SUBJECT: SPIRE Autonomy Requirements

PREPARED BY: K.J. King

DOCUMENT No: SPIRE-RAL-PRJ-001855

ISSUE: Issue 1.0 **Date:** 17th May 2006

APPROVED BY:

Instrument Engineer (B.Swinyard)

AGREED BY:

OBS Engineer (S.Molinari)

IFSI Project Manager (R. Orfei)

Distribution

K. J. King	RAL
B.Swinyard	RAL
R. Orfei	IFSI
S. Molinari	IFSI
S. Liu	IFSI
J. Long	RAL

Change Record

ISSUE	DATE	Changes
1.0 Draft 1	31st October 2003	First draft for internal discussion
1.0 Draft 2	7th January 2004	Second draft for external comment
1.0	17th May 2006	First Issue

TABLE OF CONTENTS

FIGURES

Glossary

CDMS	Command Distribution and Management System
CIS	Command Inhibition System
DRCU	Detector Readout and Control Unit
MSS	Mode Switching System
OBS	On-Board Software
PMS	Parameter Monitoring System
S/C	Spacecraft
SPIRE	Spectral and Photometric Imaging Receiver
TM	Telemetry

1. INTRODUCTION

1.1 Scope

This document defines the requirements on the SPIRE On-Board Software (OBS) resulting from an analysis of the potential failure modes of the instrument, its warm electronics and its interfaces to the spacecraft. This analysis has been documented in several places: AD01 defines the instrument internal failure modes that have to be handled by the OBS and the required actions, while AD02 defines the Spacecraft FDIR to which the OBS autonomy functions should be compatible. AD03 defines high-level requirements on the autonomy functionality of all Herschel instruments.

The autonomy of the SPIRE instrument is handled, following AD04, by three mechanisms within the OBS: A parameter monitoring system that allows the OBS to track the status of housekeeping parameters generated by the instrument and to take appropriate action if their value falls outside the expected range; checks carried out when particular hardware (e.g. interfaces) are used; and the identification of run-time errors within the OBS. These three mechanisms are addressed in the following sections:

1.2 Structure of Document

Section 2 defines the set of requirements for an OBS function that is suitable for monitoring the SPIRE instrument health and safety status and taking appropriate actions based on the information available in the housekeeping parameters.

Sections 3-5 analyse the different operational areas of the instrument and define how the facilities described in section 2 are used.

1.3 Documents

1.3.1 Applicable Documents

AD01 Hardware Software Interaction Analysis for SPIRE in-flight Autonomy Functions Specification (SPIRE-RAL-NOT-001719), Issue 1.1, 3rd December 2003

AD02 System Operation and FDIR Requirements (H-P-1-ASP-SP-0209), October 2003 Appendix 1: 1553 Bus FDIR

AD03 Herschel/Planck Operations Interface Requirements Document (SCI-PT-RS-07360), Issue 2.1, 31st December 2001

AD04 Failure Detection Isolation and Recovery Policy in the SPIRE Instrument (SPIRE-RAL-PRJ-001128)

AD05 SPIRE Data ICD (SPIRE-RAL-PRJ-001078)

1.3.2 Reference Documents

RD01 The Subsystem Control Unit (SCU) Design Description (SEDI-SCU-MM-2002-1), Issue 0.7, 17th February 2003

2. AUTONOMY SYSTEMS REQUIREMENTS

This section defines the requirements covering the functionality necessary to implement the autonomy actions that are defined in later sections. This functionality is encompassed in the following three autonomy systems.

2.1 Modes Switching System (MSS)

The instrument will be placed into different configurations (modes) depending on the operations to be performed. It is necessary that the instrument is aware of the current mode as:

- At any time it may be necessary to switch to another mode as a result of an anomaly.
- When monitoring housekeeping parameters the limits applied may be dependant on the current instrument mode.

AUT-MSS-010: The Mode Switching System, its default tables and actions, shall be available on board at start-up of the OBS.

AUT-MSS-020: The current configuration (mode) shall be maintained by the OBS during all operations of the instrument.
An explicit instrument command is provided to identify the current mode. This command will be sent at the end of any sequence of commands designed to put the instrument into one of its operating modes. The mode value provided by the command shalll be stored by the OBS for use by autonomy functions and also placed in the housekeeping telemetry as parameter MODE.

AUT-MSS-030: The OBS shall be capable of switching autonomously between the current mode and other modes in a controlled manner.
It is TBD if it is only necessary to be able to switch into SAFE and STANDBY modes.
This function could be hard coded in the OBS or, preferably, be provided as a Command List. However, the function would need to be able to identify the current mode, which is not currently possible from a Command List.

AUT-MSS-040: It shall be possible to put the instrument directly into a given mode using a single instrument command.
It is TBD if it is only necessary to be able to switch into SAFE and STANDBY modes.

2.2 Parameter Monitoring System (PMS)

The OBS is required to monitor the instrument in order to detect failures that may affect the health and safety of the instrument or one of it subsystems. It does this by comparing the actual values of a set of Monitoring Parameters against the expected/allowed range of values appropriate to the current mode (applying a Monitoring Test) and assigning a Monitored Status. In the event that the Monitoring Parameter has changed its Monitoring Status on more than the maximum allowed number of consecutive occasions, a Monitoring Action may be executed. In addition it must be possible to limit the execution of a Monitoring Test to times when a Monitoring Condition applies.

Figure 2-1 gives a outline flowchart of the Parameter Monitoring System

AUT-PMS-010: The Parameter Monitoring System, its default tables and actions, shall be available on board at start-up of the OBS.

AUT-PMS-020: It shall be possible to report the contents of the PMS tables and actions in telemetry.
A dump of the relevant tables is acceptable

This system can also be used to handle interface errors provided the relevant interface checking results can be made available as Monitoring Parameters.

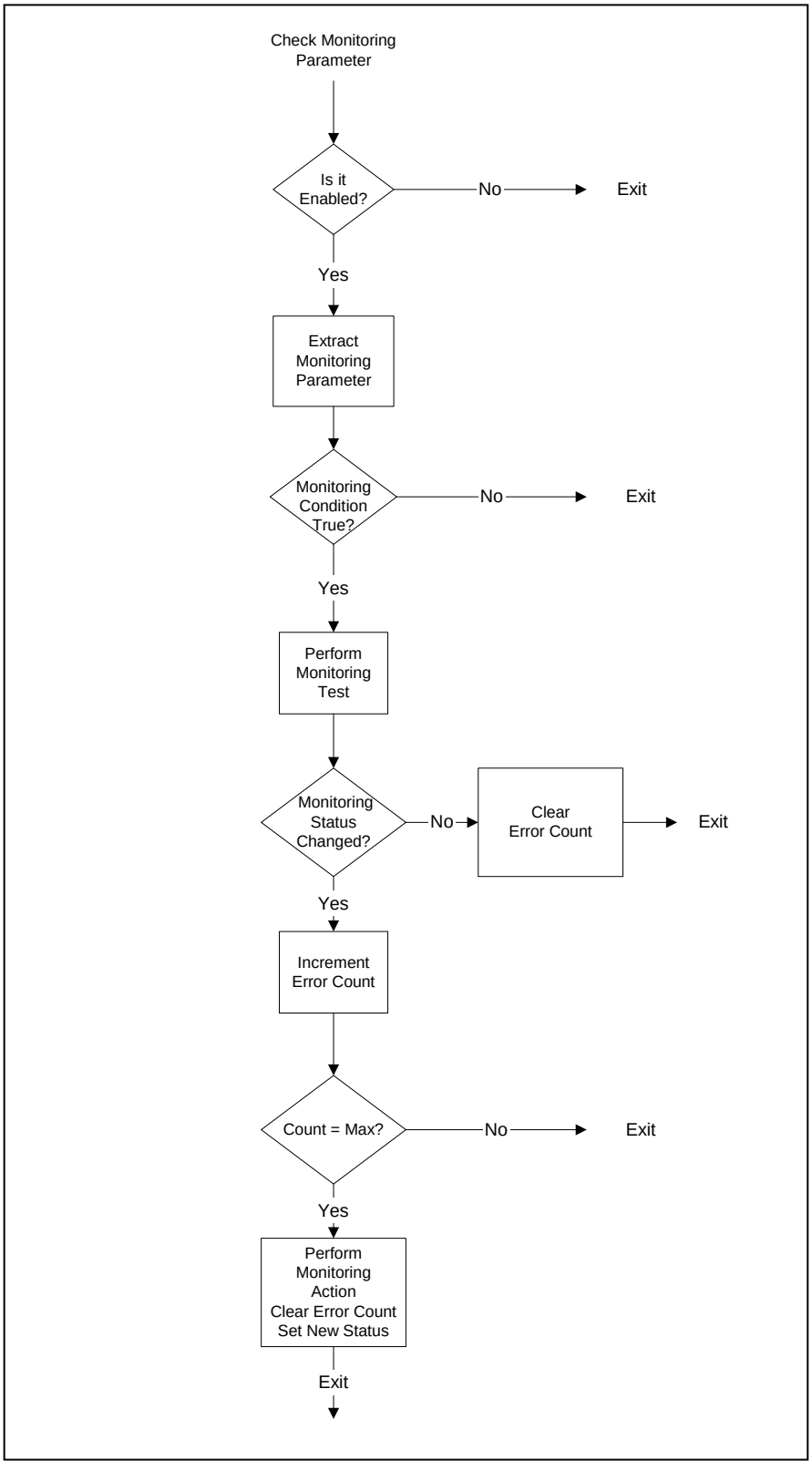

Figure 2-1 Parameter Monitoring System flowchart

2.2.1 Monitoring Parameters

Monitoring Parameters are derived from the housekeeping parameters measured by the instrument subsystems, or from OBS generated variables, contained in the Nominal Housekeeping. In most cases they are identical to the parameter/variable. However, some subsystem parameters measured on board contain more than one **Monitored Parameter** (for example a data word may contain a set of bits each giving the status of a different part of the subsystem). For this reason it should be possible to extract **the Monitoring Parameter** from the measured value.

AUT-PMS-110: The OBS shall maintain a set of Monitoring Parameters that shall be checked regularly (at least as often as the Nominal Housekeeping packet generation).
Monitoring should continue even in the event that the housekeeping telemetry generation is disabled.

AUT-PMS-120: It shall be possible to add or remove entries from the set of Monitoring Parameters, by instrument command.

AUT-PMS-130: It shall be possible to disable checking of a particular Monitoring Parameter, by command, without removing it from the monitored parameters set.
It shall also be possible to re-enable monitoring of this parameter by command at a later stage.

AUT-PMS-140: The OBS shall provide for two types of Monitoring Parameter, Analogue and Digital
Analogue parameters are used to provide the value of a continuously changing signal (e.g. voltage, current). Digital parameters define the current state of a subsystem that may reside in two or more states (e.g. ON/OFF).

AUT-PMS-150: It shall be possible to extract a Monitoring Parameter from a housekeeping parameter
The OBS should allow, as a minimum, selection of a part of a housekeeping parameter by ANDing its value with a mask (bit pattern) and shifting the result by n bits (left or right).

2.2.2 Monitoring Test

AUT-PMS-210: Monitoring of a parameter shall be performed by checking whether the current value of a Monitoring Parameter successfully passes a Monitoring Test.
A Monitoring Test will take one of two forms:
For Analogue parameters

Two ranges specifying the minimum and maximum values (in measured units) that the parameter may take. The two ranges are designated the WARNING and FAILURE ranges. When the measured value is inside both ranges the parameter's status is designated as NOMINAL. When the measured value falls outside the WARNING range but inside the FAILURE range the parameter's status is designated as CRITICAL. When the measured value falls outside both ranges the parameter's status is designated as FAILED.

For Digital parameters:

A set of possible states (maximum 16, TBC) for the parameter and a test (INCLUDED or EXCLUDED). If the parameter value is equal to one of those in the INCLUDED list, or is not equal to one of those in the EXCLUDED list, then its status is designated as NOMINAL, otherwise it is designated as FAILED.

AUT-PMS-220: The OBS shall maintain a Monitoring Test for each Monitoring Parameter.

AUT-PMS-230: It shall be possible to change the Monitoring Test limits or states associated with a Monitoring Parameter, by instrument command.

2.2.3 Monitoring Conditions

AUT-PMS-310: The OBS shall maintain a Monitoring Condition for each Monitoring Parameter
These will determine whether a particular Monitoring Test is applied to a given Monitoring Parameter. A Monitoring Condition shall be a list of Monitoring Parameters (maximum 16, TBC), whose status must all be NOMINAL for the Condition to be true (An empty list shall be deemed to represent no Condition and is always true).
In this way we can define a Monitoring Parameter such as 'MODEIS3', which always checks that the value of MODE is 3. This Monitoring Parameter can then be used as (part of) a Monitoring Condition.

Note: it is important that the order of testing of Monitoring Parameters ensures that parameters used in Conditions are evaluated before the Condition is used.

AUT-PMS-320: Before applying a Monitoring Test for a Monitoring Parameter, the OBS shall check that a Monitoring Condition is passed.
If the Monitoring Condition is not passed the Test will not take place and the Monitoring Status will not be updated. The initial status of a Monitoring Parameter shall be UKNOWN

AUT-PMS-330: It shall be possible to change the Monitoring Condition(s) associated with a Monitoring Parameter, by instrument command.

2.2.4 Monitoring Actions

Monitoring Actions are executed whenever a Monitoring Parameter's status has changed from its previous state. Normally these actions will result in an event packet being generated on any change of status, however if the status changes to FAILED then additional recovery steps may take place (e.g. to place the subsystem or instrument into a safe state.

These actions may be defined in the OBS code or could be provided as a Command List.

AUT-PMS-410: The OBS shall maintain a set of Monitoring Actions

AUT-PMS-420: It shall be possible to add or remove Monitoring Actions from the set already available in the OBS, by instrument command.

AUT-PMS-430: It shall be possible to modify the actions that may be taken when any Monitoring Parameter changes its status
Note: six different actions can be specified for a given Monitoring Parameter, corresponding to the transitions NOMINAL=>CRITICAL, NOMINAL=>FAILED, CRITICAL=>FAILED, CRITICAL=>NOMINAL, FAILED=>CRITICAL, FAILED=>NOMINAL, although the latter two are unlikely to be used.

AUT-PMS-440: It shall be possible to define a Monitoring Action as a Command List to be executed.

AUT-PMS-450: It shall be possible for a Monitoring Action to stop the execution of a VM

AUT-PMS-460: It shall be possible for a Monitoring Action to modify a PMS table. *This may be limited to changing the limits of a Monitoring Test or changing a Monitoring Action.*

2.3 Command Inhibition System (CIS)

Subsystem failures are identified by the Parameter Monitoring System. In the event of failures being found it may be necessary to inhibit commands to a subsystem, or a DRCU unit, or to inhibit commands to other subsystems that affect the failed subsystem. A Command Inhibition System has to be implemented.

AUT-CIS-010: The OBS shall maintain a set of inhibited DRCU commands that shall be checked **every time** a command is sent to the DRCU.
This includes commands sent as housekeeping requests, by OBS code, and by VMs.

AUT-CIS-020: It shall be possible to inhibit commands on an individual basis, by subsystem, or by DRCU unit.
In the case of single DRCU commands that perform several functions (e.g. one command that switches on several power supplies) it shall be possible to inhibit each individual function. I.e. it shall be possible to inhibit the switch on of a single power supply.

AUT-CIS-030: It shall be possible to modify the set of inhibited commands, by instrument command
New commands should be able to be added and commands removed from the set.

AUT-CIS-040: It shall be possible, in a controlled way, to send an inhibited command to the DRCU from a Command List
This is required for diagnostic or autonomy reasons. A mechanism should be used which ensures that it is not possible to issue an inhibited command by mistake. E.g. something like the Arm and Go sequence used to protect the launch latches.

3. DRCU INTERFACES

Subsystems are commanded and monitored through data passing through the interfaces with the DRCU. These interfaces have to be checked to be operating correctly before correct operation of the instrument can be expected. This cannot be done using the PMS as the OBS has to react at a higher speed than can be achieved using the PMS

The DRCU contains three units; the DCU, MCU and SCU, and each one has one slow speed interface (used to send commands to the unit and to collect housekeeping parameters) and one high speed interface (used to transfer science data from the unit to the DPU).

3.1 Low Speed Interface Failures

Commands to a unit of the DRCU are sent on the Low Speed Interface. Two types of command are used: SET commands pass a parameter value to the unit; GET commands request the value of a subsystem parameter from the unit. When operating correctly the unit will return a response word to the DPU containing a copy of the command identifier, an acknowledge field containing a status value and, if the command is a GET command, a parameter value. This response word should be generated within a fixed time of the command being sent to the unit.

AUT-LSI-010: The OBS shall maintain the status of each LS interface with each DRCU unit
The status may be ALIVE, SICK or DEAD

- *ALIVE indicates that the interface is operating nominally.*
- *DEAD indicates that the interface is unusable and the unit should not normally be powered on and all commands that affect the unit should be inhibited*
- *SICK indicates that a problem has occurred, and commands may not execute correctly*

AUT-LSI-020: The OBS shall set the LS Interface status for a unit to ALIVE whenever a command is successfully sent to the unit

AUT-LSI-030: The OBS shall check for a timeout waiting for a response to a command sent to a DRCU unit:
In the event of a timeout the following actions will take place:

- *If the LS Interface status for the unit is ALIVE*
  - *The LS interface status for the unit shall be set to SICK*
  - *An Event Packet TM(5,1) , unique to the unit, shall be generated indicating a Response Error*
- *Else, if the LS Interface status for the unit is SICK*
  - *If the current Mode **is not** DPUON (0)*
    - *The LS interface status for the unit shall be set to DEAD*
    - *An Event Packet TM(5,4), unique to the unit, shall be generated*
    - *A Command List, unique to the unit, shall be executed*

Otherwise the LS Interface status shall be set to ALIVE

AUT-LSI-040: It shall be possible to disable (and re-enable) the check for a timeout waiting for a response to a command sent to a DRCU unit.

AUT-LSI-050: The OBS shall check that the response word to a command to a DRCU unit always contains the correct Command Identifier, and that it also contains an echo of the command parameter field for a SET command.
In the event of the response not being correct the following actions will take place:

- *If the LS Interface status for the unit is ALIVE*
  - *The LS interface status for the unit shall be set to SICK*
  - *An Event Packet TM(5,1) , unique to the unit, shall be generated indicating an CMD Echo Error*
  - *Set the number of attempts to 1*
- *Else, if the LS Interface status for the unit is SICK*
  - *If the number of attempts **equals** the maximum allowed*
    - *The LS interface status for the unit shall be set to DEAD*
    - *An Event Packet TM(5,4), unique to the unit, shall be generated*
    - *A Command List, unique to the unit, shall be executed*

- *Else*
  - *Increment the number of attempts*
  - *Send the command again*

Otherwise the LS Interface status shall be set to ALIVE

AUT-LSI-060: It shall be possible to disable (and re-enable) the check that the response word to a command to a DRCU unit contains the expected contents

AUT-LSI-070: It shall be possible to modify the maximum allowed number of attempts to send a command to a unit. *The value shall be set to 2 for each unit by default, TBC. A different value shall be held for each unit.*

AUT-LSI-080: The OBS shall check the acknowledge field in the response word to a command to the DRCU is zero
In the event of the acknowledge field being non-zero:

- *If the LS Interface status for the unit is ALIVE*
  - *The LS interface status for the unit shall be set to SICK*
  - *A n Event Packet, TM(5.,1), unique to each error type (see AD05), shall be generated*
  - *Set the number of attempts to 1*
- *Else, if the LS Interface status for the unit is SICK*
  - *If the number of attempts **equals** the maximum allowed*
    - *The LS interface status for the unit shall be set to DEAD*
    - *An Event Packet TM(5,4), unique for each unit, shall be generated*
    - *A Command List, unique to the unit, shall be executed*
  - *Else*
    - *Increment the number of attempts*
    - *Send the command again*

Otherwise the LS Interface status shall be set to ALIVE

SPIRE-LSI-090: It shall be possible to disable (and re-enable) the check that the acknowledge field in the response word to a command to a DRCU unit is zero

3.2 High Speed Interface Failures

The high speed interfaces transfer science data from the DRCU units to the DPU. Data is collected by the DPU into FIFO buffers which trigger the DPU to empty them when they are half full.

AUT-HSI-010: The OBS shall maintain the status of each HS interface with each DRCU unit
The status may be ALIVE, SICK or TRANSPARENT

- *ALIVE indicates that the interface is operating nominally.*
- *SICK indicates that a problem has occurred, and data may not be generated correctly*
- *TRANSPARENT indicates that no checking of received data is taking place – the contents of the FIFO are copied directly into Transparent Science packets*

AUT-HSI-020: It shall be possible to command the HS interfaces from TRANSPARENT status into ALIVE status.

AUT-HSI-030: The OBS shall check that science data is generated, within a reasonable time (2 sec, TBC), after sending a data request to the DRCU an report the status of the check in the Nominal Housekeeping
The idea here is to set a flag in the OBS when a science data request is sent to a DRCU unit and to check at some later time that science data has been received.

The Housekeeping status may be used to trigger an event or recovery action

AUT-HSI-040: The OBS shall check that science data frames are correctly received from each DRCU unit (as long as the interface status is not TRANSPARENT).
The OBS shall check that the received Frame_ID, Frame Length and Frame Checksum are correct whenever it collects data from the FIFO.
In the event that one or more of these errors occurs:

- *If the HS Interface status for the unit is ALIVE*
  - *The HS interface status for the unit shall be set to SICK*
  - *If the error is in the Frame ID*
    - *An Event Packet TM(5,1), identifying the Unit, shall be generated*
  - *If the error is in the Frame Length*
    - *An Event Packet TM(5,1), identifying the Unit and FrameId, shall be generated*
  - *If the error is in the Frame Checksum*
    - *An Event Packet TM(5,1), identifying the Unit and FrameId, shall be generated*
  - *Transparent Science Data packets are generated from the FIFO contents – one half of fifo is dumped*
  - *Reset the unit's FIFO – this may lose some data*
  - *Set number of retries to 0*
- *Else*
  - *If retry number = maximum allowed*
    - *The HS Interface status for this unit us set to TRANSPARENT*
    - *Transparent Science Data packets are generated from the FIFO contents*
  - *else*
    - *Increment the retry counter*
  - *exit*

If none of these errors occurs:

- *If the HS Interface status for the unit is SICK*
  - *the HS Interface status shall be set to ALIVE*
  - *An Event Packet TM(5,1) shall be generated*

4. S/C INTERFACE

4.1 Bus Failures

In order to detect 1553 bus failures the OBS needs to identify that information from the CDMS has been lost. The only regular information coming from the CDMS is the Time information delivered by the CDMS in subaddress 8 in subframe 33

AUT-SC-100: The OBS shall check the Time information received is consistent with being updated every second and if not, generate an Event Packet TM(5,1) indicating this anomaly

5. OBS RUNTIME ERRORS

5.1 Memory Errors

These errors arise due to the lack of available memory for the allocation of bocks in the memory pools and Virtuoso fifos for use by the OBS.

AUT-OBS-100: The OBS shall check the occupation level for memory pools and Virtuoso fifos at each request.

AUT-OBS-110: If the occupancy level increases beyond 80% an Event packet, TM(5,1), identifying the memory pool or Virtuoso fifo shall be issued

AUT-OBS-120: If the occupancy level decreases below 70% an Event packet, TM(5,1), identifying the memory pool or Virtuoso fifo shall be issued

AUT-OBS-130: The OBS shall maintain a counter of failed block allocations and Virtuouso fifo allocation for each memory pool and report these in the Normal Housekeeping

5.2 LS Interface Overflow

AUT-OBS-200: The OBS shall check the number of commands to the DRCU, through the LS interface does not exceed the capacity of the interface queue.
In the event of a error the OBS shall

- *issue an Event Packet TM(5,1)*

AUT-OBS-210: The OBS shall indicate the loading of the LS interface in a housekeeping parameter
For example, this may be the number of microseconds that the i/f is busy over the previous second.