7 RTU FUNCTIONAL AND OPERABILITY REQUIREMENTS

7.1 Introduction

The RTU can contain a large number of mission specific functions and therefore some parts of the specification are optional for a given mission. There are however a set of general requirements that shall be respected to ensure maximum compatibility across missions and spacecraft architectures. If an optional requirement/function is implemented, all the associated requirements shall be fulfilled unless otherwise noted.

Therefore, the requirements are sometimes tagged as an *option*, or include some *optional* features. An option is a functionality that is not mandatory across all missions but that can be required in some missions. A RTU supplier might decide to include or exclude the functionality in a standard product. Thus, requiring an optional functionality for a specific mission could prevent re-use of a product from another mission.

The general requirements are mainly related to the *RTU Controller* and the *RTU Remote Control Interface* (also referred to as the *RTU Core*) since these provide the main operational interface to the spacecraft. It shall be noted that the *RTU Controller* functionality could also be distributed inside the RTU. However, in general there is need for a central functional block that manages the overall RTU functionality.

7.2 Operating States & Self-test

7.2.1 Operating States

Requirement Number : SAVOIR.RTU.STATE.10
Operating States

The RTU shall implement at least the following operating states (see also Figure 3):

- *Off State*
- *Initialization State*
- *Operational State*
- *Test sub-state (a sub-state of Operational State)*
- *Optional States (optional, Option OptionalStates)*

Requirement Rationale :
The mandatory operating states of the RTU (*Off*, *Initialization, Operational* and *Test*) are similar across missions and suppliers in order to maximise the reusability of the RTU design and the RTU operational concept. However, in case of mission specific needs, additional states could be added. The *Initialization* state is a temporary state with a limited duration (it can be considered a transition phase).

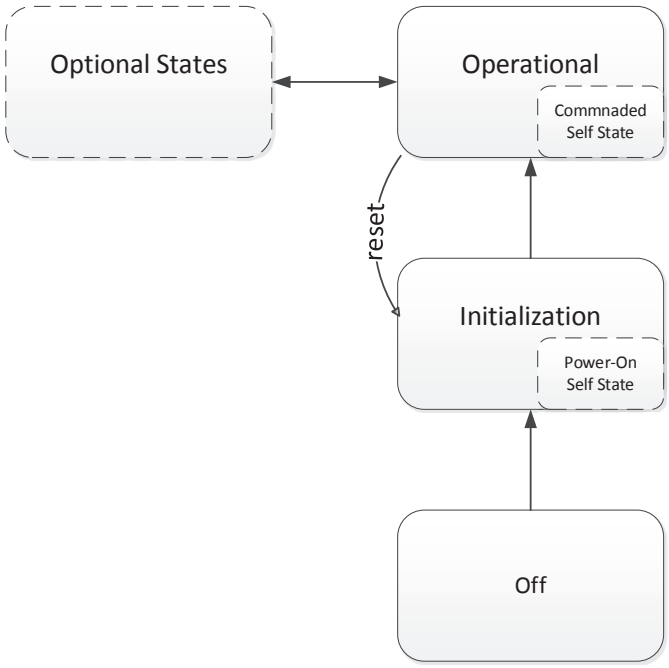

Figure 3 : RTU State Diagram

Requirement Number : SAVOIR.RTU.STATE.20
Off State
The *Off State* shall be characterised as follows:

- The RTU consumes no power (with the exclusion of the power monitoring and protection circuitry of the inlet if implemented)
- No commands via the *RTU Remote Control Interface* are processed
- No actuation of any interfaces

Requirement Number : SAVOIR.RTU.STATE.30
Initialization State
The *Initialization State* shall be characterised by the following:

- The state is entered by RTU power-on or by a reset signal.
- One redundancy of the *RTU Core* is activated.
- No commands via the *RTU Remote Control Interface* are processed.
- All configurable settings of the RTU are initialized to the default values.
- All memory areas related to programmable command or actuation sequences are cleared.
- No actuation of any interfaces
- The overall (redundancy) configuration of the RTU is set according to the default values.

- The RTU (optionally) executes a self-test as defined in 7.2.2.

Requirement Rationale :
The RTU can have configurable settings that allows pre-defined *RTU Standard User Interface* to be automatically powered on during initialization, e.g. to cope with autonomous recovery after an abnormal power loss (e.g. Solar Array Drive mechanism). Nominally, the *RTU Specific User Interface* are left in an off state until dedicated on commands have been received.

Requirement Number : SAVOIR.RTU.STATE.40
Requirement deleted.

Requirement Number : SAVOIR.RTU.STATE.50
Operational state
The *Operational state* shall be characterised by the following:

- Either nominal or redundant *RTU Core* is powered
- Commands via the *RTU Remote Control Interface* are processed, distributed and executed inside the RTU
- RTU configuration can be monitored and changed via the *RTU Remote Control Interface*.

Requirement Rationale :
The minimum meaningful configuration of the RTU in Operational state is with either nominal or redundant Power Supply and RTU CORE powered on.

Requirement Number : SAVOIR.RTU.STATE.60
Requirement deleted

Requirement Number : SAVOIR.RTU.STATE.70
Commanded Self-Test State
The *Commanded Self-Test* state (sub-state of the Operational State) shall allow full performance and nominal operation of the RTU to be maintained without impact, with the addition that it shall be possible to perform test of RTU functions and interfaces that are not actively used for the nominal operation.
Requirement Rationale :
Nominal operation is usually always required to ensure the spacecraft operations and autonomy. However, there might be situations when it is desirable to test the non-operational functions and interfaces , e.g. to perform off-line failure investigation or to prepare for a phase of hot redundant operation of specific modules. A specific case of the *Test* state is the operation of the redundant *RTU Core* in a stand-by configuration for test purposes..
OptionInfo: Option Commanded Self-Test State=Yes

Requirement Number : SAVOIR.RTU.STATE.80
Configuration State (Option)
Requirement deleted.

Requirement Number : SAVOIR.RTU.STATE.90
Optional States
Mission or RTU supplier specific states are allowed, but if implemented, they shall be implemented as transitions from and to the *Operational State*, i.e. there shall be no additional states introduced in the state transitions between *Off* => *Initialization* => *Operational*.
Requirement Rationale :
By requiring first a transition to *Operational state*, there is a minimum set of states and state transitions that are common to all RTUs, i,e, *Off* => *Initialization* => *Operational*. This allows any RTU to be switched on and to be commanded to start operating (with default settings) with a minimum of commands and ensures interoperability.
An example of optional state is a state where SW upload and patch is performed
OptionInfo: Option OptionalStates=Yes

Requirement Number : SAVOIR.RTU.STATE.100
State transitions; Outputs
No spurious commands or spikes shall be generated on any output interfaces under the transient conditions of power-on, power-off or by application or removal of primary power as well as during any state transitions.

Requirement Number : SAVOIR.RTU.STATE.110
Requirement deleted

Requirement Number : SAVOIR.RTU.STATE.120
User Interface module reset type
Requirement deleted. Covered by SAVOIR.RTU.STATE.11

7.2.2 RTU Power-On Self & Commanded-Self-Test

Requirement Number : SAVOIR.RTU.STATE.130
Power-on Self-TestSelf-Test
The RTU shall provide a self-test capability that is started automatically at power-on.
OptionInfo : Option Power-On Self-Test =Yes

Requirement Number : SAVOIR.RTU.STATE.140
Power-on Self-TestSelf-Test; Coverage
The RTU power-on Self-Test shall perform a test of at least the *RTU Controller* and the *RTU Remote Control Interface*.
Requirement Rationale :
The *RTU Controller* and the *RTU Remote Control Interface* are the basic elements needed for communication and operation of the RTU. The priority is therefore put on these items for the Power-on Self-test. The full RTU Self-test (Commanded Self-test) can then be commanded to run when the RTU has reached the Operational mode. The test coverage requirement for the power-on self-test is specified in SAVOIR.RTU.STATE.180.
OptionInfo : Option Power-On Self-Test =Yes

Requirement Number : SAVOIR.RTU.STATE.150
Commanded-Self-Test;
The RTU shall provide a Self-Test(s) that can be started by command and shall allow a test of the entire RTU, also including the tests run during Power-on Self-Test except for those that will prevent the RTU functioning (e.g. complete memory test).
Requirement Rationale:
The commanded Self-Test can be split into different tests, where each test is individually commandable, and the sum of each test comprehensively covers the entire RTU. The test coverage requirement for the commanded Self-Test is specified in SAVOIR.RTU.STATE.180. It is intentionally not specified in which state the RTU is running the self -tests since this can be implementation specific. Possibly some tests can be run in the *Operational State*, while other tests might be more suitable to be run in a dedicated *Test State*.
OptionInfo : Option Commanded Self-Test =Yes

Requirement Number : SAVOIR.RTU.STATE.160
Self-Tests Results
All Self-Test results (Power-on Self and Commanded-Self-Test) shall be stored in the RTU allowing to request them by a dedicated command via the *RTU Remote Control Interface* .
OptionInfo : Option Power-On Self-Test or Commanded Self-Test =Yes

---

Requirement Number : SAVOIR.RTU.STATE.170
Self-tests; Actuation
Self-tests shall have no effects on Standard and Specific User Interfaces (input and output) and no disturbance on the nominal operation of RTUSelf-Test
Requirement Rationale:
The self-test must not affect any RTU external units.
OptionInfo : Option Power-On Self-Test or Commanded Self-Test =Yes

Requirement Number : SAVOIR.RTU.STATE.180
Self-tests; Detailed coverage
The self-tests (Power-On Self Test and Commanded-Self-Test) coverage shall be agreed with the mission customer
Requirement Rationale:
The scope of the self-test and the content of the report depend on the specific RTU design and cannot be specified in detail. Hereafter a recommendation for the coverage of the Power-On Self and Commanded-Self-Test is provided:

- Power-on Self-Test & Commanded-Selftest:
  - Test of *RTU Controller*
  - Test of *RTU Remote Control Interface*
  - Test/monitoring of the RTU internal voltages
  - Verification of the RTU configuration status
  - Test of internal memories and associated error detection mechanisms
  - Test of any internal RTU communication bus
- Commanded-Self-Test:
  - Test of *RTU Standard User Interface* modules
  - Test of *RTU Specific User Interface* modules

In any case, the self-tests is supposed to cover all major and critical functions of the RTU.
OptionInfo : Option Power-On Self-Test or Commanded Self-Test =Yes

---

Requirement Number : SAVOIR.RTU.STATE.190
Self-Test;
The RTU supplier shall clearly identify and justify any parts of the RTU that are not covered by the self-test.
Requirement Rationale:
Certain RTU functions cannot be tested, e.g. overvoltage and overcurrent protections. Nevertheless, it is expected that untestable functions are clearly identified along with an associated justification.
OptionInfo : Option Power-On Self-Test or Commanded Self-Test =Yes

---

Requirement Number : SAVOIR.RTU.STATE.201
Self-Test:
Power-On Self-Test shall not block the transition to Operational State
OptionInfo : Option Power-On Self-Test =Yes

Requirement Number : SAVOIR.RTU.STATE.212
Internal Telemetry:
The RTU shall provide sufficient telemetry to detect and isolate any internal operational malfunction as identified in the FMECA

7.2.3 Performance Requirements

Requirement Number : SAVOIR.RTU.STATE.220
Start-up time
The RTU shall change from entry of *Initialization State* (power-on/reset command) to *Operational State* within <InitTime> seconds.
Requirement Rationale :
The RTU enters the Initialization State upon receipt of an RTU Power-on command or after a reset condition of the RTU Controller function. When the initialization is completed, the RTU changes autonomously from Initialization State to Operational State.
Typical <InitTime> value is 100ms-1s

Requirement Number : SAVOIR.RTU.STATE.230
Power-on Self-test; Test coverage
Requirement deleted in favour of STATE.18

Requirement Number : SAVOIR.RTU.STATE.240
Commanded Self-test; Test coverage
Requirement deleted in favour of STATE.180

7.3 Telemetry Acquisition & Observability

7.3.1 General Overview

The RTU telemetry acquisition concept is based on cyclic acquisition of predefined sequences of RTU input channels by means of one or more *telemetry acquisition lists*. The sequence and frequency of the telemetry acquisition is controllable by means of commands via the RTU Remote Control Interface. Composition of *telemetry acquisition lists* in term of RTU input channels is stored in the RTU non volatile memory as "default" *telemetry acquisition lists*.

The start of the acquisition of a group of channels defined in a *telemetry acquisition list* is triggered by a corresponding "Start TM Acquisition" event. This event can be a command or a discrete pulse and is typically provided to the RTU at a rate of 1 Hz. As a minimum, generation of a "Start TM Acquisition" event by means of a command is always possible.

A 1 Hz update rate is generally sufficient for housekeeping telemetry such as temperatures, voltages and status bits. However, it might be necessary to increase the sampling rate of selected parameters during operation and some parameters might need to have a constantly higher acquisition rate.

Therefore the RTU shall support several “Start TM Acquisition” events where each event would trigger the acquisition of a corresponding predefined sequence of telemetry channels (*telemetry acquisition list)*. This allows to independently and dynamically control the TM acquisition frequency for each pre-defined acquisition sequence.

Optionally, the RTU can also provide the necessary commands and associated internal control functionality to allow the RTU to be configured to cyclically acquire different sequences of input channels with different acquisition rates based on a single externally provided “Start TM Acquisition” trigger event.

It is essential that the acquisition time of each parameter can be determined. The acquisition process therefore needs to be deterministic, with known delays.

The current version of this specification assumes that the RTU does not have a Local On-board Time (LOBT) which is synchronised to the Central On-board time (COBT) of the Spacecraft. Therefore, all time stamping and timing references are provided externally to the RTU. For future revisions, a synchronised LOBT concept could be considered. Such scheme would allow the RTU to operate more autonomously and to directly time stamp the acquired telemetry.
It is also essential to be able to verify that the telemetry acquisition process is operating nominally.

7.3.2 Telemetry Acquisition - General

Requirement Number : SAVOIR.RTU.TM.10
Telemetry Acquisition; General
The RTU shall provide the capability to perform cyclic acquisition of telemetry from RTU internal housekeeping parameters and from the *Standard* and *Specific User Interfaces*
Requirement Rationale :
Examples of internal housekeeping and telemetry are : analogue voltages, thermistor values, relay status signals, AOCS sensor data etc.

Requirement Number : SAVOIR.RTU.TM.20
Telemetry Acquisition; General
The RTU shall be compliant to the Telemetry requirements as defined in Section 4.1.4 of [ECSS-E-20].

Requirement Number : SAVOIR.RTU.TM.30
Telemetry Acquisition; General RTU Observability
The RTU shall provide sufficient telemetry to allow the exact state of the RTU to be determined in Operational State. This includes in particular:

- the operating state

- on/off status of each switchable module.
- the status of any ongoing or pending command execution/actuation
- the status of any automatic protection function such as current limiter and overvoltage protections.
- the status of any commandable protection function (inhibit/disabled/enabled/armed/disarmed)
- all stored status information (volatile and non-volatile)
- internal supply voltages
- all secondary supply voltages provided to RTU external users
- RTU internal temperatures
- failure status
- autonomously taken actions
- information if the command(s) could not be executed or if the command has only been partially executed, e.g. due to a wrong operational state of the commanded module.
- reporting of bit error detection and correction and memory scrubbing status

---

Requirement Number : SAVOIR.RTU.TM.40
Telemetry Acquisition; Lists
The RTU shall sequentially acquire telemetry according to pre-defined *telemetry acquisition lists* stored in the RTU.
Requirement Rationale :
Organising the telemetry acquisition in lists simplifies the operation and optimises the transmission of acquired data since all the telemetry associated to each list can be pre-collected in the RTU before the entire list is transmitted via the *RTU Remote Control Interface*.

Requirement Number : SAVOIR.RTU.TM.50
Telemetry Acquisition; Data buffer and transmission
The data resulting from the acquisition of a *telemetry acquisition list* shall be buffered in the RTU and made available via the *RTU Remote Control Interface* in a contiguous sequence.
Requirement Rationale :
The precise format is dependent on the implemented communication bus, e.g. Mil-Std-1553B, CAN or SpW etc. However, it is essential that all data are transmitted without gaps in order not to waste communication bus bandwidth. The acquired data can be provided autonomously on the comms bus by the RTU or the data can be transmitted in response to a dedicated command. The specific approach used depends on the type of communication bus that is used by the RTU.

Requirement Number : SAVOIR.RTU.TM.60
Telemetry Acquisition; Telemetry readout
Requirement deleted

Requirement Number : SAVOIR.RTU.TM.70
Telemetry Acquisition; Starting telemetry acquisition
The RTU shall start and maintain the cyclic acquisition of each specific *telemetry acquisition list* after a corresponding "Start TM Acquisition" event has been detected.
Requirement Rationale :
Typical "Start TM Acquisition" event(s)[2] could be:

- Dedicated command via the *RTU Remote Control Interface*.
- 1553 Communication Frame start according to [Mil]
- Dedicated synchronisation signal transmitted over the *RTU Remote Control Interface* e.g. Synchronisation mode code with a 1553 bus or timecode with SpaceWire.
- External trigger pulse, e.g. Pulse Per Second (PPS)
- CAN SYNC message according to [CAN]
- RTU internal timer[1]

Requirement Number : SAVOIR.RTU.TM.80
Telemetry Acquisition; Commanded "Start TM Acquisition" event
The RTU shall be able to receive via the *RTU Remote Control Interface,* commanded "Start TM Acquisition" event(s)[2] for each of the *telemetry acquisition lists* defined in the RTU.
Requirement Rationale :
It is essential to be able to control and trigger the telemetry acquisition by means of commands to the RTU regardless of the nominal method of telemetry acquisition triggering. This requirement does thus not prescribe or restrict the methods for generation of the 'Start TM Acquisition" events with the exception that commanded triggering shall always be possible.

Requirement Number : SAVOIR.RTU.TM.90
Telemetry Acquisition; List Sequence

[1] It must be possible to know the times of telemetry acquisition. Therefore, if an internal timer is used it needs to be referenced/correlated to a known event or need to be synchronized to an external time reference. An example of an implementation is to use a PPS event to reset an internal timer every second. The internal timer is then used to trigger telemetry acquisitions according to different *telemetry acquisition lists* at defined times within the one second period.
[2] The triggering of each TM acquisition list is only done by one dedicated trigger event. However, there is nothing that prevents an implementation that allows different trigger events for a given TM list as long as these are not active concurrently, i.e. a TM acquisition cycle only uses one event type for that TM list. There could be, concurrently, different trigger events for different lists.

The *telemetry acquisition lists* shall be triggered in the same sequence as defined by the sequence of the corresponding “Start TM Acquisition” events.
Requirement Rationale :
The acquisition time of each *telemetry acquisition list* needs to be well known and deterministic.

Requirement Number : SAVOIR.RTU.TM.100
Telemetry Acquisition; Acquisition sequence
The RTU shall acquire each telemetry channel in a given *telemetry acquisition list* in the same sequence as they are defined in the *telemetry acquisition list*.
Requirement Rationale :
The acquisition time of each telemetry parameter needs to be well known and deterministic.

Requirement Number : SAVOIR.RTU.TM.110
Telemetry Acquisition; Determinism
The execution time of each RTU internal action taken in response to a “Start TM Acquisition” event shall be known and deterministic.
Requirement Rationale :
The associated performance requirement is defined inSAVOIR.RTU.TM.34 .

Requirement Number : SAVOIR.RTU.TM.120
Telemetry Acquisition; Telemetry override
The case of occurrence of a Start TM Acquisition event while an acquisition process is on-going shall be treated in a deterministic way, either by stopping the on-going process or by ignoring the new event.*Requirement Rationale* :
The case of occurrence of a Start TM Acquisition event while an acquisition process already is on-going has to be regarded as abnormal. The choice to stop the process and start a new one is not at all obvious. In general it would be easier to simply ignore the new command and such a behaviour may even be preferable for many users. .

Requirement Number : SAVOIR.RTU.TM.130
Telemetry Acquisition; Acquisition frequency
It shall be possible to control the acquisition frequency <TelAcqListFreq> for each *telemetry acquisition list* independently.
Requirement Rationale :
The acquisition frequency for a given *telemetry acquisition list* is defined by the frequency of the corresponding “Start TM Acquisition” event. Each telemetry channel within a *telemetry acquisition list* is acquired with the same frequency. Acquisition of temperatures can often be done at a lower rate (e.g. <TelAcqListFreq>=1Hz) than acquisition of voltages

(<TelAcqListFreq> = 10Hz or 100Hz) so these could be defined in separate lists. The actual acquisition frequencies are mission specific.

Requirement Number : SAVOIR.RTU.TM.140
Telemetry list;
Requirement deleted in favour of TM.34

Requirement Number : SAVOIR.RTU.TM.141
Telemetry Acquisition; Stopping telemetry acquisition
The RTU shall stop the cyclic acquisition of each specific *telemetry acquisition list* after a corresponding "Stop TM Acquisition" event has been detected.
Requirement Rationale :
A Typical "Stop TM Acquisition" event(s) could be a dedicated command via the *RTU Remote Control Interface*

7.3.3 Telemetry Acquisition - Validity

Requirement Number : SAVOIR.RTU.TM.150
Telemetry Acquisition; Acquisition validity
It shall at all times be possible to determine the validity status of each acquired telemetry parameter.
Requirement Rationale :
Since telemetry is usually acquired in a cyclic loop, there needs to be a possibility to verify that the telemetry acquisition process is working and that the telemetry parameters are actually being updated, for example implementing a counter and/or implementing internal reference channels to check the correctness of the RTU acquisition chain. A complete validity check of telemetry is normally a combination of RTU capabilities and software (currently implemented in the OBC).

Requirement Number : SAVOIR.RTU.TM.160
Telemetry Acquisition; Failure detection and reporting
The RTU shall provide internal failure detection mechanisms and associated telemetry to allow distinguishing between RTU internal failures and failures caused by a sensor external to the RTU for critical acquisitions.
Requirement Rationale :
It is essential to allow failure identification and isolation. As an example, in the case of analogue acquisitions, the lowest and highest addressable channel per group/multiplexer could be set to (different) fixed reference values. This could allow health check of both the multiplexer and the analogue to digital conversion parts. Of course, if there is a failure of an individual acquisition channel, it cannot immediately be determined if the problem is in the RTU or external, but the requirement concerns the overall acquisition chain.

Requirement Number : SAVOIR.RTU.TM.170
Telemetry Memory; Flush
Requirement deleted in favour of TM.15

Requirement Number : SAVOIR.RTU.TM.180
Telemetry Memory; Flush (other lists)
Requirement deleted in favour of TM.15

Requirement Number : SAVOIR.RTU.TM.190
Telemetry Validity;
The RTU shall provide means to allow a unique correlation between the acquired telemetry data and the corresponding “Start TM Acquisition” event.
Requirement Rationale :
This is to support the determination of the telemetry acquisition time.

Requirement Number : SAVOIR.RTU.TM.200
Telemetry Validity; Token value
Requirement deleted in favour of TM.19

Requirement Number : SAVOIR.RTU.TM.210
Telemetry Validity; Token reply
Requirement deleted in favour of TM.19

Requirement Number : SAVOIR.RTU.TM.220
Telemetry Validity; Token position
Requirement deleted in favour of TM.19

Requirement Number : SAVOIR.RTU.TM.230
Telemetry Validity; Internal event; Token init (Option)
Requirement deleted in favour of TM.19

Requirement Number : SAVOIR.RTU.TM.240
Telemetry Validity; Internal event; Token init (Option)
Requirement deleted in favour of TM.19

Requirement Number : SAVOIR.RTU.TM.250
Telemetry Validity; Internal event; Token increment (Option)
Requirement deleted in favour of TM.19

Requirement Number : SAVOIR.RTU.TM.260
Telemetry Validity; Internal event; Token wrap-around (Option)
Requirement deleted in favour of TM.19

7.3.4 Telemetry Acquisition - List Management

Requirement Number : SAVOIR.RTU.TM.270
Telemetry acquisition list; Default settings
Default *telemetry acquisition lists* should be configurable also in orbit.
Requirement Rationale :
The default values are normally stored stored in non-volatilememories/register. It is however recommended to provide the possibility to change the default values also after equipment manufacturing and also on orbit, e.g. by means of dedicated commands, or via a dedicated service interface.

Requirement Number : SAVOIR.RTU.TM.280
Telemetry acquisition list; Initialization
During *Initialization State*, the RTU shall initialize the *telemetry acquisition lists* to the corresponding default values.

Requirement Number : SAVOIR.RTU.TM.290
Telemetry acquisition list; Change of contents
It should be possible to change/reprogram *telemetry acquisition list* by means of dedicated commands.
Requirement Rationale :
If a telemetry acquisition list is small enough and only contains channels from the same failure group, changing the content of the list itself might not be needed. A change of telemetry acquisition sequence could then be done by rearranging the “Start TM Acquisition” events. However, as a minimum if a list contains acquisition of parameters from multiple failure groups, it is necessary to be able to redefine the list contents to avoid acquiring data from a failed failure group.

7.3.5 Telemetry Acquisition - Performance

Requirement Number : SAVOIR.RTU.TM.300
Telemetry Acquisition; Number of commanded “Start TM Acquisition” events
The RTU shall be able to receive messages via the *RTU Remote Control Interface* which each can contain up to <StartTMAcqNumber> “Start TM Acquisition” events.
Requirement Rationale :

The typical number of "Start TM Acquisition" events that can be included in the message to the RTU depends on the chosen communication bus and associated higher layer protocol. The current value of < StartTMAcqNumber > is 1

Requirement Number : SAVOIR.RTU.TM.310
Number of Telemetry Acquisition Lists;
The RTU shall provide <TMAcqListNumber> different programmable *telemetry acquisition lists*.
Requirement Rationale :
The number of lists <TMAcqListNumber> is mission dependant, typical values are 4 – 16.

Requirement Number : SAVOIR.RTU.TM.320
Number of Telemetry Acquisition Channels per Telemetry Acquisition List;
Each list shall contain up to <TMChannelNumber> telemetry channels.
Requirement Rationale :
Typical values of <TMChannelNumber> are 32 - 64 -128. Each *telemetry acquisition list* might be of a different size.

Requirement Number : SAVOIR.RTU.TM.330
Telemetry acquisition determinism performance
Requirement deleted

Requirement Number : SAVOIR.RTU.TM.340
Telemetry acquisition time
The acquisition time of each telemetry parameter shall be deterministic with a maximum inaccuracy of <TMParamAcqTimeAcc> ms.
Requirement Rationale :
Acquisition time could be different for different type of acquisition type/signal, Typical <TMParamAcqTimeAcc> values is 1-5 ms

---

Requirement Number : SAVOIR.RTU.TM.350
Telemetry acquisition rate
The Telemetry acquisition rate <TMAcqRate> shall be defined and verified
Requirement Rationale :
The minimum value of the acquision rate <TMAcqRate> is 4000 -10000 samples/sec.

7.4 Commanding & Actuation

7.4.1 General Overview

Generally, there are three different categories of commands identified for the RTU.

- Commands for internal control and management of the RTU configuration.
- Commands for immediate control and actuation of *Standard* or *Specific User Interfaces*
- Commands for scheduling the actuation of *Standard* or *Specific User Interfaces* in relation to a dedicated reference event.

The commands related to internal control and management of the RTU can at any time be executed and generally do not affect any other ongoing telemetry acquisition or actuation processes (unless the command specifically concerns such function).

The commands for immediate execution are typically *Standard User Interface* pulse commands, used for controlling the on/off switching of units.

Actuation of AOCS actuators such as thrusters, magnetorquers etc. are however, usually done in relation to a well-defined reference event, typically corresponding to the AOCS cycle. In these cases the actuation commands and the associated relevant parameters (delay, pulse length etc.) are pre-loaded into the RTU. The execution of the command is then triggered by a dedicated event ("Start TC Actuation").

The RTU can support several "Start TC Actuation" events where each event would trigger the execution of a corresponding pre-loaded sequence of actuation commands. This allows to independently and dynamically control the actuation parameters for each pre-loaded actuation sequence.

It is essential that the actual actuation time of each user interface is known and deterministic with a known accuracy. It is also essential to be able to verify that the command actuation process is operating nominally. To support this, the RTU should provide the relevant telemetry to be able to monitor the progress of the actuations.

The "Start TC Actuation" event can be implemented in different ways, e.g. by means of a command via the *Remote Control Interface,* a discrete pulse such as the PPS or an RTU internal timer which is synchronised to an external well known time reference. As a minimum, the generation of the "Start TC Actuation" event by means of command should be possible.

7.4.2 Commanding & Actuation - General

Requirement Number : SAVOIR.RTU.CMD.10
Command Execution

The RTU shall decode, validate and process any received command immediately after the receipt of the command.
Requirement Rationale :
However, this does not imply that the RTU immediately actuates interfaces. Some commands will only be used to pre-load actuation sequence tables for later execution in relation to the “Start TC Actuation” event.

Requirement Number : SAVOIR.RTU.CMD.20
Command Execution; Acknowledgement and Status
The RTU shall provide sufficient means to be able to verify the command reception, acknowledgement and execution status for each received command.
OptionInfo : Option CmdAck=Yes
Requirement Rationale :
The command reception, acknowledgement and execution is optional mission and command dependent

---

Requirement Number : SAVOIR.RTU.CMD.30
Commanding; Addressing
The command interface and telemetry interface addresses for accessing the RTU internal modules shall be identical when accessing a module through nominal or redundant *RTU Core*.

Requirement Number : SAVOIR.RTU.CMD.40
Command Sequence
Commands shall be executed in the same sequence as they are provided to the RTU.
Requirement Rationale :
It is essential to have a clear sequence of command execution and any pre-loading of tables shall be done in the same sequence as defined by the received commands.

Requirement Number : SAVOIR.RTU.CMD.50
Commanding; Actuation lists
It shall be possible to pre-load a sequence of actuation commands in dedicated *command actuation list*s in the RTU for later execution in response to a corresponding “Start TC Actuation” event.
Requirement Rationale :
As for Telemetry list also commands could be grouped in list(s) and a single event (“Start TC Actuation” event) can start the execution of them.
OptionInfo : Option ActList=Yes

Requirement Number : SAVOIR.RTU.CMD.60
Command Actuation; Starting actuation
The RTU shall start the actuation according to each specific *command actuation list* after a corresponding "Start TC Actuation" event has been detected.
Requirement Rationale :
Typical "Start TC Actuation" events could be:

- Dedicated command via the *RTU Remote Control Interface*.
- 1553 Communication Frame start according to [Mil]
- Dedicated synchronisation signal transmitted over the RTU Remote Control Interface e.g. Synchronisation mode code with a 1553 bus or timecode with SpaceWire.
- External trigger pulse, e.g. Pulse Per Second (PPS)
- CAN SYNC message according to [CAN]
- RTU internal timer[3]

The actuation frequency for a given command actuation list is defined by the frequency of the corresponding "Start TC Actuation" event. Each command channel within a command actuation list is actuated with the same frequency although individual commands can be delayed depending on parameter settings (see SAVOIR.RTU.CMD.7).
OptionInfo : Option ActList=Yes

Requirement Number : SAVOIR.RTU.CMD.70
Command Actuation; Actuation sequence
The RTU shall actuate each commanded user interface in a given *command actuation list* in the same sequence as they are defined in the *command actuation lists* or in the sequence defined by any associated command parameters (delay, pulse length etc.).
Requirement Rationale :
The execution sequence of the commands needs to be controllable, well known and deterministic.
OptionInfo : Option ActList=Yes

Requirement Number : SAVOIR.RTU.CMD.80
Command Actuation; Determinism
The execution time of each RTU internal action taken in response to a "Start TC Actuation" event shall be known and deterministic.
Requirement Rationale :
The associated performance requirement are defined in 7.4.4.

[3] It must be possible to precisely know the command actuation times. Therefore, if an internal timer is used it needs to be referenced/correlated to a known event or need to be synchronized to an external time reference. An example of an implementation is to use a PPS event to reset an internal timer every second. The internal timer is then used to trigger command actuations according to different *command actuation lists* at defined times within the one second period.

Requirement Number : SAVOIR.RTU.CMD.90
Command Actuation; Actuation frequency
Requirement deleted in favour of CMD.6

Requirement Number : SAVOIR.RTU.CMD.100
Operation; Concurrent operation
It shall be possible to issue commands and telemetry requests and acquire associated telemetry responses to/from any specific RTU Standard User Interface without affecting the operation of other interfaces
Requirement Rationale :
It must be possible to independently operate different interfaces in parallel.

7.4.3 Commanding & Actuation - Failure Protection

Requirement Number : SAVOIR.RTU.CMD.110
Command Execution; Output protection.
No single failure inside the RTU shall cause inadvertent (continuous) activation of pulse commands.
Requirement Rationale :
The inadvertent (and continuous) pulse commands activation has to be avoided due the possible critical consequences. An output interface can have two independent switches in series or a watchdog functionality can be implemented that disables the interface in case a pulse duration exceeds a pre-defined value. Definition of acceptable inadvertent pulse duration is left to the specific mission.

Requirement Number : SAVOIR.RTU.CMD.120
Command Execution; Two stage ("Arm & Fire")
The RTU shall support a two stage commanding capability ("Arm and Fire") for critical functions.
Requirement Rationale :
The definition of critical functions is mission specific. Nevertheless, functions as Propulsion drive Electronics usually require a two stage commanding scheme. The requirements related to two stage commanding below are generic and only defines the basic principles. The physical implementation of the logic can be specific for each *User Interface* module. Any further details are specified in the associated paragraphs (7.6.4 and 7.6.6).

Requirement Number : SAVOIR.RTU.CMD.130
Command Execution; Two stage sequence
Requirement deleted in favour of CMD.14

Requirement Number : SAVOIR.RTU.CMD.140
Command Execution; Two stage sequence; Power-on
Any actuator interface function that implements a two stage commanding scheme shall set the "Arm/Disarm" status to "Disarm(Inhibit)" at power-on of the respective functional module, in case of a RTU reset condition or in case of a time-out of the "Armed" state duration.

Requirement Number : SAVOIR.RTU.CMD.150
Actuator protection mechanisms
In case of loss of communication with the OBC, loss of the *RTU Controller* functionality or the *RTU Controller* is not powered anymore, commanding of critical attitude control actuators shall become inactive within a known and predefined time.
Requirement Rationale :
As a general rule in case of loss of communication with the OBC, loss of the *RTU Controller* functionality or the *RTU Controller*, critical actuators (e.g. propulsion) stop automatically after a defined time and a new command to the RTU are needed to start actuation again. This is essential to prevent that the RTU itself starts to activate, e.g. flow-control valves that could cause the S/C to start spinning. A Solar Array Drive Electronics (SADE) module can continue to rotate the Solar Array as commanded in order to optimise the sun illumination angle. The details are mission specific.

Requirement Number : SAVOIR.RTU.CMD.160
Command protection mechanisms
The RTU shall provide error protection and/or detection functions for the complete command chain from the *RTU Remote Control Interface* up to the *User Interface* output ports in order to avoid:

- sending of wrong serial data or discrete commands,
- erroneous actuation or powering of connected units,

Requirement Rationale :
It must be ensured that the probability for an incorrect actuation of an RTU output is kept extremely low since such failure could have severe mission impact.

7.4.4 Commanding & Actuation - Performance

Requirement Number : SAVOIR.RTU.CMD.170
Requirement deleted in favour of CMD.20

Requirement Number : SAVOIR.RTU.CMD.180
Command Execution; Determinism
The maximum execution time of each RTU internal action taken in response to an external command shall be known and deterministic.
Requirement Rationale :
This is essential to allow proper scheduling of the commanding or actuation.

Requirement Number : SAVOIR.RTU.CMD.190
Command execution and interface actuation
For a given RTU implementation, the maximum actuation rate <ActRate> , and maximum number of simultaneously commandable channels <SimActChannel> for each type of interface shall be defined.
Requirement Rationale :
Typical value of the range <ActRate> is 10-1000 actuation/sec, for <SimActChannel> a typical value is 1-10.

Requirement Number : SAVOIR.RTU.CMD.200
Command Execution; Command determinism performance
The time of an RTU external interface actuation relative to the time of the associated "Start TC Actuation" event shall be deterministic within a range <ActExecTimeAcc> μs.
Requirement Rationale :
RTU internal timing needs to be known to allow proper scheduling of the commanding of the RTU. Typical value of the range <ActExecTimeAcc> is 100-1000μs .

Requirement Number : SAVOIR.RTU.CMD.210
Command Execution; Command and Control
Requirement deleted in favour of CMD.19.

7.5 Redundancy Requirements

This paragraph specifies a set of generic requirements for redundancy and failure isolation with particular focus on requirements that are specific to an RTU. There are possibly more specific requirements defined for each *RTU Standard User Interface* and *RTU Specific User Interface* modules in the corresponding sub-section in paragraphs 7.6.4 and 7.6.6.

Redundancy is usually implemented by means of full duplication of functional modules in combination with suitable usage of the principle of failure groups for similar interface channels. The failure group concept could be the preferred redundancy solution for interfaces that are implemented in a high multiplicity (e.g. pulse command interfaces or analogue acquisition interfaces).

Requirement Number : SAVOIR.RTU.RED.10
Redundancy; General Requirements
The RTU shall be compliant to the redundancy requirements as defined in [ECSS-E-20], Section 4.2.1.

Requirement Number : SAVOIR.RTU.RED.20
Redundancy; Concept
Requirement deleted. Comment added in beginning of 6.4.

Requirement Number : SAVOIR.RTU.RED.30
Redundancy; Failure group size
The size of each failure group shall be specified and justified for any RTU function for which failure groups redundancy concept is adopted.
Requirement Rationale :
The definition of an acceptable failure group size is mission dependent and a system task. The failure group size is usually specified and included in mission specific requirements specification with a wording per interface type like: “No single failure shall cause the loss of more than TBD interface type X.". In general, smaller failure group sizes could simplify the channel allocation and redundancy management at spacecraft level. For analogue acquisition channels, a typical failure group could consist of 8 or 16 channels, while for flow control valve interfaces a failure group could consist of a single channel.

Requirement Number : SAVOIR.RTU.RED.40
Redundancy; Failure group isolation
If the concept of failure groups is adopted for a group of interface channels, the channels shall be grouped such that no single failure internal to a group will cause a loss or a misbehaviour of more than one group.
Requirement Rationale :
There must per definition not be any common failure modes causing loss of multiple failure groups. There might be specific requirements concerning failure modes for failure groups defined for different *RTU Standard User Interface* types in paragraph 7.6.4.

Requirement Number : SAVOIR.RTU.RED.50
RTU Interfaces; Interface selection
Selection of the RTU internal redundancies shall be static, deterministic and fully controlled via the *RTU Remote Control Interface*.
Requirement Rationale :
This means that selection between nominal or redundant (internal and external) interface signals is performed by means of dedicated control/redundancy select signals.As a

consequence, implicit selection of the active redundancy by using OR-ing or similar techniques by monitoring the nominal and redundant interface signals at the receiving end shall be avoided. Such mechanisms can be very sensitive to disturbances or introduce undesirable failure modes.

Requirement Number : SAVOIR.RTU.RED.60
Internal Command Link Management
Requirement deleted

Requirement Number : SAVOIR.RTU.RED.70
RTU Core; Reconfiguration
Requirement deleted

Requirement Number : SAVOIR.RTU.RED.80
Redundancy; Combinations
The RTU should allow any On/Off combination and operation of any redundant function. Any specific constraints to be explicitly declared by the RTU supplier.
Requirement Rationale :
This is to allow using the full redundancy capabilities of the RTU without constraints.

Requirement Number : SAVOIR.RTU.RED.90
General Redundancy Management
Every RTU reconfiguration change shall be triggered by external commands, received through the RTU Remote Control Interface
Requirement Rationale :
RTU behaviour is deterministic and fully controlled by the OBC

Requirement Number : SAVOIR.RTU.RED.100
Interface Redundancy Management
Requirement deleted in favour of RED.9

Requirement Number : SAVOIR.RTU.RED.110
Interface Redundancy Management; States
Requirement deleted in favour of RED.9

Requirement Number : SAVOIR.RTU.RED.120
Interface Redundancy Management; States

RTU voltages and currents exceeding defined limits may cause an autonomous power off of a RTU function
Requirement Rationale :
No automatic reconfiguration of the RTU is possible with the exception that voltages and currents exceeding defined limits can cause an autonomous power off.

7.6 Module Requirements & Interfaces

7.6.1 General Overview

Detailed requirements concerning the implementation of the RTU is out of scope for this specification. However, the following paragraphs define some general requirements related to the *RTU Core* and the *User Interfaces*.

The term *RTU Core* is used as reference for the overall RTU control and monitoring authority combined with the associated *Remote Control Interface* towards the spacecraft. This does not imply a particular implementation. The overall RTU control and monitoring can also be distributed within the RTU, depending on the chosen architecture.

7.6.2 RTU Core

Requirement Number : SAVOIR.RTU.CORE.10
RTU Core Redundancy
The *RTU Core* modules shall nominally be operating in cold redundancy.

Requirement Number : SAVOIR.RTU.CORE.20
RTU Core; Standby operation
When one *RTU Core* is working in nominal conditions, it shall be possible to power on the redundant *RTU Core* for standby operation w/o affecting the operations of the nominal RTU core.
Requirement Rationale :
This allows for performing tests on the redundant RTU Core function. It is obvious that an RTU Core in stand-by operation must not interfere with the nominal operation. However, since there might be a dormant failure in the redundant core, it needs to be ensured that it is always safe to switch on the redundant *RTU Core* despite such potential failure. It shall be noted that there is no requirement that the redundant *RTU Core* needs to work, only that a failure must not cause damage to any other part of the RTU.
OptionInfo : Option RTU Core Stand-By=Yes

Requirement Number : SAVOIR.RTU.CORE.30
RTU Core; Operating state
Requirement deleted.

Requirement Number : SAVOIR.RTU.CORE.40
RTU Core; Standby operation impact
Requirement deleted and comment added to CORE.20.

Requirement Number : SAVOIR.RTU.CORE.50
Power-on Self-test; Redundant *RTU Core* module
When one *RTU Core* module is working in standby, it shall be possible to retrieve from the standby *RTU Core* module the power-on self-test results as defined in paragraph 7.2.2.
Note: Self-Test result is usually made available via the remote control interface
OptionInfo : Option Power-On Self-Test and RTU Core Stand-By =Yes

Requirement Number : SAVOIR.RTU.CORE.60
Commanded Self-test; Redundant *RTU Core* modules
When one *RTU Core* module is working in standby, it shall be possible to perform a commanded self-test of the standby *RTU Core* module as defined in paragraph 7.2.2 and to retrieve the corresponding results.
Requirement Rationale:
Consequently, in this case the commanded self-test is limited to the standby *RTU Core* function and does not allow testing any other function within the RTU.
OptionInfo : Option Commanded Self-Test RTU Core Stand-By =Yes

7.6.3 RTU Remote Control Interface

Requirement Number : SAVOIR.RTU.CIF.10
RTU Controller Command Interface
Requirement deleted

Requirement Number : SAVOIR.RTU.CIF.20
RTU Remote Control Interface; Types
The *RTU Remote Control Interface* shall provide at least one of the following interfaces:

- Mil-Std-1553B (see paragraph 7.6.3.1)
- CAN (see paragraph 7.6.3.2)
- Spacewire (see paragraph 7.6.3.3)

Requirement Rationale :
A single interface standard shall be selected for a given mission. It is however recommended that the design of the RTU is made modular such that any of the above interfaces can be easily selected and implemented.

Requirement Number : SAVOIR.RTU.CIF.30
RTU Remote Control Interface; Specification
The *RTU Remote Control Interface* shall be compliant to the requirements defined in the corresponding paragraph (7.6.3.1, 7.6.3.2, or 7.6.3.3).

7.6.3.1 Mil-Std-1553

In case the Mil-Std-1553B is used for the RTU Remote Control Interface the following requirements apply:

Requirement Number : SAVOIR.RTU.CIF.40
Command and Control link; 1553
The *RTU Remote Control Interface* shall be compliant to [MIL].
OptionInfo : Option 1553=Yes

Requirement Number : SAVOIR.RTU.CIF.50
Command and Control link; 1553 Validation
The compliance of the *RTU Remote Control Interface* shall be demonstrated by test as defined in [MIL], paragraph 9.3.
OptionInfo : Option 1553=Yes

7.6.3.2 CAN

In case the CAN bus is used for the *RTU Remote Control Interface* the following requirements apply:

Requirement Number : SAVOIR.RTU.CIF.60
Command and Control link; CAN
The *RTU Remote Control Interface* shall be compliant to [CAN].
OptionInfo : Option CAN=Yes

Requirement Number : SAVOIR.RTU.CIF.70
Command and Control link; CAN Validation
Requirement deleted

7.6.3.3 Spacewire

In case Spacewire is used for the *RTU Remote Control Interface* the following requirements apply:

Requirement Number : SAVOIR.RTU.CIF.80

Command and Control link; Spacewire
The *RTU Remote Control Interface* shall be compliant to [SPW]
OptionInfo : Option SpW=Yes

Requirement Number : SAVOIR.RTU.CIF.90
Command and Control link; Spacewire
The *RTU Remote Control Interface* should be compliant to [SPW_RMAP].
Requirement Rationale :
Use of RMAP protocol for a RTU is considered the most straightforward solution.
OptionInfo : Option SpW=Yes

Requirement Number : SAVOIR.RTU.CIF.100
Requirement deleted

7.6.4 Standard User Interfaces

Requirement Number : SAVOIR.RTU.UIF.10
Interfaces; General
The RTU shall be compliant to the Interface requirements as defined in Section 4 of [ECSS-E-20].

Requirement Number : SAVOIR.RTU.UIF.20
RTU Standard User Interfaces
The RTU shall include at least one or several of the following standard user interfaces and associated interface functionalities as specified further in the referenced paragraph:

1. Analogue signal (par 7.6.4.1)
2. Bi-Level Discrete Input (par 7.6.4.2)
3. Pulse Command Interface (par 7.6.4.3)
4. Serial Digital Interfaces (par 7.6.4.4)
5. Synchronization Signal (In/Out) [4]

Requirement Rationale :
The number and the type of interfaces implemented in the RTU is mission dependent. It is also possible that more than one RTU is used on board a Spacecraft. It is therefore impossible to define precisely the interface configurations. Nevertheless, virtually all Spacecraft implement on-board a selection of standard discrete interfaces.

[4] RTU can be used to distribute additional synch signals to the Spacecraft, therefore input signal(s) arrive from the OBC and are distributed by the RTU. Standard Balanced Digital link (SBDL) are typically used

Requirement Number : SAVOIR.RTU.UIF.30
RTU Standard User Interfaces
Requirement deleted

Requirement Number : SAVOIR.RTU.UIF.40
RTU Standard User Interfaces; Off state tolerance 1
Requirement deleted. Req. expected to be covered by ECSS-E-ST-50-14C.

Requirement Number : SAVOIR.RTU.UIF.50
RTU Standard User Interfaces; Off state tolerance 2
Requirement deleted. Req. expected to be covered by ECSS-E-ST-50-14C.

Requirement Number : SAVOIR.RTU.UIF.60
RTU Standard User Interfaces; Inductive loads
All interfaces to inductive loads shall provide protective measures (e.g. freewheeling diodes, transzorb diodes) to avoid any damage to the driver interface or connected units due to (self-)induced voltages

- during nominal operation
- due to coupling between operational and inactive components
- due to unexpected trigger of a switch-off protection function
- during inadvertent switch-off of the primary power supply

Requirement Number : SAVOIR.RTU.UIF.70
RTU Standard User Interfaces; Signal acquisitions
For each signal acquisition chain/failure group, complementary telemetry shall be acquired to allow confirming the validity of the acquired data.
Requirement Rationale :
It is common practice that several discrete interface channels are acquired sequentially through a common set of multiplexers. In order to verify that this process is operating without failures, additional reference channels are acquired in addition to the acquisition of the data from the external *User Interfaces*. By careful selection of the signal levels and acquisition time for the reference channels compared to the *User Interface* signals, a comprehensive fault coverage can normally be achieved.

7.6.4.1 Analogue signal interfaces

Requirement Number : SAVOIR.RTU.UIF.80
Analogue Interfaces; A/D Conversion Path Error Detection

Each signal chain used for analogue signal interface acquisitions shall provide mechanisms that allow detecting errors in A/D Converter path functionality (including A/D conversion curve changes or drift).
Requirement Rationale :
It is important to be able to verify the correct operation of the signal acquisition path. It is recommended that independent reference voltages, e.g. at 25% and 75% of the full range, are implemented and sampled by the signal acquisition path. These reference voltages should be independently generated from the reference voltage used by the A/D Converter and independently generated for each A/D converter.

Requirement Number : SAVOIR.RTU.UIF.90
Analogue Interfaces; Reference voltages
Requirement deleted

7.6.4.1.1 Analogue signal monitor (ASM) interface

Requirement Number : SAVOIR.RTU.UIF.100
Analogue Interfaces; Interface type
The analogue signal monitor (ASM) interfaces shall be as per [DISC], paragraph 5.2.

7.6.4.1.2 Temperature sensors monitor (TSM) interface

Requirement Number : SAVOIR.RTU.UIF.110
Temperature monitoring; Interface type
The temperature sensors monitor (TSM) interfaces shall be as per [DISC], paragraph 5.3.

Requirement Number : SAVOIR.RTU.UIF.120
Temperature monitoring; Failure groups
It shall be possible to group a number <TSMAcqChanNumber> of temperature sensors monitor (TSM) acquisition channels in a single failure group f*Requirement Rationale* :
Thermistor acquisitions for thermal control purposes is nominally done using three independent thermistors. The actual temperature is then derived from the three measurements by means of majority voting, median value or averaging. This implies that each of the three thermistors needs to be connected to an independent failure group, ensuring that for a single failure, only one thermistor channel is lost.
<TSMAcqChanNumber> values could vary from 32 to 128.

Requirement Number : SAVOIR.RTU.UIF. 121
Temperature monitoring; Number of Failure groups
The RTU shall implement a number of <TSMFailGroupNumber> of temperature sensors monitor (TSM) acquisition channels failure groups.
Requirement Rationale :

The minimum value of <TSMFailGroupNumber> is three (3)
OptionInfo : Option FailureGroup= YES

7.6.4.2 Bi-Level Discrete Input Interfaces

Requirement Number : SAVOIR.RTU.UIF.130
Bi-Level Discrete Interfaces; Interface type
The Bi-Level Discrete (BDM/BSM) interfaces shall be as per [DISC], paragraph 6.

7.6.4.3 Pulse Command Interfaces

Requirement Number : SAVOIR.RTU.UIF.140
Pulse command; Interface function
The Pulse Command Interface Function shall provide one or several of the following standard pulse command interfaces:

- High Voltage High Power (HV-HPC) Command interface
- Low Voltage High Power (LV-HPC) Command interface
- High Current High Power (HC-HPC) Command interface
- Low Power (LPC) Command interface

Note: The number of channels for each interface type is mission dependant. HV-HPC, LV-HPC, HC-HPC and LPC are interface type defined in [DISC].

Requirement Number : SAVOIR.RTU.UIF.150
Pulse command; Interface logic
A single command to the RTU shall initiate only one Pulse Command

---

Requirement Number : SAVOIR.RTU.UIF.160
Pulse command; Spurious Pulse
For (pulse) command interfaces, no failure within a failure group shall cause more than <SpurPulse> additional (pulse) command to be generated in addition to the one commanded.
Requirement Rationale :
Typical value of <SpurPulse> is 1 (one) , it is 0 (zero) in critical applications

7.6.4.4 Serial Digital Interfaces

Requirement Number : SAVOIR.RTU.UIF.170
Serial Digital Interfaces; Interface type
The Serial Digital interfaces (ISD/OSD/BSD) shall be as per [DISC], paragraph 8.

7.6.5 Secondary Communication Interfaces

The RTU may include one or several Secondary Communication Interfaces such as UART, CAN, SpW or the support for a SpW router. The specification of these is however out of scope for the current version of this RTU specification

---

Requirement Number : SAVOIR.RTU.SCI.10
Secondary Communication Interfaces; Interface type
The RTU shall optionally include one or several UART interfaces as per [RS422].
OptionInfo : Option UART=Yes

7.6.6 Specific User Interfaces

The RTU may include one or several Specific User Interface modules. The specification of these is however out of scope for the current version of this RTU specification.