingsArea**) it can interpret low battery level and wait for a cuff deflation before rebooting in [**Safe**] mode. Helped by parameters in (**FunctionalSettingsArea**) it can verify the time during which the cuff has been inflated and order a cuff deflation and reboot in [**Safe**] mode. Upon reception of {**DeactivateCmd**} command it orders a cuff deflation and changes mode to [**Deactivated**] mode. During all its execution, it can write useful information into (**DataLogArea**) and (**DebugLogArea**) logging tables.

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_SYS\_0000\_0290,  EAUS\_RSK\_0000\_0197,  EAUS\_RSK\_0000\_0131 |
| In Therapy mode / Deactivated, switching to Activated mode by a PRC incoming wireless command shall be authorized if and only if the previous deactivation source was PRC. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0002 | EAUS\_SYS\_0000\_0070 |
| First time in activated mode the CU shall go to baseline position.  Test:  after a goToActivatedMode command from follow up mode or implantation mode we check the motor position, it shall be baseline | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0003 | EAUS\_SYS\_0000\_0106, EAUS\_SYS\_0000\_0108 |
| The event is stored in logs.  Test:  goTo follow up, Set the UroTimer to default value (12 hours), read the value in flash and check if its 12 hours, come back to activated mode, and execute a voiding command, after exactly 12 hours check when the CU goes to deactivated mode, the time shall be 12 hours +/- 1 minute | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0004 | EAUS\_SYS\_0000\_0249, EAUS\_SYS\_0000\_0250 |
| After a voiding order, the CU shall wait a period of time with cuff deflated before coming back to the baseline position.  Test:  go to activate, set the voiding period to default one (75 seconds), read the voiding period and check if it is the same that we already send, send a voiding order wait the exactly the same period of time, read the motor position multiple times until the returned position is baseline. check if the time is the same in a chronometer +/- 1 seconds | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0005 | EAUS\_SYS\_0000\_0070 |
| First time in activated mode the CU shall go to baseline position. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0006 | EAUS\_SYS\_0000\_0249 |
| After a voiding order the CU shall wait a period of time with cuff deflated before coming back to the baseline position. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_xxxx | EAUS\_RSK\_0000\_0182 |
| Upon entering in Deactivated mode, the CU shall deflate the occlusive cuff (voiding position) without returning to baseline position. | |

|  |  |
| --- | --- |
|  |  |
|  | |

#### Follow-Up Mode

Control unit Follow-up sequence is responsible for the following features:

* Wait for connection requests and connect with CP when available
* Execute command on CU from CP
* Retrieve patient/usage information on the CP
* Switch to [**Maintenance**] mode upon technician authentication

The follow-up interfaces are:

* All peripherals including battery, pump, communication and pressure sensor.
* Reduced command interpretation: {**GetStatusCmd**}, {**GoToMaintenanceCmd**}, {**SetFunctionalParameterValueCmd**}, {**GoToMaintenanceCmd**}, {**GoToTherapyCmd**}, {**GetPatientChartCmd**}, {**GetFunctionalSettingCmd**}, {**GetPatientLogs**}, {**VoidingCmd**}.
* Restricted data storage peripheral read-only access, (**DataLogArea**), (**DebugLogArea**), (**PatientChartArea**)
* Restricted data storage peripheral read/write access, (**FunctionalSettingsArea**), (**PressureVolumeArea**).

Every time the CP is connecting to CU in Therapy sequence it sends {**GoToFollowUpCmd**} command and cause the CU to reboot in Follow-up sequence. When launched, the Follow-up sequence enters [**Autotest**] mode in order to verify that conditions are met for proper operation. Upon autotest failure it reboots in [**Safe**] mode. Upon autotest success it enters in [**Follow-up**] mode.

In [**Follow-up**] mode, the software is waiting for incoming commands to execute. Upon reception of a command the appropriate action(s) is (are) performed. Upon reception of {**GoToTherapyCmd**} command the it reboots in [**Deactivated**] mode of Therapy sequence. {**GetPatientChartCmd**} command retrieves patient and device information in (**PatientChartArea**). {**VoidingCmd**} command will apply the ‘**VoidingMotorPositionValue**’ on fluid circuit. {**GoToMaintenanceCmd**} command will reboot in [**Maintenance**] mode of Bootloader software. {**SetFunctionalParameterValueCmd**} command writes a new parameter in (**FunctionalSettingsArea**). {**GetFunctionalSettingCmd**} command retrieves the parameters in (**FunctionalSettingsArea**). {**GetPatientLogs**} command retrieves the logs in (**DataLogArea**). {**GetFirmwareVersionCmd**} command returns firmware identification. Helped by parameters in (**TechnicalSettingsArea**) it can interpret low battery level and wait for a cuff deflation before rebooting in [**Safe**] mode.

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_SYS\_0000\_0069, EAUS\_SYS\_0000\_0247 |
| The CU shall be able to perform a second pressure/volume calibration in Follow-Up mode.  Test:  just in follow-up mode where this calibration is authorized ????? | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0002 | EAUS\_SYS\_0000\_0069 |
| The CU shall be able to perform a second pressure/volume calibration in Follow-Up mode (see MEMO\_BRO\_2003041100\_REV\_0\_1). | |

#### End of Life Mode

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_xxxx | Need syspec,  EAUS\_RSK\_0000\_0103,  EAUS\_RSK\_0000\_0173,  EAUS\_RSK\_0000\_0270,  EAUS\_RSK\_0000\_0322,  EAUS\_RSK\_0000\_0351, |
| When EOL warning is detected the CU shall go to EOL mode. | |

|  |  |
| --- | --- |
|  |  |
|  | |

## Pressure & Pump Control

The role of the pressure measurement is to estimate the pressure applied on the urethra. To measure the absolute pressure in the reservoir, a force sensor is used, and the force measured is linked to the pressure itself.

### Fluid Control











|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_SYS\_0000\_0246 |
| The CU shall do a first pressure calibration: CU shall drive automatically the motor to different positions (200 points default value) and check every time the force sensor returned value (see MEMO\_BRO\_2003041100\_REV\_0\_1). | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0003 | EAUS\_SYS\_0000\_0069 |
| In the second pressure/ volume calibration: The CU shall drive automatically the motor to different positions (200 points default value) and check the relative pressure using the first calibration table [The CU is connected to the Cuff] (see MEMO\_BRO\_2003041100\_REV\_0\_1). | |



|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0004 | Need syspec,  EAUS\_RSK\_0000\_0219,  EAUS\_RSK\_0000\_0227,  EAUS\_RSK\_0000\_0237,  EAUS\_RSK\_0000\_0253, |
| The CU shall detect overpressure tendency by following this:   * Before every pump movement the CU check relative pressure value * If this pressure is higher than the overpressure threshold (default value 150cmH2O) for a consecutive number of times (default value is 10) a warning shall be set   If the number of overpressure occurrence in flash memory is 0, the CU shall deactivate this feature of overpressure detection. | |

|  |  |
| --- | --- |
|  |  |
|  | |

### Force Sensor/Pressure Calibration

The goal of Force Sensor/Pressure calibration (FS/P calibration) is to link the force sensor’s output to the absolute pressure in the reservoir. For further details see [P08] MEMO LDJ2011161600 - Force Pressure Calibration Procedure.

#### How to calibrate

|  |  |
| --- | --- |
|  |  |
| The CU shall be able to store from 2 up to 10 force sensor/pressure calibration tables. | |

|  |  |
| --- | --- |
|  |  |
| One force sensor/pressure calibration table shall be stored as follows:   * Table number – 4 bytes * Pressure in cmH2O – 4 bytes * Number of points – 4 bytes * Motor position/force sensor value mapping table composed of:   + Motor position – 4 bytes   + Force sensor ADC value scaled to battery level – 4 bytes * CRC – 4 bytes | |

|  |  |
| --- | --- |
|  |  |
| The CU shall be able to store up to 200 points in motor position/force sensor value mapping table. | |

|  |  |
| --- | --- |
|  |  |
| Before starting the FS/P calibration procedure, the motor shall be moved to its highest position (0x0000). | |

|  |  |
| --- | --- |
|  |  |
| The FS/P calibration starts from the highest motor position (0x0000) to the lowest (0x1DC7). | |

|  |  |
| --- | --- |
|  |  |
| The FS/P calibration tables shall be sorted with the lowest pressure (P1) at the beginning (table n° 1) and the highest at the end. | |

|  |  |
| --- | --- |
|  |  |
| For each calibration, the CU shall record the motor position and the associated force sensor output scaled to battery level. | |

|  |  |
| --- | --- |
|  |  |
| The motor shall be moved to its highest position (0x0000) after each FS/P calibration table record. | |

|  |  |
| --- | --- |
|  |  |
| If an error event (i.e.: motor stalls) happens during the FS/P calibration, the CU shall abort the procedure, log the error and do not taken into account the current calibration. The remaining calibration tables shall be kept as is. | |

#### Computing the absolute pressure in cmH2O

|  |  |
| --- | --- |
|  | EAUS\_SYS\_0000\_0247 |
| Once the FS/P calibration is finished, the CU shall compute the pressure in cmH2O as follows:   1. Read current motor position. 2. Read current force sensor ADC output. 3. Read current battery level. 4. Scale force sensor output to the battery level. 5. For the given motor position, use linear interpolation to calculate the ADC output (scaled to battery level) for each calibration pressure value. 6. Compare the found values with the reading of the sensor, in order to find two ADC values. 7. Calculate the pressure in cmH2O by performing a linear interpolation using the couple of values found in step 6. 8. Return the pressure value in cmH2O. | |

|  |  |
| --- | --- |
|  |  |
| The CU shall start the comparison from lowest FS ADC value and continues up until it finds a value bigger than the one it reads. | |

|  |  |
| --- | --- |
|  |  |
| If the FS ADC value read is lower than the one stored in the first table, use the two smallest pressures recorded (i.e. P1 and P2). | |

|  |  |
| --- | --- |
|  |  |
| If the FS ADC value read is higher the one stored in the last table, use the two highest pressures recorded. (i.e. PN and PN-1 where N is lower than 10). | |

### Pressure Sensor

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_SYS\_0000\_0246 |
| The CU shall drive the force sensor to read returned value (see MEMO\_BRO\_2003041100\_REV\_0\_1).  Test:  Apply an external pressure on the reservoir and send a read pressure command and see that when the pressure goes up the command returned value goes up too | |

|  |  |
| --- | --- |
|  |  |
|  | |

### Motor

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_SYS\_0000\_0248 |
| The CU first calibration shall consider all motor range to be able to calculate the largest possible perimeter  (see MEMO\_BRO\_2003041100\_REV\_0\_1).  Test:  check the first and last points in 1 calibration table in flash, it shall be 0 and 0x1CF4 | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0002 | EAUS\_RSK\_0000\_0219,  EAUS\_RSK\_0000\_0253,  EAUS\_RSK\_0000\_0280,  EAUS\_RSK\_0000\_0281 |
| The motor shall read the hall-effect sensors when it is running to control its direction and position with an accuracy of +/- 100 steps. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0003 | EAUS\_RSK\_0000\_0227 |
| The maximum motor position authorized in CU shall matches with 150 cmH2O. The CU, before executing setMotorPosition, shall check if the position is less than the maximum authorized. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0004 | EAUS\_SYS\_0003\_0005 |
| The CU shall fill the entire reservoir in less than 15 seconds. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0005 |  |
| The maximum step for the motor shall be equal to 0x1CF4 | |

|  |  |
| --- | --- |
|  |  |
|  | |

## Battery Management

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_SYS\_0000\_0142 |
| The CU shall save the ERI in flash.  Test:  will be tested with all EOL mode | |

### End Of Life and ERI



|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | Need syspec,  EAUS\_RSK\_0000\_0273, |
| The CU shall detect an ERI warning when the battery level is lower than ERI threshold (default value 2.5v) for 3 consecutive times just before pump movement. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0002 | Need syspec,  EAUS\_RSK\_0000\_0273, |
| The CU shall detect an EOL warning when the battery level is lower than EOL threshold (default value 2.4v) for 3 consecutive times just before pump movement. | |

## Memory access

### File system

#### Clinical Logs

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_RSK\_0000\_0082 |
| The CU shall define the maximum data size in Flash memory from the beginning, see memory mapping section in SAD | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_SYS\_xxxx\_xxxx |
| The CU shall provide a way to store timestamped logs. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0002 | EAUS\_SYS\_0000\_0158 |
| The CU shall log warnings of the following list with the following context:   |  |  | | --- | --- | | Warning category | Warning context to log | | Motor issue | current position of the motor, hall-effect state | | Wireless attack |  | |  | Probleme de com [tentative de connexion avec mauvaise authentification] contexte = vide | |  | Tentative d'attaque radio [mauvais ID en 2.45]: contexte = IMPLANT ID | |  | Probleme de batterie : contexte = niveau de batterie | |  | Tentative d'attaque [commande interdite]: contexte = identifiant de la commande | |  | Probleme de capteurs: contexte = valeur hors range | | |

Test: Erase warning from logs

* Simulate a warning [eg. Radio attack]
* Retrieve logs, [see CMD spec]
* Check the warnings timestamps/reason/context

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0003 | EAUS\_SYS\_0002\_0037 |
| The CU shall provide a way to store the last 20 Activation/Deactivation events.  Each logged event shall contain the following information:   * Timestamp * Event type [activation/deactivation] * Source of the event [PRC/other] | |

Test: Erase logs, set the CU to activated mode

* With a paired PRC, push on the ACTIVATION/DEACTIVATION button
* With a paired PRC, push on the ACTIVATION/DEACTIVATION button
* With a CP, connect to the CU and send a Therapy switch cmd
* Retrieve logs, ensure there are 3 activation/deactivation events logged, check the source of the events [PRC/PRC/other].

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0004 | EAUS\_SYS\_xxxx\_xxxx |
| The CU shall provide a way to store the last 20 UroTimer delay change events.  Each logged event shall contain the following information:   * Timestamp – 4 bytes * UroTimer delay value [minutes] – 4 bytes | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0005 | EAUS\_SYS\_xxxx\_xxxx |
| The CU shall provide a way to store the last 20 Cuff Open Time change events.  Each logged event shall contain the following information:   * Timestamp – 4 bytes * Cuff Open Time value [seconds] – 4 bytes | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0006 | EAUS\_SYS\_xxxx\_xxxx |
| The CU shall provide a way to store the last 20 Voiding Volume change events.  Each logged event shall contain the following information:   * Timestamp – 4 bytes * Voiding Volume value [motor steps] – 4 bytes | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0008 | EAUS\_SYS\_xxxx\_xxxx |
| The CU shall provide a way to store the last 20 Baseline Volume change events.  Each logged event shall contain the following information:   * Timestamp – 4 bytes * Baseline Volume value [motor steps] – 4 bytes | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0009 | EAUS\_SYS\_xxxx\_xxxx |
| The CU shall provide a way to store the last 20 Lying-Down Volume change events.  Each logged event shall contain the following information:   * Timestamp – 4 bytes * Lying-Down Volume value [motor steps] – 4 bytes | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0010 | EAUS\_SYS\_xxxx\_xxxx |
| The CU shall provide a way to store the last 20 End-Of-Life threshold change events.  Each logged event shall contain the following information:   * Timestamp – 4 bytes * End-Of-Life threshold value [ADC value] – 4 bytes | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0011 | EAUS\_SYS\_xxxx\_xxxx |
| The CU shall provide a way to store the last 20 UroTimer detection events.  Each logged event shall contain the following information:   * Timestamp – 4 bytes | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0012 | EAUS\_SYS\_xxxx\_xxxx |
| The CU shall provide a way to store the last End-Of-Life detection event.  Each logged event shall contain the following information:   * Timestamp – 4 bytes * Battery level value [ADC value] – 4 bytes | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0013 | EAUS\_SYS\_xxxx\_xxxx |
| The CU shall provide a way to store for 5 years the daily count for the following PRC events [later referred to as “PRC events”]:   * Voiding Button press * Lying-down Button press * Baseline Button press | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0014 | EAUS\_SYS\_xxxx\_xxxx,  EAUS\_RSK\_0000\_0064,  EAUS\_SYS\_0000\_0234 |
| The CU shall provide a way to store the following data [later referred to as “periodic data”] before each pump movement:   * Timestamp – 4 bytes * Force Sensor [ADC raw value] – 4 bytes * Pressure [cmH2O] – 4 bytes * Battery level [mV] – 4 bytes * Motor position [motor steps] – 4 bytes * Last available PRC barometric data [UNIT] – 4 bytes * Identified cause of the pump movement [TBD: Voiding from PRC/Lying down/Baseline/Urotimer…] – 4 bytes | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0015 | EAUS\_SYS\_xxxx\_xxxx,  EAUS\_RSK\_0000\_0082 |
| The “periodic data” memory size to be allocated is 7300 data points (365 days of logging if 20 data points are collected each day).  When this memory range is full of data, the logs shall be rolling (earliest logs can be erased, and it is acceptable to lose at most 2048 bytes of data when rolling – erasing the oldest log points). | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0016 | EAUS\_SYS\_xxxx\_xxxx |
| The “PRC events” memory size to be allocated is 1825 data points (5 years of daily data).  When this memory range is full of data, the logs shall stop. | |

**AJOUTER UNE COMMANDE “SetCurrentPRCPressure” qui enregistre dans la CU la derniere valeur de pression atmospherique lue par la PRC.**

#### Debug Logs

|  |  |
| --- | --- |
|  |  |
| Upon an error event, the CU shall provide a way to store the context in which the event has happened as a debug log.  Each debug log shall contain the following information:   * Error ID – UINT64 * Force Sensor [ADC raw value] – UINT32 * Pressure [cmH2O] – UINT32 * Battery level [mV] – UINT32 * Last RF command received [see P04 – SRS CMD] – UINT32 * Overpressure occurrence – UINT16 * ERI occurrence – UINT8 * EOL occurrence – UINT8 * Implantation executed – UINT32 * Motor stall occurrence – UINT32 * Free text – 24 bytes | |

#### Cycle Logs

|  |  |
| --- | --- |
|  | Req6 |
| Before each pump stroke, the CU shall save in the flash the following information:   * Timestamp - UINT32 * Force Sensor [ADC raw value] – UINT32 * Pressure [cmH2O] – UINT32 * Battery level [mV] – UINT32 * Motor position – UINT32 | |

|  |  |
| --- | --- |
|  | Req6 |
| The CU shall be able to store at least 40000 cycles in the flash memory. | |

### Data Accessor

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_SYS\_0000\_0246 |
| The CU shall in first calibration: save all the data (motor position, force sensor value) in flash  Test:  read table in flash integrity 200pts ( valeurs cohérente ) | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0002 | EAUS\_SYS\_0000\_0108,  EAUS\_RSK\_0000\_0109,  EAUS\_RSK\_0000\_0175,  EAUS\_RSK\_0000\_0180,  EAUS\_RSK\_0000\_0261,  EAUS\_RSK\_0000\_0295,  EAUS\_RSK\_0000\_0304,  EAUS\_RSK\_0000\_0322,  EAUS\_RSK\_0000\_0428 |
| The UroTimer duration shall be stored in flash CU, and default value is 12 hours. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0003 | Need syspec |
| The CU shall save in flash:   * The overpressure detection warning * The number of times the pressure exceeds the overpressure threshold * The overpressure threshold * The overpressure number of occurrences | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_RSK\_0000\_0315,  EAUS\_RSK\_0000\_0084, |
| The CU shall add and verify the CRC at the end of this data ranges in Flash:   * Motor position * Technical parameters * Firmware * … | |

AJOUTER UNE SPEC ICI pour le risk 0438

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0002 | EAUS\_RSK\_0000\_0227 |
| The CU shall save the max motor position corresponding to 150 cmH2O | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0003 | EAUS\_RSK\_0000\_0240 |
| The CU shall save all critical configurations in no volatile memory:   * Technical parameters * Clinical parameters * Calibration pressure tables * Calibration link radio * Current implant stats | |

## Peripherals

### AlarmClock



|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_SYS\_0000\_0106,  EAUS\_RSK\_0000\_0163,  EAUS\_RSK\_0000\_0109,  EAUS\_RSK\_0000\_0175,  EAUS\_RSK\_0000\_0180,  EAUS\_RSK\_0000\_0261,  EAUS\_RSK\_0000\_0295,  EAUS\_RSK\_0000\_0304,  EAUS\_RSK\_0000\_0322,  EAUS\_RSK\_0000\_0428 |
| In Activated mode CU shall count the time without receiving Cuff deflation order. If this time is higher than UroTimer duration, the CU shall go to Deactivated mode.  Test:  goTo follow up, Set the Urotimer to default value (12 hours), read the value in flash and check if its 12 hours, come back to activated mode, and execute a voiding command, after exactely 12 hours check when the the CU goes to deactivated mode, the time shall be 12 hours +/- 1 minute | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0002 | EAUS\_SYS\_0000\_0106,  EAUS\_RSK\_0000\_0109,  EAUS\_RSK\_0000\_0175,  EAUS\_RSK\_0000\_0180,  EAUS\_RSK\_0000\_0261,  EAUS\_RSK\_0000\_0295,  EAUS\_RSK\_0000\_0304,  EAUS\_RSK\_0000\_0322,  EAUS\_RSK\_0000\_0428 |
| The accuracy of UroTimer shall be +/- 1 minute. | |



|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0003 | EAUS\_SYS\_0000\_0249, EAUS\_SYS\_0000\_0250 |
| The accuracy of voiding period shall be +/- 1 seconds.  Test:  go to activate, set the voiding period to default one (75 seconds), read the voiding period and check if its the same that we already send, send a voiding order wait the exactly the same period of time, read the motor position multiple times untle the returned position is baseline. check if the time is the same in a chronometer +/- 1 seconds | |

## Safety

### UroTimer

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_xxxx | EAUS\_RSK\_0000\_0163 |
| UroTimer shall always be active, the CU cannot deactivate safety features. | |

### OverPressure

### Motor Stall

The CU shall detect a motor stall using hall-effect sensors.

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_RSK\_0000\_0290,  EAUS\_RSK\_0000\_0281,  EAUS\_RSK\_0000\_0285 |
| If CU drives the motor and the requested position is not reached, the CU shall detect a stall and try a second time. If the motor is still stalled the CU shall raise an error. | |

|  |  |
| --- | --- |
|  |  |
| The motor stall occurrences shall be logged in the Debug logs. This value shall be cleared once the motor is driven to the requested position successfully. | |

|  |  |
| --- | --- |
|  |  |
| If a motor stall error has been raised in Clinical firmware, the CU shall provide a way to retry the operation by configuring the time between each try and the total number of retries. | |

### Error Management

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 |  |
| In any mode except [**safe**] and [**deactivated**] mode, if no activity is detected for 5 seconds and the running software is not in waiting state the CPU shall be put in standby mode. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0002 |  |
| In [**follow-up**], [**maintenance**] and [**implantation**] modes, the CPU shall be wake-up if radio alarm or RTC interrupt occur. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0003 |  |
| In [**activated**] mode, the CPU shall be wake-up if radio alarm or RTC interrupt occur. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0004 |  |
| In [**deactivated**] modes, after all necessary actions specific to the mode have been done, the running software shall put the core in shutdown mode with Radio interrupt as wake-up condition. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0005 |  |
| In any mode, before entering standby mode the running software shall set the RTC alarm interruption to trigger on the earliest timeout delay applicable. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 |  |
| The software shall be able to automatically reset the RTC counter after 1 year in order to prevent counter overflow. When reset is performed, it shall save the current counter value in order to have the total time elapsed since power-up. | |
| Note : 1 year is half the maximum time of RTC. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0001 | EAUS\_RSK\_0000\_0282 |
| Upon reading the hall-effect sensors output, if the returned values are not correct (all three sensors values are HIGH or all three sensors values are LOW), the CU shall go to Safe Mode. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0003 | EAUS\_RSK\_0000\_0273,  EAUS\_RSK\_0000\_0322 |
| Upon reading an out-of-bounds battery level value from the ADC, the CU shall go to Safe Mode. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0004 | EAUS\_SYS\_0000\_0232\_2,  EAUS\_RSK\_0000\_0189,  EAUS\_RSK\_0000\_0040,  EAUS\_RSK\_0000\_0228,  EAUS\_RSK\_0000\_0241,  EAUS\_RSK\_0000\_0271,  EAUS\_RSK\_0000\_0427,  EAUS\_RSK\_0000\_0438 |
| Upon an unexpected reset (watchdog triggered or MCU exception), the CU shall go to Safe mode. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0005 | EAUS\_RSK\_0000\_0051,  EAUS\_RSK\_0000\_0271,  EAUS\_RSK\_0000\_0297,  EAUS\_RSK\_0000\_0392 |
| If an error is found during the autotest of the MCU, the CU shall go to Safe Mode. | |

|  |  |
| --- | --- |
| EAUS\_SRS\_1610171143\_0006 | EAUS\_RSK\_0000\_0189,  EAUS\_RSK\_0000\_0315 |
| If the Applicative firmware (Production FW or Clinical FW) integrity is not guaranteed, the CU shall go to Safe Mode. | |

## Usability and human factors engineering

The requirements here may have traceability with the results of IEC 62366 standard implementation.

**Visas**

|  |  |  |  |
| --- | --- | --- | --- |
| **Signatures represent review and approval of the indicated revision of the below document** | | | |
| Function | Name | Signature | Date  (DD MMM YYYY) |
| QARA representative |  |  |  |
| Software Project Manager |  |  |  |

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Description | Author | Effective Date  (DD MMM YYYY) |
| A | Creation | V. Beslouin | 15 May 2019 |