Springer Aerospace Technology

Jens Eickhoff *Editor*

The FLP Microsatellite Platform

Flight Operations Manual

Springer

Springer Aerospace Technology

More information about this series at http://www.springer.com/series/8613

Jens Eickhoff
Editor

The FLP Microsatellite Platform

Flight Operations Manual

Springer

Editor
Jens Eickhoff
Airbus DS GmbH
Friedrichshafen
Germany

ISSN 1869-1730 ISSN 1869-1749 (electronic)
Springer Aerospace Technology
ISBN 978-3-319-23502-8 ISBN 978-3-319-23503-5 (eBook)
DOI 10.1007/978-3-319-23503-5

Library of Congress Control Number: 2015950460

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Foreword

Dear Readers,

One of the most common trends in the space sector is the evolution of small satellites. This development will be intensified in the next years. Small satellites of any kind offer a lot of opportunities; no matter if it is small satellite of micro-, nano- or pico-class.

Not least, small satellites enable a lot of different entities with new chances to be space actors. Entities like universities or developing countries bring a new look on space related questions. That is a reason to welcome any new technology helping those to participate. At the end we find in some respects a new space "democracy."

This generation experiences more and faster technological new developments. Miniaturization implies the core of it. Therefore it is a forward-looking idea to develop not only a new class of satellites but a new platform. A platform with its subsystems provides adaptations and can be used for a variety of different payloads. That is the basic idea of the *Future Low-cost Platform* (FLP). The German Aerospace Center DLR supported that idea from its early beginnings up to now.

Editor Jens Eickhoff and all the authors submit a formidable compendium—the "Flight Operations Manual."

This book is a useful introduction and a practical help for all who want to base their work on the FLP, even if they want to familiarize or to deepen their knowledge.

I wish to any reader great benefit by studying the manual and great success for this oeuvre.

June 2015

Dr. Gerd Gruppe
Member of the Executive Board of DLR
German Aerospace Center DLR
Cologne, Germany

Preface

This manual represents the flight operations handbook for the "Future Low-Cost Platform" (FLP), a microsatellite platform developed by the Institut für Raumfahrtsysteme, Universität Stuttgart, Germany with the engineering support from Airbus DS GmbH, Friedrichshafen, Germany.

The FLP is a three-axis-stabilized satellite platform for the 100–150 kg class of satellites corresponding to approximately 1 m^3 size. The manual was developed in parallel to final assembly and test phase of the first satellite based on this FLP platform—the "Flying Laptop" from the IRS, Stuttgart.

Since the manual serves as handbook for the flight operators during mission control it provides a thorough insight into the performance and architecture of the platform and the and the operations techniques—information not available for students in classic academic literature. The manual is therefore structured for an efficient training of new spacecraft operators (i.e., students being trained to control the university satellite) and thus it does not follow document content guidelines for Flight Operations Manuals from ESA, NASA, or any other agency standards.

Although the manual focuses on the generic platform to provide a documentation for future FLP-based spacecraft, it quotes for some passages and elements mission specific data, taken from the "Flying Laptop" project for better illustration purposes, such as for topics like

- the interfaces between platform and payloads,
- the thermal control subsystem—which obviously always covers the entire spacecraft,
- the ground infrastructures from Stuttgart and DLR/GSOC being used for this mission and
- a number of limit table values and event tables in the chapters and annexes.

The first FLP-based satellite "Flying Laptop" provides only an attitude control subsystem but does not yet feature propulsion module for orbit control. It also does not yet provide a SpaceWire-based payload control and payload data processing network. Both such elements have been studied by Airbus DS, Friedrichshafen for

- an ESA Phase-A space debris detection mission called “Space-based Space Surveillance” and
- in an R&T study called OBC-SA together with DLR, Fraunhofer FOKUS and other partners.

Some very brief sections on these topics—including the upgraded onboard computer design under test—are included in this manual to demonstrate the potential of the architecture. The corresponding literature and papers are cited in the references. In house Airbus DS this enhanced design with an upgraded onboard computer, SpaceWire payload equipment network and propulsion subsystem is called “FLP Generation 2.”

June 2015 Prof. Dr.-Ing. Jens Eickhoff

Donation for Life

With the royalties of this book the authors sponsor the German and international bone marrow donor's database for the fight against leukemia and other blood cancer variants. Many of these patients can only be saved by a bone marrow transplant and need a compatible donor.

Germany:
DKMS Deutsche
Knochenmarkspenderdatei
gemeinnützige Gesellschaft mbH
Phone: +49-(0) 70 71-9 43-0
https://www.dkms.de/

USA:
www.deletebloodcancer.org/

UK:
www.deletebloodcancer.org.uk

Poland:
www.dkms.pl

Spain:
www.dkms.de/de/fundación-dkms-españa

Contents

Contributors

Bastian Bätz Institute of Space Systems, University of Stuttgart, Stuttgart, Germany

Maximilian Böttcher Institute of Space Systems, University of Stuttgart, Stuttgart, Germany

Nico Bucher Institute of Space Systems, University of Stuttgart, Stuttgart, Germany

Gianluca Cerrone RHEA System S.A., Wavre, Belgium

Jens Eickhoff Airbus DS GmbH, Friedrichshafen, Germany

Ciprian Furtuna Deutsches Raumfahrt Kontrollzentrum (GSOC), Oberpfaffenhofen, Germany

Marcin Gnat Deutsches Raumfahrt Kontrollzentrum (GSOC), Oberpfaffenhofen, Germany

Philipp Hagel Institute of Space Systems, Deutsches Zentrum für Luft- und Raumfahrt, Bremen, Germany

Wolfgang Heinen RHEA System S.A., Wavre, Belgium

Dougie Johnman Satellite Services B.V., Noordwijk, The Netherlands

Sebastian Keil Institute of Space Systems, University of Stuttgart, Stuttgart, Germany

Kai-Sören Klemich Institute of Space Systems, University of Stuttgart, Stuttgart, Germany

Rolf Kozlowski Deutsches Raumfahrt Kontrollzentrum (GSOC), Oberpfaffenhofen, Germany

Michael Lengowski Institute of Space System, University of Stuttgart, Stuttgart, Germany

Ulrich Mohr Institute of Space Systems, University of Stuttgart, Stuttgart, Germany

Fabian Steinmetz Institute of Space Systems, University of Stuttgart, Stuttgart, Germany

Bryan Tatman Satellite Services B.V., Noordwijk, The Netherlands

Charles Thibaut Institute of Space Systems, University of Stuttgart, Stuttgart, Germany

Paul Walker 4Links Ltd, Milton Keynes, UK

Klaus Wiedemann Deutsches Raumfahrt Kontrollzentrum (GSOC), Oberpfaffenhofen, Germany

Peter Willburger Deutsches Raumfahrt Kontrollzentrum (GSOC), Oberpfaffenhofen, Germany

Rouven Witt Institute of Space Systems, University of Stuttgart, Stuttgart, Germany

Oliver Zeile Institute of Space Systems, University of Stuttgart, Stuttgart, Germany

Abbreviations

General Abbreviations

a.m.	Above mentioned
cf.	Confer
e.g.	Example given
i.e.	Id est -> that is
w.r.t.	With respect to

Technical Abbreviations

AD	Acceptance Check (Bypass Flag = 0) and Data (Control Command Flag = 0) Mode or AD-Mode
AFD	Automated File Distribution System
AIS	Automatic Identification System (Ship Tracking)
AIT	Assembly, Integration and Test
AOS	Attitude of Sight
API	Application Programming Interface
APID	Application ID
ASM	Attached Synchronization Marker (of a CCSDS CADU)
BCR	Battery Charge Regulator
BD	Bypass of Acceptance Check (Bypass Flag = 1) and Data (Control Command Flag = 0) Mode or BD-Mode
BER	Bit Error Rate
BoL	Begin of Life
BRDF	Bidirectional Reflectance Distribution Function (of observed ground targets)
CAD	Computer Aided Design
CADU	Channel Access Data Unit
CC	Combined-Controller
CCSDS	Consultative Committee for Space Data Systems

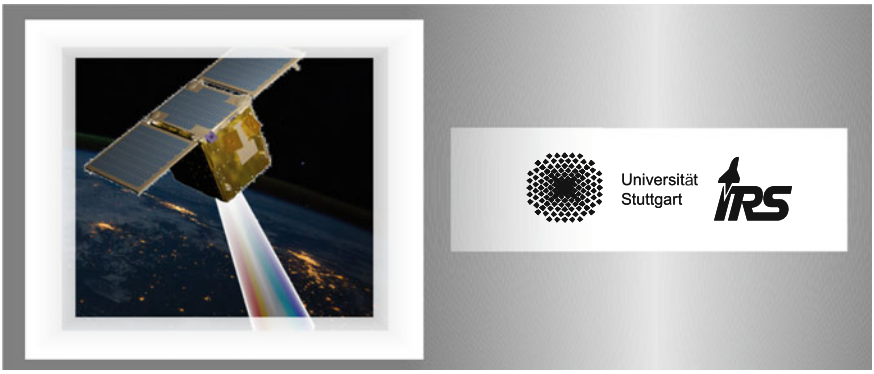
CD	Clock Divider
CDM	Clock Distribution Module
CDPI	Combined Data and Power Management Infrastructure
CLCW	Command Link Control Word
CLTU	Command Link Transmission Unit
CPDU	Command Pulse Decoding Unit
CPU	Central Processing Unit
CRC	Cyclic Redundancy Code
CSP	Command Sequence Parameter File
CTR	Controller
CVP	Command Verification Profiles
CVS	Concurrent Versioning System
CW	Continuous Wave Mode or CW-Mode
DFT	Data Flow Test
DH	Device Handler
DoD	Depth of Discharge
DOM	De-Orbiting-Mechanism
DPU	Data Processing Unit (of the Star Trackers)
DRAM	Dynamic Random Access Memory
ECSS	European Cooperation on Space Standardization
EDAC	Error Detection and Correction
EEPROM	Electrically Erasable PROM
EoL	End of Life
ESA	European Space Agency
ESD	Electrostatic Discharge
FCLTU	Forward Communications Link Transmission Unit
FDE	Flight Dynamics Events
FDIR	Failure Detection, Isolation, and Recovery
FM	Flight Model
FOG	Fiberoptic Gyro
FOM	Flight Operations Manual
FPGA	Field Programmable Gate Array
FRAM	Ferroelectric Random Access Memory
GCC	Ground Control Center
GDS	Ground Data Systems (DLR)
GPS	Global Positioning System
HF	High Frequency
HK	Housekeeping
HPC	High Priority Command
HW	Hardware
I/O	Input/Output
IADC	United Nations Inter-Agency Space Debris Coordination Committee
IF	Interface
IMBU	Integrated Modem and Baseband Unit

IRS	Institut für Raumfahrtsysteme, Institute of Space Systems, University of Stuttgart, Germany
ITAR	International Traffic in Arms Regulations
ITU	International Telecommunication Union
JTAG	Joint Test Actions Group (Test Interface)
LCL	Latching Current Limiter
LEOP	Launch and Early Orbit Phase
LHCP	Left Hand Circular Polarization
LOS	Line of Sight
LTAN	Local Time of Ascending Node
LTDN	Local Time of Descending Node
MAP-ID	Multiplexer Access Point Identifier
MCS	Mission Control System
MGM	Magnetometer
MGT	Magnetotorquer
MIB	Mission Information Base
MOIS	Manufacturing and Operations Information System
MRAM	Magnetoresistive Random Access Memory
NASA	National Aeronautics and Space Administration
NCTRS	Network Control and Telemetry Routing System
NIS	Network Integration System
NORAD	North American Aerospace Defense Command
NRZ-L	Non-Return-to-Zero Level
NRZ-M	Non-Return-to-Zero Mode
NV-RAM	Non-Volatile RAM
OBC	Onboard Computer
OBSW	Onboard Software
OBSW-DP	Onboard Software Data Pool
OS	Operating System
PCB	Printed Cirquit Board
PCDU	Power Control and Distribution Unit
PLOC	Payload Controller—"Payload Onboard Computer"
PPS	Pulse Per Second
PROM	Programmable Read-Only Memory
PST	Polling Sequence Table
PSU	Power Supply Unit
PUS	ESA Packet Utilization Standard
QA	Quality Assurance
RAF	Remote Acquisition Forwarding
RAM	Random Access Memory
RCF	Remote Call Forwarding
RF	Radio Frequency
RF-SCOE	Radio Frequency Special Checkout Equipment
RHCP	Right Hand Circular Polarization
RISC	Reduced Instruction Set Computer

RMS	Rate Monotonic Scheduling
ROM	Read-Only Memory
RTEMS	Real-Time Executive for Multiprocessor Systems
RTOS	Real-Time Operating System
RTS	Real-Time Simulator
RW	Reaction Wheel
S/C	Spacecraft
S2K	SCOS-2000
SA	Solar Array
SBC	Single Board Computer
SCID	Spacecraft Identifier
SCOE	Special Checkout Equipment
SCV	Spacecraft Configuration Vector
SEU	Single Event Upset
SIF	Service Interface
SLE	Space Link Extension
SoC	Battery State of Charge
SPARC	Scalable Processor Architecture
SpW	SpaceWire
SRAM	Static Random Access Memory
SSB	SLE Switchboard
SSO	Sun-Synchronous Orbit
SSRAM	Synchronous Static Random Access Memory
STR	Star Tracker
SUS	Sun Sensor
SV	State Vector
SW	Software
TAA	Technical Assistance Agreement
TC	Telecommand
TCC	Telecommand Channel Layer
TCS	Thermal Control System
TF	Transfer Frame
TID	Total Ionizing Dose
TM	Telemetry
VC	Virtual Channel
VCID	Virtual Channel Identifier
WS	Workstation
WSP-C	Weilheim SLE Provider to Cortex
XNS	Extended Navigation Solution

Chapter 1
Introduction to the Microsatellite Platform

Jens Eickhoff



Abstract The book represents the Flight Operations Manual of the microsatellite platform. Before diving into the operations of the platform and its subsystems, into failure management and ground station functions, this chapter provides a brief overview on the Future Low-cost Platform (FLP). It explains the basic concept of the platform versus a complete satellite, it sketches out the first FLP prototype satellite "Flying Laptop" and explains which parts represent the re-usable platform. These platform elements can be taken over for other missions to significantly reduce the development effort.

Keywords Stuttgart small satellite program · Future low-cost platform (FLP) · Prototype satellite "Flying Laptop" · Orbit and operational modes · Platform re-usability · Subsystems overview · Combined Data and Power Management Infrastructure

J. Eickhoff (✉)
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_1

1.1 The University Small Satellite Program

Over the past 6 years a microsatellite platform has been developed at the "Institut für Raumfahrtsysteme" (IRS) of the University of Stuttgart, Germany with substantial coaching and co-funding from its partner Airbus DS GmbH, Friedrichshafen, Germany. This platform is called the "Future Low-cost Platform" (FLP) and it is targeted for the 50–150 kg class of small satellites, corresponding approximately to the up to 1 m^3 volume class and it is designed for satellites launched as secondary passengers.

The FLP is a fully three-axis stabilized platform. It is suited for a variety of mission scenarios performing either nadir-pointing Earth observation with e.g. scanner instruments, or target pointing Earth observation e.g. with camera instruments or inertial pointing e.g. for astronomical missions and telescope payloads—and obviously for combinations thereof. The total pointing error during one pass is less than 150 arcsec and the pointing knowledge is better than 7 arcsec. To achieve these values, state of the art star trackers, magnetometers and fiberoptic gyros as well as GPS receivers are used as avionics sensors. Reaction wheels and magnetotorquers are used as actuators.

The "Flying Laptop" (see Fig. 1.1) is the first satellite in the SmallSat Program of the IRS and is based on the FLP design. It has been developed and built primarily

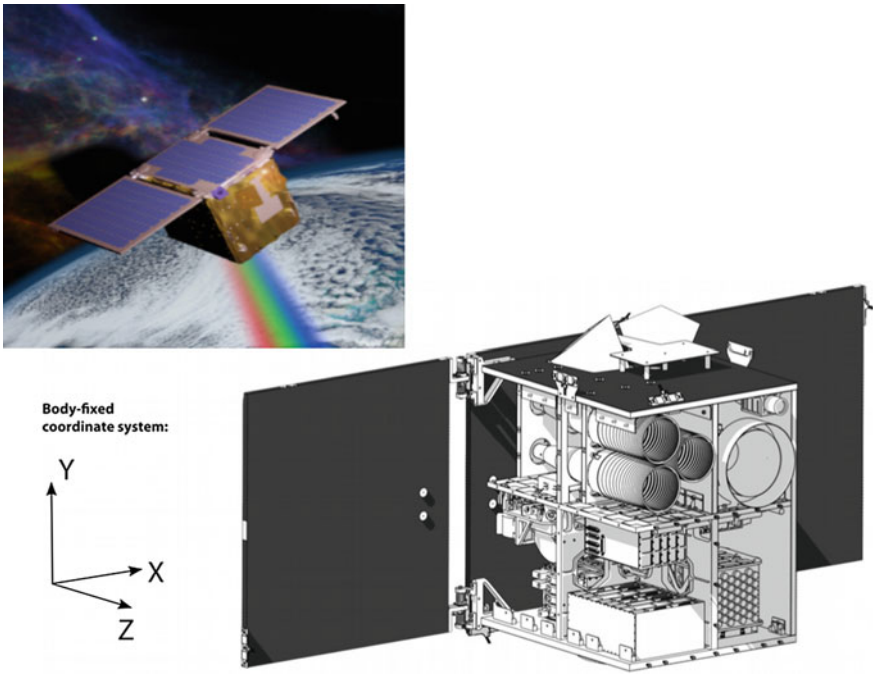


Fig. 1.1 Flying Laptop artist's view and mechanical configuration. © IRS, University of Stuttgart

by Ph.D. and graduate students, supported in manpower, material and facility usage by Airbus DS GmbH, by the Airbus DS daughter TESAT Spacecom, by space agency and research facilities (Deutsches Zentrum für Luft- und Raumfahrt, DLR). The project is financed by the federal state of Baden-Württemberg, the university and the industry partners. At time of manuscript submission the "Flying Laptop" has successfully completed its environmental test campaign, its final Onboard Software (OBSW) tests and is waiting for a launch opportunity. The partners share diverse interests in this platform/satellite project:

- The main project goals for the industry and agency partners is to qualify electronic components for space, in particular the elements of the innovative, integrated OBC/PCDU infrastructure CDPI [47] and to qualify the "Future Low-cost Platform" (FLP)—"future" since at time of manuscript submission—it still lacks in-orbit qualification.
- For the university the mission goals are and have been to establish the expertise and infrastructure for development, integration, test and operations of satellites at the IRS. The program is also used to improve the education of students by providing the possibility for hands on experience within a challenging space project. Once in orbit, the satellite will be used to demonstrate new technologies, to perform Earth observation and to train students as mission operators using a state-of-the-art mission control infrastructure.

The "Flying Laptop"—as FLP Generation 1 spacecraft—is not yet equipped with any propulsion subsystem for orbit control. For more details please refer to Sect. 1.11. Airbus DS GmbH meanwhile has studied an enhanced version of the platform—FLP Generation 2—featuring a hydrazine propulsion subsystem for orbit control and de-orbiting from higher orbit altitudes (Table 1.1).

Table 1.1 "Flying Laptop" satellite characteristics

Dimensions	60 × 70 × 80 cm
Mass	117 kg
Launch Type	Piggy-Back (secondary payload)
Desired Orbit	circular, polar
Orbit Altitude	500–650 km
Attitude Control	3-axis stabilized
Communications	S-band
Solar Panels	3 (2 deployable)

1.2 Satellite Orbit

The satellite has been designed for a circular Sun-Synchronous Orbit (SSO) with a Local Time of Descending Node (LTDN) between 9:30 h and 11:00 h. As the operational lifetime of this satellite is targeted to be two years and the satellite should not stay in orbit for more than 25 years after end of life (considering the European Code of Conduct on Space Debris Mitigation [1]), the desired orbital altitude is between 500 and 650 km. For de-orbiting after operational use the satellite is equipped with an experimental De-Orbiting-Mechanism (DOM) from Tohoku University, Japan.

1.3 Mechanical Design and Launcher Interface

The "Flying Laptop" is a cuboid with two deployable solar panels. It has a total mass of 117 kg. Figure 1.1 shows the satellite configuration with deployed solar panels. Its main dimensions during launch with undeployed solar panels are 600 by 702 by 866 mm^3 as depicted in Fig. 1.2.

A launcher adapter ring is installed on the satellite. The depicted variant is designed to be compliant to the Soyuz piggy-back separation adapter. A deployable De-Orbiting-Mechanism (DOM) is located inside the launch adapter ring.

The structure of the "Flying Laptop" is a hybrid design. The lower part is made of integral aluminum parts and the upper part, where the optical payloads are installed, consists of carbon-fiber reinforced sandwich structures which provide a more stable alignment of the cameras due to their low thermal expansion. The thermal control subsystem (TCS) of the satellite consists of several temperature sensors and heaters inside the satellite as well as Multi-Layer-Insulation and radiators on the outside. No active cooling system is used (Fig. 1.3).

1.4 Technology and Payloads

As mentioned above, the purpose of the "Flying Laptop" mission is to demonstrate new technologies and to space qualify the FLP platform design. The main asset—although it is not a payload—concerns the "Combined Data and Power Management Infrastructure" (CDPI) with the Onboard Computer (OBC) based on a LEON3FT processor, SpaceWire driven I/O-Boards which serve as a communication interface between the Processor-Boards, full CCSDS/PUS standard based Telecommand (TC) and Telemetry (TM) via CCSDS-Decoder/Encoder-Boards and the OBC reconfiguration performed by the CDPI Combined-Controller in the PCDU. The international partner consortium for the CDPI architecture is Airbus DS GmbH (Germany), Aeroflex Colorado Springs Inc. (USA), Aeroflex Gaisler AB

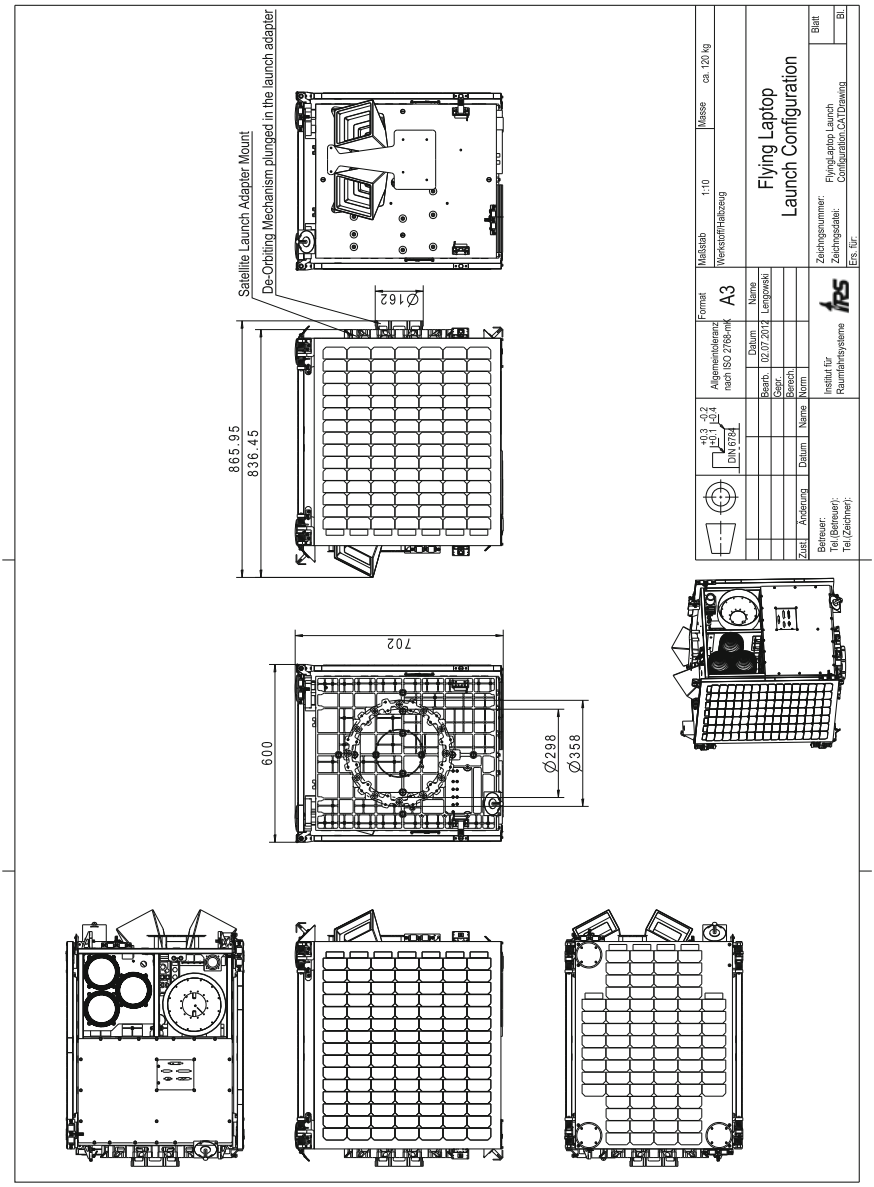


Fig. 1.2 Flying Laptop satellite launch configuration. © IRS, University of Stuttgart

(Sweden), 4Links Ltd. (UK), HEMA Kabeltechnik GmbH & Co. KG (Germany) and Vectronic Aerospace GmbH (Germany). For more details please refer to [47].

The main payload of the FLP mission "Flying Laptop"is the Multi-spectral Imaging Camera System (MICS), which consists of three separate cameras, using

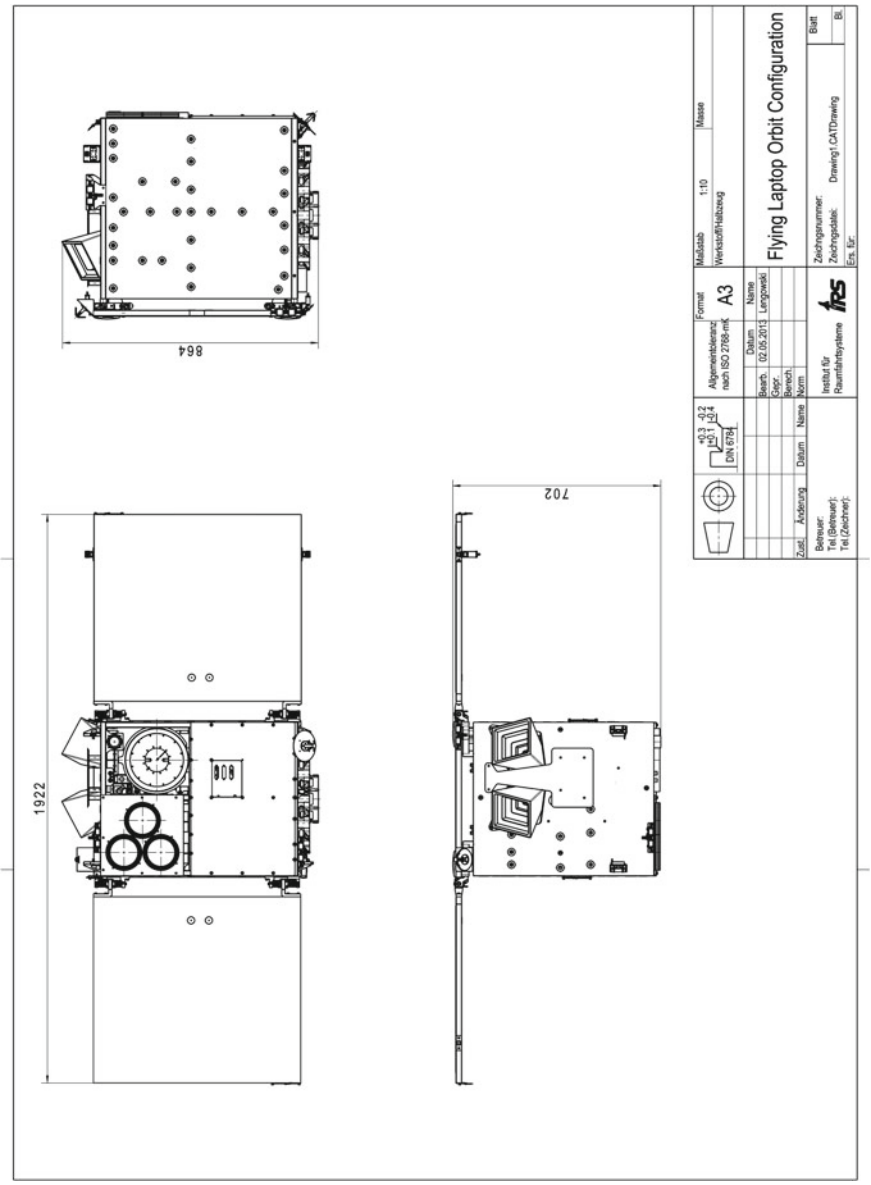


Fig. 1.3 Flying Laptop with solar arrays deployed. © IRS, University of Stuttgart

filters for green, red and near-infrared spectral ranges. It is used for multi-angle and multi-spectral imaging of the Earth with a ground sample distance of approximately 25 m and a swath width of 25 km. The MICS is additionally used in a ship observation experiment. For this experiment the satellite is equipped with an Automatic Identification System (AIS) to receive ship transponder signals for navigation and ship traffic control. The AIS-Receiver is developed and built at the Institute of Space Systems of DLR, Bremen. By mapping the AIS information over the optical data of the MICS the satellite based observation of ship traffic shall be studied and improved. Another camera with a wider field of view is used to get a better overview of the area observed with the MICS. This Panoramic Camera (PAMCAM) is based on a commercial off-the-shelf product and has a swath width of approximately 200 km. Furthermore on instrument side the satellite is equipped with an Optical Infrared Link System (OSIRIS) to demonstrate high speed down-link capabilities using an optical terminal which was developed by DLR, Oberpfaffenhofen, Germany.

All payloads are controlled via a dedicated FPGA-based payload controller—the Payload On-Board Computer (PLOC). Besides that, three GPS sensors shall be used to precisely determine the position of the satellite (and in the GENIUS experiment even the attitude).

1.5 Platform Re-usability

Over the development phase of the spacecraft, the satellite platform design was tailored subsequently to meet international agency and industry standards, such as the CCSDS [15] and ESA Packet Utilization Standard [13] based communication protocols, SpaceWire data handling standard [3] etc.

The platform setup is depicted in Fig. 1.4. It comprises a data handling subsystem, an attitude control subsystem, the power subsystem and the telecommand/telemetry and control subsystem.

Figure 1.5 depicts the platform equipment integration into the first FLP based satellite, the “Flying Laptop”. Color codes are kept compatible with Fig. 1.6. Its payloads are marked in orange while structural elements are presented in grey shades. Although this Flight Operations Manual, focuses on the satellite platform management in orbit, in some sections performance data of the entire satellite configuration are cited, such as solar array total power, mechanical parameters like moments of inertia and others. The usability of the platform elements in other geometric satellite configurations is treated later in Sect. 1.15.

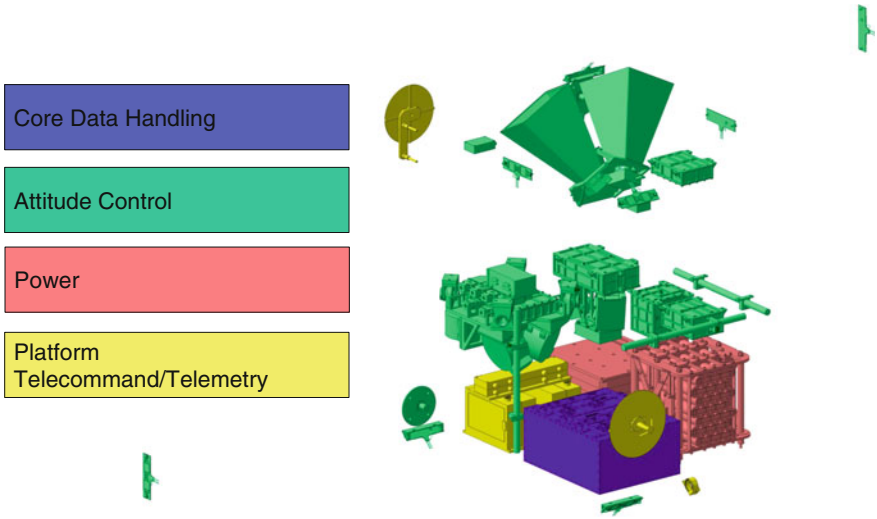


Fig. 1.4 FLP Generation 1 platform infrastructure. © IRS, University of Stuttgart

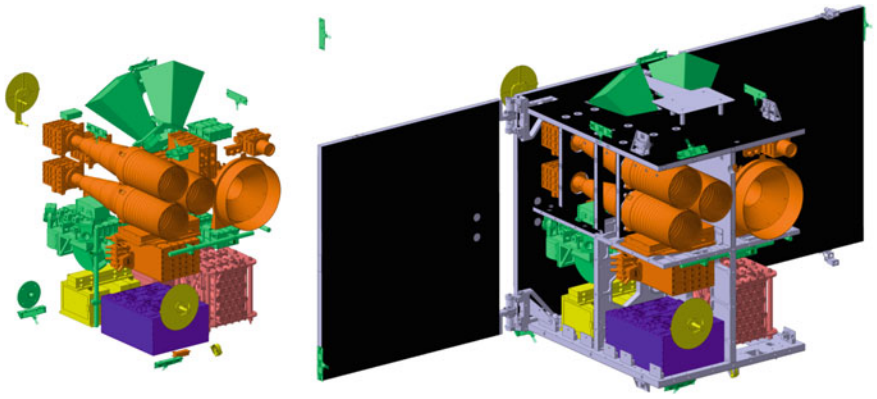


Fig. 1.5 Platform plus payload plus structure of the first FLP based mission, the "Flying Laptop". © IRS, University of Stuttgart

1.6 Platform Redundancy Concept

The platform design is fully redundant with either internally redundant units—like OBC or PCDU, 3-out-of-4 redundancy as for reaction wheels and fiberoptic gyroscopes, or external redundancy as for the receivers and transmitters.

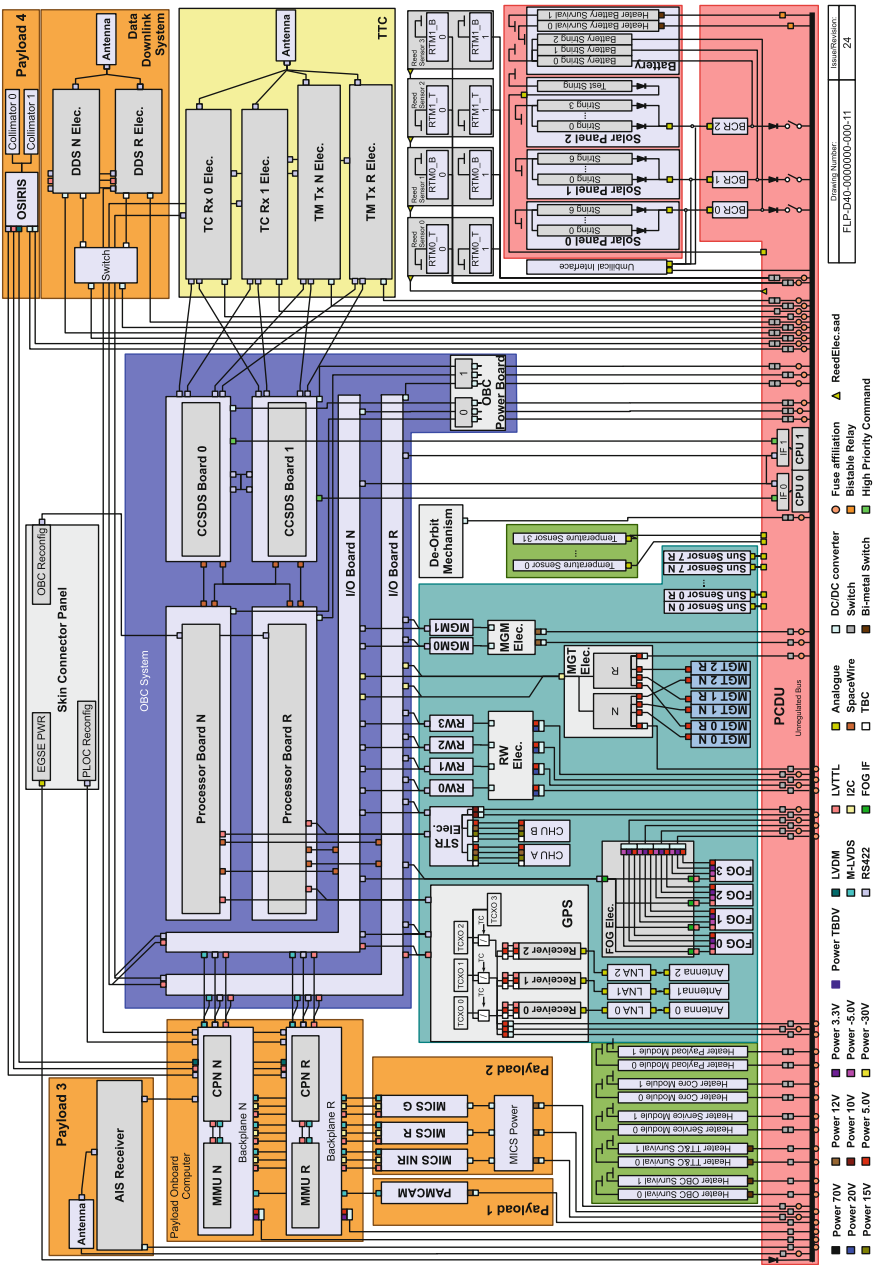


Fig. 1.6 Flying Laptop satellite electrical block diagram. © IRS, University of Stuttgart

Thus every subsystem provides internal redundancy to prevent the mission from being lost due to a single point failure. As cross-coupling is applied for all externally redundant equipment units, the harness is very unlikely to initiate a mission failure. Please also refer to Fig. 1.6.

1.7 Power Subsystem and Electrical Block Diagram

1.7.1 Solar Panels

The solar panels of the "Flying Laptop" consist of GAGET1-ID/160-8040 cells manufactured by AZUR SPACE Solar Power GmbH. These cells are triple junction $GaInP_2$/GaAs/Ge cells with an average efficiency of 25.3 % and area of 30.18 cm^2. The EoL efficiency of the final panel assembly is estimated to be approximately 19 % [59]. There are 105 cells on each of the two wing panels allocated in seven strings of 15 cells each.

On the body mounted panel which will be warmer due to increased heat conduction from the main body there are four strings of 16 cells each in order to achieve a similar operating voltage to the cooler wing panels. The panels feature shunt diodes for every cell so that damaged or shaded cells do not cause an entire string to fail.

The body mounted panel also features an additional test string made of more advanced triple junction cells of type RWE3G-ID2*/150-8040. These cells are are similar to the standard cells described above, but can reach an average efficiency of 27.6 % [60]. The performance of these cells will be thoroughly monitored during the mission.

The maximum achievable power in this overall solar panel configuration sums up to 230 W.

To deploy the solar panels after launch, no pyrotechnics are used. Instead, a melting wire mechanism has been developed by IRS. For this mechanism, the bolts retaining the solar panels during launch are attached to the satellite using a split pod, which is held together by the melting wire. After the separation from the upper stage of the launcher, the wire is melted by heat resistors and the panels are deployed. This process is controlled automatically by the Power Control and Distribution Unit (PCDU) during satellite boot-up in orbit. For more details see Sect. 5.3.

1.7.2 Battery

The battery consists of ANR 26650 M1-B Lithium-Iron-Phosphate ($LiFePO_4$) cells manufactured by A123 Systems. One cell has a nominal capacity of 2.5 Ah at a nominal voltage of 3.3 V. One battery string is directly attached to each of the solar panels. The strings for the side panels consist of 35 cells each, with blocks of five

cells in parallel and seven of these blocks in series, yielding a total capacity of 12.5 Ah at a voltage between 18.5 and 25.2 V. For the center panel string, there are only four cells in parallel, reducing the nominal capacity to 10 Ah. Thus the entire battery has a nominal BoL capacity of 32.5 Ah.

In sunlight, the battery is charged with the maximum possible current from the solar panels until a single cell reaches a voltage of 3.6 V at which point trickle charging is applied until the end of the sunlight period. During nominal operations, the battery is only discharged to approximately 70 % SoC. The Battery Charge Regulators (BCR) for the three strings are part of the PCDU.

The selected cells have a high cycle life and can deliver high currents without being damaged. However, they are also susceptible to overcharging and the voltages of cells in series tend to drift apart. In order to minimize this effect, the cells in a string are selected by comparing their charge/discharge characteristics and bringing them to the same State-of-Charge (SoC) before integrating them into the string. Furthermore a passive equalizing concept using bleed resistors is applied to balance the state of charge of the cells. There is a bleed resistor of 820 Ω for every block of cells in parallel in order to reduce the SoC of cells with higher voltages. A simple comparator circuit is used to control the voltage of the single blocks during charge. If the voltage of one block exceeds the maximum charge voltage of 3.6 V a signal is sent to the PCDU to stop charging of the affected string.

1.7.3 The Power Control and Distribution Unit

The power subsystem is controlled by a Power Control and Distribution Unit (PCDU), which includes the Battery Charge Regulators, provides an unregulated bus voltage to the satellite platform and instrument units. In the configuration of the "Flying Laptop" due to solar array size and resulting number of cells per string the bus voltage ranges between 18.0 and 25.0 V. In a configuration with larger panels the usually applied 28 V bus voltage can be achieved for future FLP spacecraft.

The PCDU is custom designed and is manufactured by Vectronic Aerospace, Berlin, Germany. It is controlled by the OBC and is able to switch the individual platform and payload equipment power lines. For every component the voltage conversion is done by DC/DC-converters on dedicated electronic boards. The PCDU manages the charge control of the cited battery.

Inside the PCDU the electric design is as follows (for details see Fig. 5.9):

- Each solar panel is connected to a separate Battery Charge Regulator (BCR) and to a single battery string.
- Depending on the input from the solar panels and the battery SoC, the PCDU switches between battery charge and discharge.
- It also measures several voltages, currents and temperatures.
- The loads are connected to the PCDU via fuses and switches using an unregulated power-bus of 18–28 V DC.

- Each load has its own DC/DC-converter to achieve the specified supply voltage.
- The control and measurement parameters which are transferred between Onboard Computer and PCDU as well as the PCDU commanding are treated in more detail in Chap. 5.

Besides its main function to control and distribute power to the loads, the PCDU is able to restart the satellite system in case of a complete power outage shutdown. Furthermore, it is used to distribute High Priority Commands (HPC) in case the OBC needs to be power-cycled for hard reset or to be reconfigured. In this case, these commands are routed directly from the CCSDS-Board to the PCDU, which can then switch any unit on or off. Further details are provided in Chap. 10.

1.8 Core Data Handling Subsystem

The core data handling subsystem (Core DHS) of the FLP platform consists of a so-called Combined Data and Power Management Infrastructure (CDPI) as it is described in detail in [47]. In the FLP Generation 1 design this CDPI controls the satellite platform and a dedicated payload control computer is foreseen for payload management as it is described in Sect. 1.9.

In the upgraded FLP Generation 2 design conceptualized by Airbus DS both platform and payload control are managed by a single CDPI, based on a SpaceWire network architecture. For more details see Sect. 9.3.2 and Fig. 9.14.

A CDPI is a logical merge of a classic Onboard Computer (OBC) and a Power Control and Distribution Unit (PCDU). The key features are:

- The OBC only includes the digital I/O interfaces.
- The PCDU covers the analog I/O as it is anyway equipped with D/A and A/D converters.
- The infrastructure distributes I/O functions over OBC and PCDU and it is equipped with one single reconfiguration controller, which is able to handle both power failures as well as PCDU and OBC board reconfiguration cases.
- The reconfiguration and power control chip is based on a real processor plus firmware.
  - This saves qualification and manufacturing of a dedicated OBC reconfiguration control ASIC.
  - Furthermore such a unit is able to directly process emergency commands of HPC type without an intermediate pulse decoding unit.

Details on the CDPI functionality and its embedding in the overall Core DHS can be found in Chap. 4.

For the first FLP based mission, the “Flying Laptop”, the two units OBC and PCDU are kept in two separate housings (see Fig. 1.7) just for reasons of physical mounting in separate compartments of the spacecraft (cf. Figs. 1.4 and 1.5). They

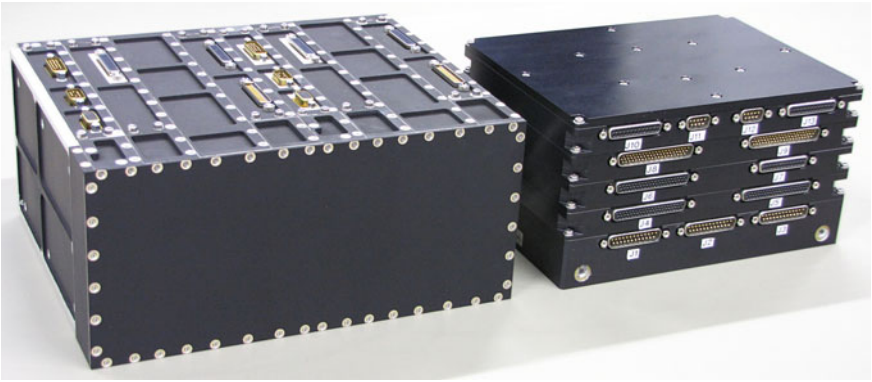


Fig. 1.7 OBC and PCDU forming a Combined Data and Power Management Infrastructure. © IRS, University of Stuttgart—from [47]

can also easily be combined into one single box for other missions with different mechanical design of the spacecraft structure.

In the assembly for the "Flying Laptop" the Reconfiguration Unit is physically included in the PCDU.

1.9 FLP Payload Control Computer

A Payload Module Controller (PMC) and a science data Mass Memory Unit (MMU) are not part of the FLP Generation 1 satellite platform itself, since they are highly dependent on the number of individual payloads and their type—both subject to the individual mission goals. The first FLP platform based satellite, the "Flying Laptop" for example uses a university made, FPGA based, internally redundant PMC with directly coupled MMU Modules. In case of the "Flying Laptop" this payload control computer is called "Payload Onboard Computer" (PLOC). The EM of this assembly is depicted in Fig. 1.8.

This volume of the Flight Operations Manual only covers the general commands to manage the PMC w.r.t. power, mode switching. An IRS internal volume 2 covers the "Flying Laptop" mission specific payloads comprises all payload command and control via the PMC.

Fig. 1.8 Payload module computer: "Flying Laptop" mission example. © IRS, University of Stuttgart

1.10 Attitude Control Subsystem

The attitude control subsystem (ACS) of the FLP satellite and its algorithms were fully defined, developed and tested as described in [41, 42, 124, 125] in a Matlab/Simulink environment. In a second step they were converted into C++ and integrated into the OBSW objects. The ACS key features are:

- Damping of rotational rates after its separation from the launch-vehicle or in case of emergency.
- A safe-mode in which power supply is ensured by utilizing reliable equipment only.
- Coarse attitude determination by using reliable sensors.
- Coarse nadir attitude acquisition without Reaction Wheels (RWL) for RWL servicing in orbit and for taking the RWLs into operation during the satellite Commissioning Phase.
- The capability of pointing the satellite to any given target with an absolute pointing accuracy of 150 arcsec.
- State estimation and rating of sensor data.

1.10.1 Sensors and Actuators

Platform sensors
Table 1.2 shows all sensors of the FLP Generation 1 platform that are used for the attitude control subsystem:

- Two redundant Magnetometers (MGM),
- four Fiberoptic Gyros (FOG),
- eight Sun Sensors (SUS) distributed all over the satellite,
- a GPS system with three receivers and antennas and
- a high precision Star Tracker (STR) system with two camera units.

Magnetometers (MGM) The satellite platform is equipped with two three axis anisotropic-magnetoresistive (AMR) magnetometers manufactured by ZARM-Technik, Bremen, Germany. The measured vector of the magnetic field of the Earth is used as input information for the magnetic torquers, for detumble, safe mode and for the desaturation of the reaction wheels.

Sun Sensor Unit (SSU) The design of the sun sensor unit is based on two GaAs solar cell on each PCB to make it a hot redundant configuration. Eight such units on the FLP body are utilized to construct the overall 4π view.

GPS unit The GPS system consists of three independent GPS receiver boards, each connected with a separate antenna and low noise amplifier. The chosen configuration is a requirement of an experiment which will be conducted in cooperation with the German Space Agency (DLR/GSOC) for GPS based attitude determination (GENIUS).

FOG Unit Four fiber optic gyros are used in a tetrahedron configuration for the measurement of the body angular rates. The tetrahedron configuration offers a

Table 1.2 Sensors overview

	MGM	FOG	SUS	GPS	STR
Output	magnetic field vector (3 × 1)	rotational rate (scalar)	solar cell current (scalar)	position (3 × 1) velocity (3 × 1)	inertial quaternion (4 × 1)
Sensors	2	4	6	3	2
Unit	[T]	[deg/s]	[A]	[m] [m/s]	[–]
Resolution/ Accuracy	5 nT (LSB)	2×10^{-6} deg/s (LSB)	50 mA	10 m 0.1 m/s	5 arcsec
Control Rate	1.5 Hz/3 Hz/6 Hz	10 Hz	10 Hz	1 Hz	5 Hz
Connection Type	RS422	FOG IF	RS422 (PCDU)	RS422	RS422
Manufacturer	ZARM	NG-LITEF	Vectronic Aerospace	DLR	DTU

Table 1.3 Actuators overview

	MGT	RWL
Input	magnetic dipole moment	rotational rate
Quantity	1 unit 3 redundant coils	4 reaction wheels
Unit	[Am^2]	[Nms] resp. [Nm]
Performance	Max. magnetic momentum: 6 [Am^2]	Max. angular momentum: ±0.12 [Nms] Max. torque: ±0.015 [Nm]
Control Rate	10 Hz	10 Hz
Connection Type	IIC	RS422
Manufacturer	ZARM	Rockwell Collins

3-out-of-4 redundancy. The type of the sensor is a C-FORS (Commercial Fiber Optic Rate Sensor) from Northrop-Grumman/Litef, Freiburg, Germany. The C-FORS as specified by the manufacturer has a maximum rate bias of 27°/h over −40° to +77° and a maximum angular random walk of 0.15°/$\sqrt{h}$.

Star Tracker (STR) The star tracker system consists of a micro data processing unit and two camera head units (CHU), each equipped with a baffle system. The selected model is a micro-Advances Stellar Compass (μASC) developed by the Technical University of Denmark (DTU), Kgs. Lyngby, Denmark. Each CHU uses a CCD-chip with a size of 7.95 mm × 6.45 mm with 752 × 558 pixels to take an image of the stars. With the used optics the field of view (FOV) results to 13.4° × 18.4° and its diagonal covers 22.76°.

Platform actuators

Table 1.3 gives the key parameters of the FLP satellite's actuator system which is composed of:

- Three Magnetotorquers (MGT) and
- four Reaction Wheels (RW).

Magnetic Torquers (MGT) Three magnetic torquers with a linear dipole moment of 6 Am^2 are used as an actuator in detumble and safe mode and during normal operation they dump the momentum accumulated by the reaction wheels. The magnetic torquers are also provided by ZARM-Technik, Bremen, Germany. Each torquer consists of two concentric coils wound parallel on top of each other. Both can be operated independently, which is providing redundancy. The mass of one torquer is 0.24 kg.

Reaction Wheels (RW) Four reactions wheels are arranged in a hot redundant tetrahedron like configuration. Each of the wheels has an angular momentum capacity of 0.12 Nms and a reaction torque of 5 mNm over the range of ±3,000 rpm. The selected reaction wheel is manufactured by Rockwell Collins of type RSI 01-5/28 having a mass of 0.7 kg. Each RW is connected to an FPGA-based electronics assembly in a star like configuration via its RS-422 bidirectional interface. This allows the parallel RW commanding via ACS commands.

1.10.2 Satellite System Modes

The FLP satellite control system can be operated in several different modes of which here only a brief overview shall be given. The mode details and the transitions and fallback transitions are explained in detail later in Chap. 2. Figure 1.9 depicts the satellite orientation corresponding to the diverse modes.

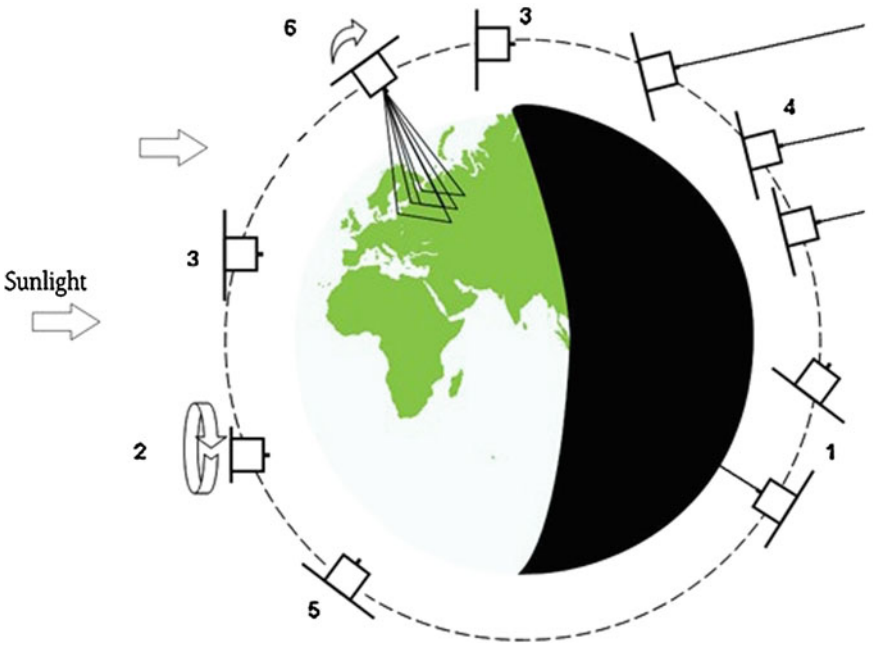


Fig. 1.9 Satellite control modes. © IRS, University of Stuttgart

The start-up and commissioning modes are
0 Launch Mode
1a Detumble Mode
1b Coarse Nadir Pointing Mode
which are not included in Fig. 1.9.

The safe spin stabilized orientation of the satellite is achieved in
2 Safe Mode
where the satellite is controlled only by means of sun sensors, magnetometers and magnetotorquers.

The nominal operational modes of the spacecraft are just differing w.r.t. the satellite's orientation and the ACS submodes—see also Fig. 1.9. In Idle Mode the S/C is Sun oriented three axis stabilized.

For payload operation scenarios the platform offers nadir pointing, target pointing and inertial pointing.
3 Idle Mode
4 Inertial Pointing Mode
5 Nadir Pointing Mode
6 Target Pointing (rollover) Mode

Detumble and Safe Mode
After separation from the launcher an FLP satellite will be spinning with rotational rates $|\omega B|$ of up to 10°/s. For that situation and in case of an emergency the De-tumble Mode is designed to reduce the rotational rate to a threshold of $|\omega B| = 0.4°/s$ by using only magnetotorquers as actuators.

The satellite will then be switched into the Safe Mode where it orients the negative z-axis (cf. Fig. 1.1) to the sun and builds up a spin of $|\omega B| = 2°/s$ around the z-axis to stabilize this orientation. This mode intends to provide a safe state in which power supply is secured and only a minimum of reliable sensors and actuators are used.

The Detumble Mode and Safe Mode are the fallback modes of the satellite they have to bring it to a safe state and have to be designed very robust in order to extend the satellite survival to a maximum in case of an on-board failure. Therefore in both Detumble and Safe Mode the spacecraft orientation is only coarse and due to limited sensors also the attitude information is only coarse.

Coarse Nadir-Pointing Mode
The Coarse Nadir-Pointing Mode is used during the satellite Commissioning Phase for taking the RWs into operation carefully and later during normal operation of the satellite it can be used for RWL servicing.

The mode is characterized by the spacecraft already having its high performance ACS sensors in operation (GPS, STR and potentially also the FOGs), whereas the RWLs are still off and the coarse nadir orientation is achieved by means of the

magnetotorquers. For the detailed sequence of taking spacecraft ACS units into operations please refer to Sect. 11.7 on platform commissioning tasks.

The mode avoids a complete reorientation of the satellite to the Sun and offers an improved antenna visibility during ground station passover compared to Safe Mode or Idle Mode which are both Sun-pointing modes.

Idle Mode

The Idle Mode is designed for recharging the batteries during normal operation when the FLP satellite is in idle with respect to ground commanding or between scheduled observations of the loaded mission timeline. In this mode the satellite is 3-axis controlled with the necessary sensors and actuators being operational and the satellite is oriented Sun-pointing with its solar arrays. The sun sensors and star trackers are used to determine the attitude and the fiber optic gyros are used to determine the body rates. Reaction wheels are used as actuators in this mode. In this mode the satellite is not spinning around its Z-Axis as in Safe Mode.

Operational Pointing Modes

To perform any payload operation task the FLP satellite has three different operational pointing modes:

- The first one is designed to orient the satellite's coordinate system to the inertial coordinate system so this mode is called Inertial Pointing Mode. For this mode no information about the current position is required since the inertial reference vectors are not depending on it. This mode is foreseen for astronomy experiments in case where a camera or telescope payload on an FLP mission is oriented towards a target relative to the fixed stars.
- In contrast the other two pointing modes, Nadir Pointing Mode and Target Pointing Mode, do need satellite position information since the direction to nadir, the earths center, or to any given ground target is depending on the satellites current position. Star trackers and fiber optic gyros are used to determine attitude and reaction wheels are used for the actuation in this mode. The nadir coordinate system is derived from the GPS output. So target pointing and nadir pointing with respect to attitude control laws is essentially the same except for a fixed target $\vec{t}_{nadir} = (0, 0, 0)^T$ in Nadir Pointing Mode.

1.11 Reaction Control Subsystem

The first FLP satellite—the "Flying Laptop"—is not yet equipped with a propulsion subsystem for orbit correction maneuvers. The satellite platform by design however allows to provide an additional CAN-Bus port by the OBC and thus to be able to equip it with an SSTL 150 electric propulsion subsystem (see [154]).

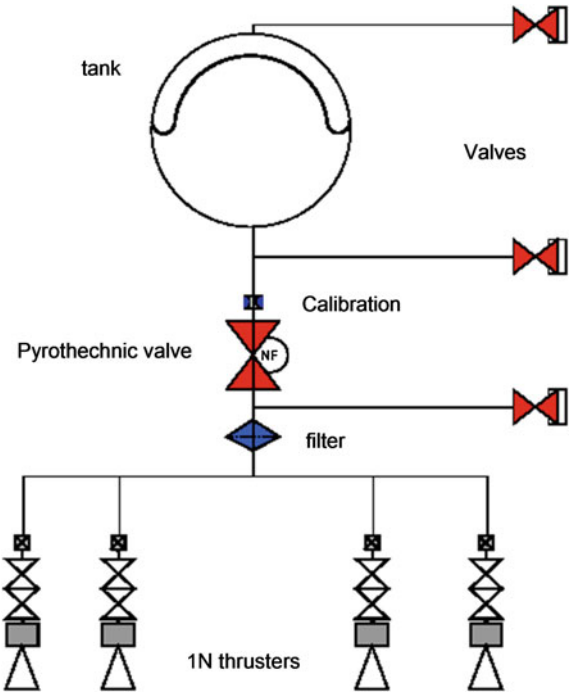


Fig. 1.10 Schematic of optional propulsion module for FLP Generation 2. © Airbus DS from [120]

In the frame of the ESA Space-based Space Surveillance study for a low-cost mission for space debris detection Airbus DS GmbH has studied the feasibility of the FLP Generation 2 platform which is based on FLP but enhanced with an upgraded OBC and a hydrazine propulsion subsystem (see [144]). The propulsion module is a mono-ergol hydrazine system from Airbus Defence and Space, Germany, operated in blow-down mode. It is used to perform orbit corrections requiring large ΔV (initial transfer, launcher dispersion correction, orbital parameters tuning) and for orbit maintenance (for which small ΔV in the cm/s range is required)—see Fig. 1.10.

1.12 Communication Subsystem

The "Telemetry, Tracking and Command" (TTC) subsystem—or TTC subsystem for short—consists of a transceiver manufactured by STT Systemtechnik, Munich, Germany (see [70]), and two omnidirectional turnstile antennas manufactured by SpaceQuest, Fairfax, VA, USA (see [72]). The frequencies used for the "Flying Laptop" are 2.2635 GHz for the downlink and 2.0835 GHz for the uplink. Details on the polarization, modulation etc. can be found in [77] and in Chap. 6.

As can be seen in Fig. 1.6 the system consists of two receivers and two transmitters, in combination referred to as the transceiver, one nominal and one redundant each. The nominal and redundant "paths" both use a separate diplexer for the connection to the antennas. A 3 dB hybrid coupler is used to split the signal to the two antennas.

The transceiver is connected to both redundant CCSDS-Boards of the OBC as shown in Fig. 1.6 which establishes the TM/TC connection to the OBC Processor-Boards (see Sect. 4.2). The TTC subsystem is powered by the PCDU. An additional connection between the transceiver and the OBC is used to acquire housekeeping on-board TM on the TTC subsystem status by the OBSW. This TM includes temperatures, state and validity flags and performance data.

1.13 Thermal Control Subsystem

The FLP platform is tailored for a passive thermal control subsystem (TCS). Most of the main body of the satellite is covered in Multi-Layer-Insulation (MLI) to minimize heat exchange with the space environment. On all surfaces not covered with MLI, radiators emit heat from the satellite while reflecting the sunlight. Each of the satellite modules possesses a pair of redundant electrical heating films controlled by thermostats. These heaters are used in case where the temperature within the modules is below the specified operating temperature range. Furthermore the most critical components (battery, OBC, PCDU, TTC) have their own heaters to assure permanent operability. Pt1000 thermistors are distributed across the satellite to measure the temperatures of the components. For the "Flying Laptop" as first satellite being built on the FLP Platform, it is divided into three segments which are structural horizontal plates—the service module, the core module and the payload module (see also Chap. 8 and Fig. 1.11). The segments are connected by vertical shear plate crosses in the center and two shear panels at the side of the cube (see Fig. 1.5).

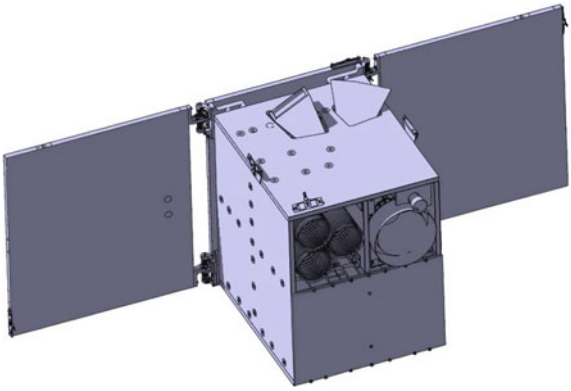
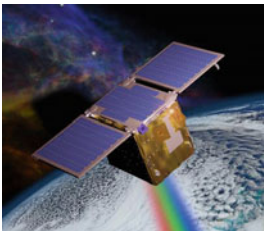


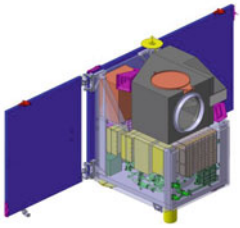
Fig. 1.11 Mechanical configuration of the Flying Laptop. © IRS, University of Stuttgart

1.14 Satellite Deorbiting at End of Life

The last platform element is the already mentioned "De-Orbiting-Mechanism" (DOM) to meet the 25-year maximum post-operational deorbiting requirement proposed by the United Nations Inter-Agency Space Debris Coordination Committee (IADC) and mandatory according to the European Code of Conduct on Space Debris Mitigation [1]. At end of the satellite lifetime the DOM releases a large-area foil shield to generate of a higher interaction with the residual atmosphere in orbit, resulting in a negative ΔV and thus a faster reentry of the satellite. The De-Orbiting-Mechanism is stowed within the launch adapter ring until its deployment at the end of the satellite mission. It is a flat square sail and is deployed using a non-explosive bi-metal switch.



© IRS, Uni Stuttgart

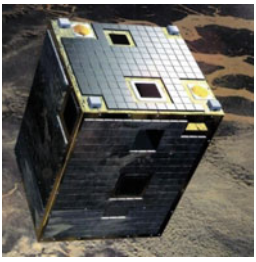


© Airbus DS

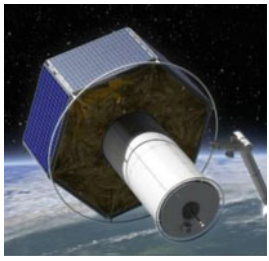
© CNES / Airbus DS

Configurations with deployable Solar Arrays

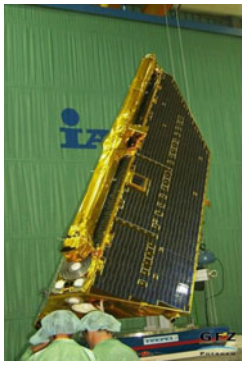
Configurations with body-mounted Solar Arrays



© ESA



© Airbus DS



© GFZ/IABG

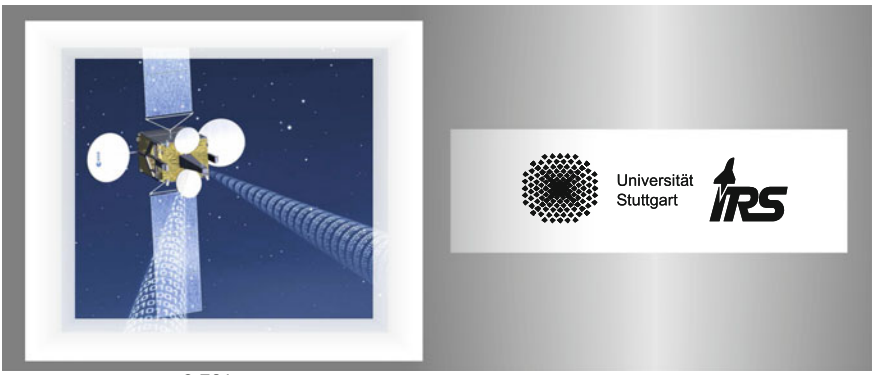
Fig. 1.12 Possible satellite body configurations for the FLP

1.15 Possible Mechanical Platform Configurations

The FLP microsatellite platform is mainly a functional platform comprising the power subsystem, the core data handling subsystem, the telecommand and telemetry subsystem for satellite control, the attitude and orbit control subsystem and the thermal control subsystem. The entire infrastructure can be implemented by design into a variety of structure variants, with deployable or body-mounted solar arrays. Some example layouts cited from other mission are depicted in Fig. 1.12.

Chapter 2
The FLP Platform Operability

Kai-Sören Klemich and Jens Eickhoff



© ESA

Abstract This chapter provides the basic overview on how an FLP-based spacecraft is operated. It explains the basic system modes and commanded transitions as well as fallback transitions. It explains the table-driven specification of mode-transitions. And it finally details all the command/control services which the platform provides—both standard PUS services, according to ECSS as well as defined private services. Command addressing and telemetry source definition via Virtual Channels and APIDs are also addressed.

Keywords Spacecraft modes · Attitude control subsystem modes · Mode transitions and control tables · Telecommand packet definition · Telemetry packet definition · Virtual channels · Application IDs · ECSS standard PUS services · Private PUS services

K.-S. Klemich (✉)
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: klemich@irs.uni-stuttgart.de

J. Eickhoff
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_2

2.1 Spacecraft Configuration Handling Concept

In this section the spacecraft configuration management mechanisms are explained. The configuration comprises the information on

- which devices
  - are active on and are
  - commanded into which mode and
- which software processes
  - are in which mode.

Thus the configuration defines the operational mode of the entire spacecraft. In order to better understand the configuration handling, some concepts of the OBSW are introduced hereafter.

2.1.1 System Mode Changes Using the OBSW Object Hierarchy

Before explaining the platform modes in more detail, the principle of mode transitions and transition commanding shall be explained. See Sect. 3.2 then for further details. The FLP OBSW is based on the object-oriented paradigm and features a hierarchy of "objects". In this hierarchy the *System* object is the top-level one. The modes of the spacecraft correspond 1:1 to the modes of the *System* object.

Below the *System* object (see Fig. 3.3 and Sect. 3.2.6), there exist the subsystem objects (see Sect. 3.2.5), *ACS_Subsystem* (Attitude Control), *PSS_Subsystem* (Power Supply), *TCS_Subsystem* (Thermal Control), *TTC_Subsystem* (Telemetry/Telecommand), *Payload*. *System* and subsystem objects also are called "high level" objects. Below these subsystem objects are the "low level" objects, which are

- device handlers (see Sect. 3.2.2) which are configuring and communicating with the hardware components,
- assemblies (see Sect. 3.2.3) which manage redundancies of device handlers in case of redundant devices like the RWL and
- controllers (see Sect. 3.2.4) which are software objects using data from the devices to calculate control variables, e.g. the *ACS_Controller* which performs the attitude control.

All of these objects have different operational modes, which are defined by a combination of a mode (ID) and a submode (ID). For device handlers the available modes are defined by the device's internal modes, for controllers the modes are defined by different control strategies, for high level objects the modes are defined as lists of modes of their sub-objects (explained later in Sect. 2.1.12).

If *System* is commanded to a certain mode (e.g. Idle Mode) it commands its sub-objects (i.e. the subsystem objects) to the necessary sub-object modes. For example *ACS* would be commanded to the ACS Idle Mode. Also *PSS* will be commanded to the required submode and similarly *TCS* etc.

The subsystems themselves then command their sub-objects, i.e. device handlers, assemblies and controllers to their necessary modes, e.g. for ACS Idle Mode the reaction wheels must be active and the *ACS_Controller* must be set to a mode in which it applies the Idle Mode control strategy. Following this concept, commanding the *System* object to a mode triggers mode changes of all lower level objects, i.e. of all objects.

2.1.2 Overview on the FLP Operational Modes

The diagram Fig. 2.1 shows the mode transitions of the spacecraft—in the OBSW represented by the modes of the *System* object. All of its lower modes, up to Idle Mode, are closely coupled to the respective *ACS_Subsystem* modes and the *ACS_Controller* modes which share the same mode names. The ACS modes are

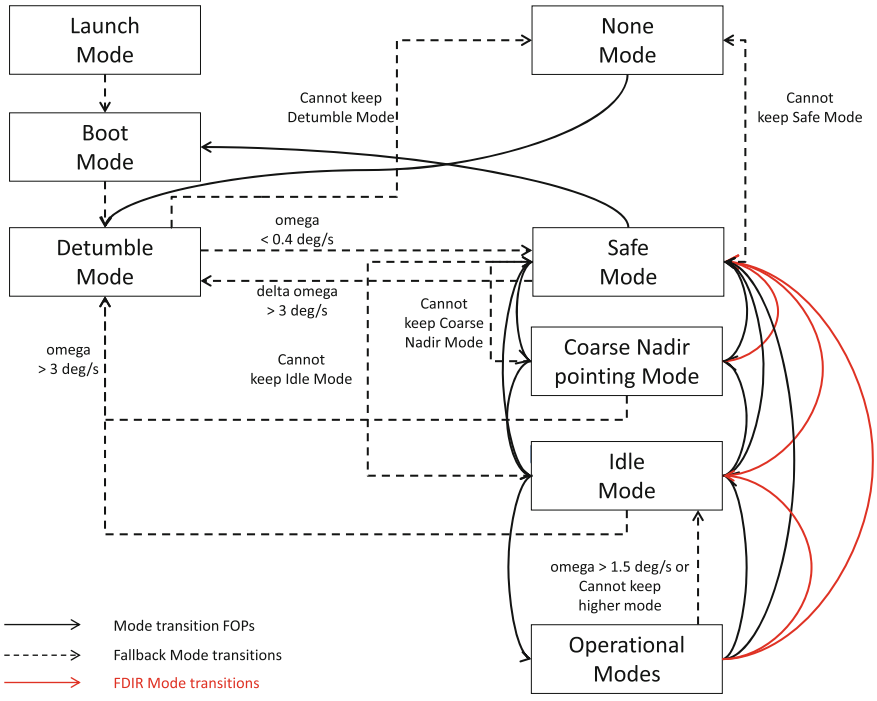


Fig. 2.1 FLP spacecraft modes and transitions. © IRS, University of Stuttgart

illustrated in Fig. 2.2. The different operational modes of the *System* object are explained in the following sections.

The "operational modes" cited in Figs. 2.1 and 2.2 include all payload modes as well as modes where no payload is activated but ACS is in a dedicated pointing mode e.g. target pointing for TM/TC ground station contact. There exist three kinds of mode transitions depicted in the diagram:

- The black continuous arrows show mode transitions which are available as " Flight Procedures" (see Chap. 15). Note that technically the mode machine in the OBSW does not restrict the modes which are allowed to be commanded. This restriction is induced by the availability of corresponding Flight Procedures. There are procedures to transit from lower to higher modes step by step, but also from Safe Mode to Idle Mode, and to from higher to lower modes in arbitrary steps, i.e. also from the highest modes to Safe Mode.
- The black dashed arrows show fallback mode transitions which are automatically triggered by the mode machine as fallback modes in case where a higher mode cannot be kept. Such a fallback transition is triggered in two cases:
  - A mode cannot be kept or cannot be reached, e.g. because the "Failure Detection, Isolation and Recovery" controller *FDIR* has marked a vital device for this mode as faulty or a vital device goes into an error mode or an

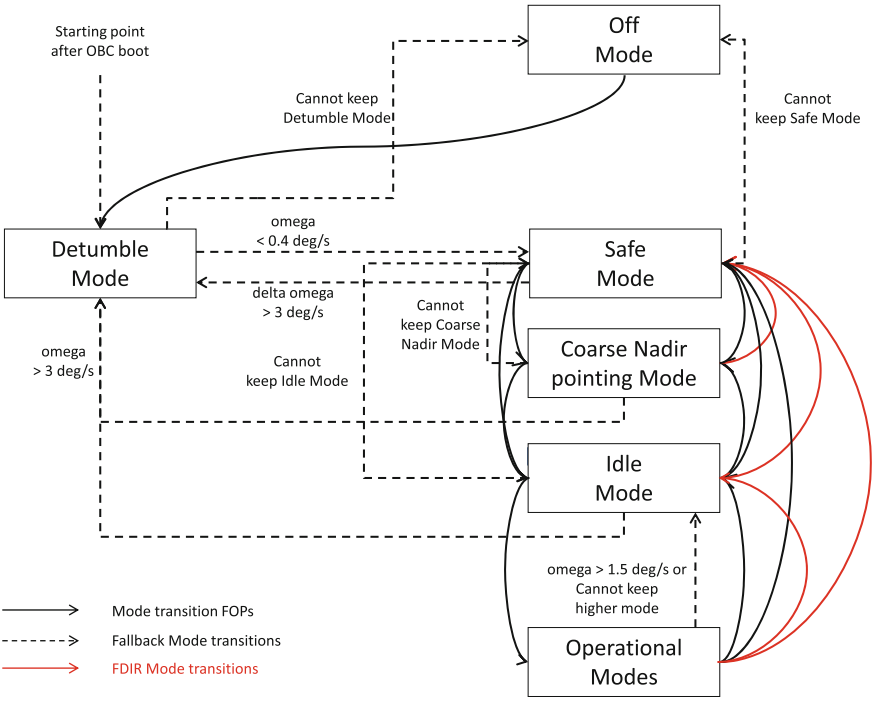


Fig. 2.2 FLP ACS modes and transitions. © IRS, University of Stuttgart

assembly (a logical group of identical devices—like for the 4 reaction wheels) cannot reach its target mode.
  - The *ACS_Controller* detects that the rotational rates or mispointings are too large for the respective mode. In such case, the *ACS_Controller* will notify the *ACS_Subsystem* that it cannot keep its mode which will trigger the fallback transition of the *ACS_Subsystem*, which in turn will trigger the depicted fallback transition of the System object (i.e. of the entire spacecraft).

- The red arrows show additional transitions triggered by FDIR from higher modes to a lower mode. These are induced in critical cases—for example when the battery State-of-Charge drops below certain limits (see Sect. 10.3).

The FLP is designed for satellites which are launched as secondary passengers and which fulfill the requirement of being entirely switched off until after separation from launcher.

Starting from Launch Mode, where the system is completely switched off, the OBSW performs its boot sequence (Boot Mode) and then starts to detumble the satellite (Detumble Mode). Once the overall rotational rate drops below ca. 0.4°/s, it orients the satellite's solar panels to the sun and spins the satellite about the sun axis (Safe Mode). If the estimated rotational rate diverges from the target rotational rate of 2°/s by more than 3°/s (named "delta omega" in the diagram), the system is set back to Detumble mode.

Higher modes than Safe Mode have to be commanded from ground, which is generally possible in both directions, i.e. from high to low and from low to high. In Coarse Nadir Pointing and Idle mode, the satellite system falls back to Detumble mode when the total rotational rate exceeds 3°/s. The diverse modes are explained in more detail from Sect. 2.1.3 onwards.

Once the satellite has reached an operational mode with a pointing control strategy, Idle Mode serves as an additional fallback option to avoid a transition to the Safe Mode in case of a failure with low severity. This fallback is triggered when the total rotational rate exceeds 1.5°/s or the higher mode cannot be kept, e.g. due to a failure in a payload device. If the failure is more severe, another fallback from Idle to Safe mode will be triggered.

As the flight modes of the satellite are closely related to the corresponding ACS control strategies, the ACS modes and transition diagram as illustrated in Fig. 2.2 looks rather similar.

2.1.3 Launch Mode

The Launch Mode is included in Fig. 2.1 although it does not really exist as operational mode. In the Launch Mode the PCDU is still off—except for the launcher separation detection circuit—and the Onboard Computer (OBC) is still off. Upon detection of the separation from the launcher, the spacecraft transits through the Boot Mode and switches to the Detumble/Safe Mode automatically.

2.1.4 Boot Mode

The Boot Mode in Fig. 2.1 also is no real mode but just an operational transition. It can however be commanded in case where a reboot of the satellite system becomes necessary. As soon as the PCDU's launcher separation circuit detects separation, the PCDU boots and thereafter initiates power-up of diverse equipment for the Safe Mode, namely OBC and Transceivers. The OBC/OBSW then perform the final steps to reach Detumble/Safe Mode. The overall satellite system boot at launcher separation is explained in more detail later in Sect. 4.4.

During boot the PCDU switches on both TTC receivers and both OBC CCSDS-Boards in hot redundancy to avoid single point failures. In addition it activates the power lines for magnetotorquers and magnetometers.

2.1.5 Detumble/Safe Mode

After separation from the launcher, the satellite tumbles with random rates. The actual rates depend on the used launch vehicle and the separation mechanism. During Launch and Early Orbit Phase (LEOP) and if the satellite starts tumbling during nominal operations, the orientation of the solar panels to the Sun and thus the generated energy cannot be predicted and changes very quickly. Therefore the active devices are reduced to a minimum. Only the magnetotorquers are used as actuators to detumble the satellite. Magnetometers and Sun sensors are used as sensor equipment in this mode. In order to achieve a sufficient torque, both, the nominal and redundant coils of the magnetotorquers are used. The solar panel deployment mechanisms will be activated for a maximum of 1 min during LEOP. More details are provided in Sect. 5.3 and in Sect. 11.6.1.

As soon as the total angular rate of the satellite decreases below 0.4°/s, the satellite ACS changes the control strategy from rate damping to Sun orientation (see Sect. 2.3) and spin-up around the −z axis up to a rate of approximately 2°/s (see Figs. 1.9 and 2.3).

The angle between the panel normal and the principal axis of inertia is approximately 8°. The accuracy of the measurement of the direction of the sun by the Sun sensors depends on the direction to the Earth. The maximum error is approximately 20°. Thus the maximum pointing error is in the range of 28°. However due to the spin its mean value is significantly smaller.

Besides the ACS components, only the most vital components are switched on, in order to avoid unnecessary power consumption. Thus, the battery can be recharged and the satellite is kept safely in operation even in case of a loss of control by ground. This mode is also entered in case of a contingency and can be automatically initiated by the FDIR or can be commanded from ground (see Fig. 2.1).

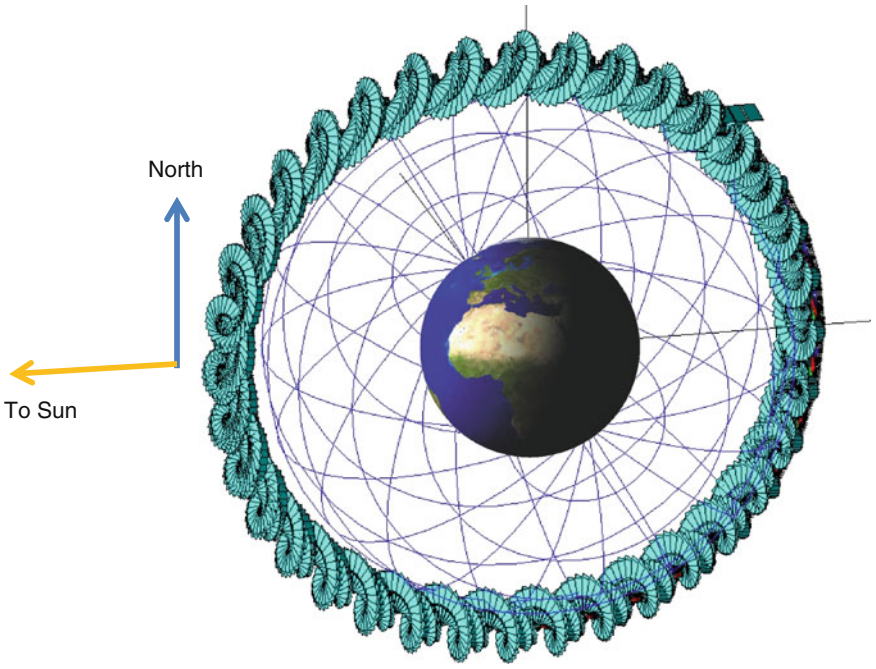


Fig. 2.3 Satellite safe mode. © IRS, University of Stuttgart

From Safe Mode upwards the normal transition is to Idle Mode. The Coarse Nadir Pointing Mode which is illustrated in Fig. 2.1 is used only during the spacecraft Commissioning Phase or during failure debugging (see Sect. 11.7).

2.1.6 Idle Mode

If the satellite operates nominally in Safe Mode and the battery is sufficiently recharged, the Idle Mode can be entered—commanded by telecommand (TC) from ground. In this mode, the solar panels are accurately oriented to the sun using the full capabilities of the ACS, using reaction wheels as actuators and GPS and STRs as sensors. Note that the attitude control algorithms avoid blinding of the STR by the Earth and the Sun at all times. In order to achieve this, the satellite is rotated once per orbit (see Fig. 2.4). No payload equipment is activated in Idle Mode.

From this state, the satellite can be brought into the operational modes like for Earth Observation (Nadir and Target Pointing Mode) or Inertial Pointing Mode for observation of celestial bodies. These operational modes are mission specific.

If an observation or data downlink period is finished and the battery needs to be recharged, the satellite will always be commanded back to Idle Mode.

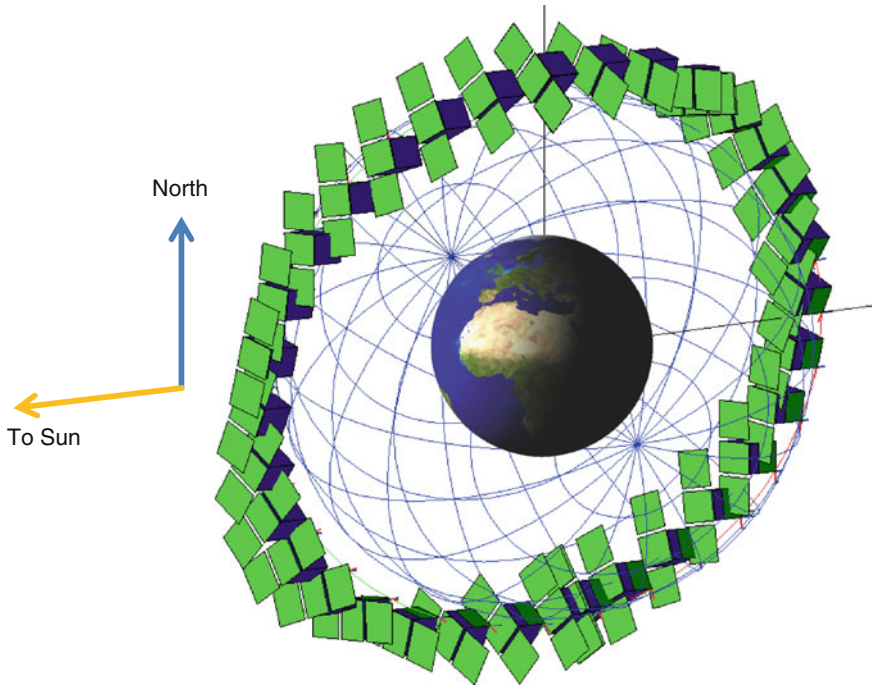


Fig. 2.4 Satellite idle mode. © IRS, University of Stuttgart

2.1.7 Coarse Nadir Pointing Mode

The Coarse Nadir Pointing Mode is an intermediate mode which is used during the spacecraft platform Commissioning Phase. Its control strategy is nadir pointing as this allows to achieve longer ground station contact times as in Safe Mode. However, in Coarse Nadir Pointing Mode the Earth orientation is achieved purely based on magnetotorquer control. The RWs are off when entering the mode from Safe Mode. The Coarse Nadir Pointing Mode is used to safely take the RWs into operations for the first time in the Commissioning Phase—see also Sect. 11.7.

The Coarse Nadir Pointing Mode can also be used for deeper spacecraft inspection in contingency operations to perform ground based FDIR or to perform Earth observation in case the RW cannot be used anymore.

2.1.8 Operational Modes—Ground Contact

In this mode the FLP points to and tracks a ground station antenna by target pointing. On S-band contact establishment the satellite starts to downlink life

telemetry (TM), playback telemetry and Event telemetry via its diverse TM virtual channels (for details see Sect. 2.2.1). The satellite can also be commanded by ground via telecommands.

Payload data is transmitted to ground using the science "Data Downlink System" (DDS) of the satellite which is mission specific and can be S-band (as for the first FLP mission of the University), X-Band (as foreseen for FLP Generation 2) or even Ka-Band.

With respect to ACS operation this mode represents a standard ACS target pointing mode as also used for target pointing payload operations—see next section.

2.1.9 Operational Modes—Payload Operations

In these modes, the full ACS capabilities are required to perform platform operations with payload pointing and mission product generation. The PLOC is switched on and it is possible to activate the other payload components according to mission timeline or life manual command. The Data Downlink System (DDS) is not active. As part of this mode, several different sub-modes are possible which depend on the dedicated mission. The FLP as platform supports by default:

- Earth observation in nadir pointing
- Earth observation in target pointing
- Inertial pointing for astronomical observations etc.

The activation of the necessary devices and the switching of the different required ACS modes is controlled by the OBC taking into account the required ACS mode transitions (see Sect. 2.1.12 and following). If the satellite starts spinning with a rate exceeding 1.5°/s during payload operations, it is automatically switched back to the Idle Mode (see Fig. 2.1).

2.1.10 None Mode

The None Mode is a fallback mode for extreme failure situations where the OBSW is not able to control the spacecraft anymore. Only the bare minimum equipment is kept operational which are PCDU, OBC and TTC Receivers and Transmitter. The heater strategy is reduced to battery thermal conditioning. No ACS control is performed. All payloads are disabled. The spacecraft attends ground support. If battery state of charge is decreasing further the PCDU will even shut off TTC and OBC and finally itself. For more details see Sects. 10.2 and 10.3.

2.1.11 Open Mode Concept

On the FLP a so-called open mode concept (as explained in [140]) is applied, which means that the devices and software functions turned on during a specific mode can be modified by telecommands from the ground station. The actual modification can be done for the system and subsystem objects—like the ACS in the example in Sect. 2.1.2. The implemented concept is very flexible and leads to a design, where no PUS[1] (see Sect. 2.4) Service 18 (On-board Control Procedures) is mandatory and no Event-Action Service 19.

For every system or subsystem mode a standard set of mode/submode combinations of the lower level objects is preprogrammed into the OBSW image in the OBC memory. For system level modes, these sets consist of subsystem modes and for subsystem modes, these sets consist of assembly, controller and device handler modes. This way, different configurations of the complete spacecraft are saved. Changes of these predefined mode sets as well as new mode definitions can be commanded at all times via loadable tables (see the following section).

Note that assemblies manage device handlers of redundant devices. For such devices, only the assembly mode is defined as part of the a.m. mode tables and the device modes are controlled by the assembly. This way, switching between redundant units can be performed by setting health flags rather than changing any predefined modes. If a nominal unit is set to faulty, its assembly will automatically switch it off and switch on the redundant unit.

To save the health statuses of all devices a Spacecraft State Vector is stored in dedicated memory areas on the I/O-Boards of the OBC (see Sects. 4.3.2 and 10.2.11). At launch, no component should be defective, so all flags are set to "HEALTHY". These can be changed automatically to "FAULTY" by the FDIR upon a detection of an error or by a telecommand from ground. However, to change the status of a device back to "HEALTHY", this is only possible by telecommand from ground—see also Sect. 2.4.16.

2.1.12 Mode Tables and Sequences Control Service

The satellite modes are defined in the OBSW by tables which contain mode/submode combinations and by sequences which contain sets of tables. In order to manipulate these tables and sequences, a dedicated on-board Packet Utilization Standard (PUS) Service is used, which is referred to as Service 202—see later in Sect. 2.4.17 for details.

[1]ESA Packet Utilization Standard (PUS)—see [13].

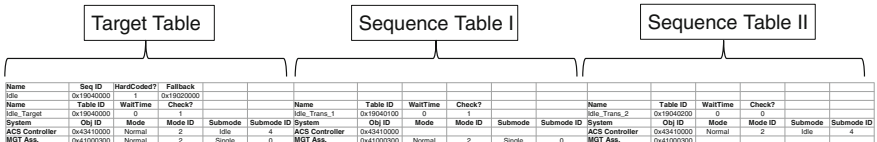
Satellite mode target tables are defined for the spacecraft, one for Safe Mode, one for Idle Mode etc. There also may be tables defined for mission specific modes. Mode table comprise n ≥ 0 mode entries. Each entry (a dedicated line in the table) consists of an object-ID as well as the object's mode ID and submode ID.

As example the Idle Mode target table contains mode/submode entries for the STR, the RW—defining these devices to be in On mode. The entry for the Safe Mode target table of the spacecraft will also comprise entry lines for STR and RWs —however defining these devices to be in Off mode.

The concept of mode transitions is explained at hand of Fig. 2.5 and Table 2.1:

A mode transition sequence includes one target table and one or multiple so-called sequence tables. Figure 2.5 depicts the principle of an overall layout as sketch—not for readability. Table 2.1 further below depicts the detailed table for the satellite system modes and transitions. Each subsystem chapter later in this book contains such a table set for the subsystem modes and transitions—see for example Table 6.12. For typesetting layout purposes in this book here the sequence tables are placed below the target table.

- The first table in a sequence—the target table—contains the system/subsystem target mode—i.e. into which status the system shall be brought—plus the intended modes of the child objects.
- A sequence table has in principal the same logic and layout as a target table, but corresponds to an intermediate stage only. Several sequence tables make up a switch sequence, and define each an intermediate mode status of the devices/ assemblies/controller.
- The mode transitions of the next sequence table are started when the transitions of the previous table are finished, plus a configurable time offset (WaitTime). For example the transition of the system to Boot Mode in Table 2.1 shows that 3 sequence tables are to be processed one by one when the system enters transition to boot mode. These sequence tables are executed in the defined order until the target mode settings are reached.
- One target table plus one or more sequence tables corresponds exactly to one mode of the controlled system/subsystem. E.g.—on satellite system level—there will be one such arrangement for Detumble/Safe Mode, one for Idle Mode etc.
- The actual spacecraft/subsystem mode is regularly checked against the target table once the mode transition sequence to the target has finished.



Name	Seq ID	HardCoded?	Fallback
Idle	0x19040000	1	0x19020000

Name	Table ID	WaitTime	Check?
Idle_Target	0x19040000	0	1

System	Obj ID	Mode	Mode ID	Submode	Submode ID
ACS Controller	0x43410000	Normal	2	Idle	4
MGT Ass.	0x41000300	Normal	2	Single	0
MGM Ass.	0x41000200	Normal	2	Fast	2
RW Ass.	0x41000100	Normal	2	Torque	1
STR Ass.	0x41000800	Normal	2	Default	7
FOG Ass.	0x41000400	Normal	2	Default	0
GPS Ass.	0x41000500	On	1	Single	0

Name	Table ID	WaitTime	Check?
Idle_Trans_1	0x19040100	0	1

System	Obj ID	Mode	Mode ID	Submode	Submode ID
ACS Controller	0x43410000				
MGT Ass.	0x41000300	Normal	2	Single	0
MGM Ass.	0x41000200	Normal	2	Fast	2
RW Ass.	0x41000100	Normal	2	Torque	1
STR Ass.	0x41000800	Normal	2	Default	7
FOG Ass.	0x41000400	Normal	2	Default	0
GPS Ass.	0x41000500	On	1	Single	0

Name	Table ID	WaitTime	Check?
Idle_Trans_2	0x19040200	0	0

System	Obj ID	Mode	Mode ID	Submode	Submode ID
ACS Controller	0x43410000	Normal	2	Idle	4
MGT Ass.	0x41000300				
MGM Ass.	0x41000200				
RW Ass.	0x41000100				
STR Ass.	0x41000800				
FOG Ass.	0x41000400				
GPS Ass.	0x41000500				

Fig. 2.5 Mode switch tables—sequence sketch. © IRS, University of Stuttgart

Table 2.1 FLP system level modes and transition table

	System Modes	**0x01000100**				
Commanding to None Mode						
#						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	None	0x79000000	1			
	Name	**Table ID**	**WaitTime**	**Check?**		
	None_Target	0x79000000	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**				
	Power Subsystem	**0x01000300**				
	TTC Subsystem	**0x01000400**				
	TCS Subsystem	**0x01000500**				
	Payload Subsystem	**0x01005000**				
#						
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	None_Trans1	0x79000100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	Off	0x19000000	None	0
	Power Subsystem	**0x01000300**	Off	0x49000000	None	0
	TTC Subsystem	**0x01000400**	Off	0x69000000	None	0
	TCS Subsystem	**0x01000500**	Off	0x59000000	None	0
	Payload Subsystem	**0x01005000**	Off	0x39000000	None	0

(continued)

Table 2.1 (continued)

Commanding to Boot Mode						
#						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Boot	0x79010000	1	0x79000000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	Boot_Target	0x79010000	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**				
	Power Subsystem	**0x01000300**				
	TTC Subsystem	**0x01000400**				
	TCS Subsystem	**0x01000500**				
	Payload Subsystem	**0x01005000**				
#						
Achieved by transitions						
	1					
	Name	**Table ID**	**WaitTime**	**Check?**		
	Boot_Trans1	0x79010100	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	Off	0x19000000	None	0
	Power Subsystem	**0x01000300**	Off	0x49000000	None	0
	TTC Subsystem	**0x01000400**	Off	0x69000000	None	0
	TCS Subsystem	**0x01000500**	Off	0x59000000	None	0
	Payload Subsystem	**0x01005000**	Off	0x39000000	None	0

(continued)

Table 2.1 (continued)

	2					
	Name	**Table ID**	**WaitTime**	**Check?**		
	Boot_Trans2	0x79010200	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**				
	Power Subsystem	**0x01000300**	Boot	0x49020000	None	0
	TTC Subsystem	**0x01000400**				
	TCS Subsystem	**0x01000500**				
	Payload Subsystem	**0x01005000**				
	3					
	Name	**Table ID**	**WaitTime**	**Check?**		
	Boot_Trans3	0x79010300	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**				
	Power Subsystem	**0x01000300**	Off	0x49000000	None	0
	TTC Subsystem	**0x01000400**				
	TCS Subsystem	**0x01000500**				
	Payload Subsystem	**0x01005000**				
#						
Commanding to Detumble/Safe Mode						
#						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Detumble	0x79020000	1	0x79000000		

(continued)

Table 2.1 (continued)

	Name	**Table ID**	**WaitTime**	**Check?**		
	Detumble_Target	0x79020000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	Detumble	0x19010000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**				
	TCS Subsystem	**0x01000500**	Survival	0x59020000	None	0
	Payload Subsystem	**0x01005000**	Off	0x39000000	None	0
#						
Achieved by transitions						
	1					
	Name	**Table ID**	**WaitTime**	**Check?**		
	Detumble_Trans1	0x79020100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	Detumble	0x19010000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**	Standby	0x69020000	None	0
	TCS Subsystem	**0x01000500**	Survival	0x59020000	None	0
	Payload Subsystem	**0x01005000**	Off	0x39000000	None	0
#						
Commanding to Coarse Nadir Mode						
#						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	CoarseNadir	0x79040000	1	0x79030000		

(continued)

Table 2.1 (continued)

	Name	**Table ID**	**WaitTime**	**Check?**		
	CoarseNadir_Target	0x79040000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	CoarseNadir	0x19030000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**				
	TCS Subsystem	**0x01000500**	Default	0x59010000	None	0
	Payload Subsystem	**0x01005000**				
#						
Achieved by transitions						
	1					
	Name	**Table ID**	**WaitTime**	**Check?**		
	CoarseNadir_Trans1	0x79040100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	CoarseNadir	0x69020000	None	0
	Power Subsystem	**0x01000300**	Default	0	None	0
	TTC Subsystem	**0x01000400**	Standby	0	None	0
	TCS Subsystem	**0x01000500**	Default	0	None	0
	Payload Subsystem	**0x01005000**	Off	0	None	0
#						
Commanding to Idle Mode						
#						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Idle	0x79050000	1	0x79030000		

(continued)

Table 2.1 (continued)

	Name	**Table ID**	**WaitTime**	**Check?**		
	Idle_Target	0x79050000	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	Idle	0x19040000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**				
	TCS Subsystem	**0x01000500**	Default	0x59010000	None	0
	Payload Subsystem	**0x01005000**				
#						
Achieved by transitions						
	1					
	Name	**Table ID**	**WaitTime**	**Check?**		
	Idle_Trans1	0x79050100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	Idle	0x19040000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**	Standby	0x69020000	None	0
	TCS Subsystem	**0x01000500**	Default	0x59010000	None	0
	Payload Subsystem	**0x01005000**	Off	0x39000000	None	0
#						
Commanding to Ground Contact Mode						
#						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	TargetPt_GS	0x79060000	1	0x79050000		

(continued)

Table 2.1 (continued)

	Name	**Table ID**	**WaitTime**	**Check?**		
	TargetPt_GS	0x79060000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	NadirPt	0x19060000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**				
	TCS Subsystem	**0x01000500**	Default	0x59010000	None	0
	Payload Subsystem	**0x01005000**				
#						
Achieved by transitions						
	1					
	Name	**Table ID**	**WaitTime**	**Check?**		
	TargetPt_GS_Trans1	0x79030100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	TargetPt	0x19060000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**	Comm	0x69010000	None	0
	TCS Subsystem	**0x01000500**	Default	0x59010000	None	0
	Payload Subsystem	**0x01005000**	Off	0x39000000	None	0
#						
Commanding to Nadir Pointing EO Mode						
#						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	NadirPt	0x790B0000	1	0x79050000		

(continued)

Table 2.1 (continued)

	Name	**Table ID**	**WaitTime**	**Check?**		
	NadirPt_Target	0x790B0000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	NadirPt	0x19050000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**				
	TCS Subsystem	**0x01000500**	Default	0x59010000	None	0
	Payload Subsystem	**0x01005000**				
#						
Achieved by transitions						
	1					
	Name	**Table ID**	**WaitTime**	**Check?**		
	NadirPt_Trans1	0x790B0100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	NadirPt	0x19050000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**	Standby	0x69020000	None	0
	TCS Subsystem	**0x01000500**	Default	0x59010000	None	0
	Payload Subsystem	**0x01005000**			None	0
#						
Commanding to Target Pointing EO Mode						
#						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	TargetPt	0x79130000	1	0x79050000		

(continued)

Table 2.1 (continued)

	Name	**Table ID**	**WaitTime**	**Check?**		
	TargetPt_Target	0x79130000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	TargetPt	0x19060000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**				
	TCS Subsystem	**0x01000500**	Default	0x59010000	None	0
	Payload Subsystem	**0x01005000**				
#						
Achieved by transitions						
	1					
	Name	**Table ID**	**WaitTime**	**Check?**		
	TargetPt_Trans1	0x79130100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	TargetPt	0x19060000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**	Standby	0x69020000	None	0
	TCS Subsystem	**0x01000500**	Default	0x59010000	None	0
	Payload Subsystem	**0x01005000**			None	0
#						
Commanding to Inertial Pointing Mode						
#						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	InertialPt	0x79070000	1	0x79050000		

(continued)

Table 2.1 (continued)

	Name	**Table ID**	**WaitTime**	**Check?**		
	InertialPt_Target	0x79030000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	InertialPt	0x19070000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**				
	TCS Subsystem	**0x01000500**	Default	0x59010000	None	0
	Payload Subsystem	**0x01005000**				
#						
Achieved by transitions						
	1					
	Name	**Table ID**	**WaitTime**	**Check?**		
	InertialPt_Trans1	0x79070100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Subsystem	**0x01000200**	InertialPt	0x19070000	None	0
	Power Subsystem	**0x01000300**	Default	0x49010000	None	0
	TTC Subsystem	**0x01000400**	Standby	0x69020000	None	0
	TCS Subsystem	**0x01000500**	Default	0x59010000	None	0
	Payload Subsystem	**0x01005000**			None	0
#						

- Thus a mode transition to a new target mode—by ground command, by fallback or triggered by FDIR—starts with the first sequence table of the mode. All necessary equipment/subsystem switching for such a sequence table happens semi-parallel. If a subsystem/equipment already is in the prescribed mode, no switching is necessary. If the CheckFlag is set, the status of all items is checked against the sequence table and if the check is successful, the next sequence table is processed. In a failure case the fallback is triggered. The target table defines the final state to be reached during the mode transition.
- All of these target/sequence tables for the satellite system level mode transitions are included in Table 2.1 at the end of this section.
- As an exception, on system level (Table 2.1), the subsystems are defined as to be commanded "subunits". In the standard subsystem tables like Tables 4.5, 5.8, 6.12, 7.16 and 8.6 the subitems are the subsystem controller and the assemblies and, for those elements where no assembly is available, the devices. Where assemblies are available, only the assemblies are commanded. They command their controlled lower level devices into the right mode.
- Each table has a "WaitTime" in seconds associated, which indicates how long the OBSW shall wait after a sequence table has been commanded until the next one is commanded. In addition, a check flag indicating whether or not the OBSW shall check the successful commanding of all objects in a sequence table can be set. The WaitTime and Check Flag are ignored for target tables. The "HardCoded" flag indicates whether the sequence shall be stored in the OBC safeguard memory to be available after reboot, or whether it s only a handcoded test sequence which will not be used later.
- Also, fallback modes may be given in Table 2.1, which correspond to the fallback transitions depicted in Fig. 2.1. Similarly fallbacks for ACS in the mode transition table of ACS (Table 7.16) correspond to the fallbacks in Fig. 2.2. A fallback in fact indicates a transition to another mode, which means another target/sequence table arrangement. In Table 2.1 for example the fallback of the boot mode is 0x09000000 which is the None Mode. The lowest operational mode of system or a subsystem has no further fallback. The transition to a fallback mode is triggered when an object reports that it cannot keep its mode any longer, e.g. due to a vital device marked faulty, or when a mode transition fails, e.g. due to a device handler not reaching its commanded mode.
- If the status of a subsystem or controller/assembly/device shall not be checked in a certain mode (so that Ground can switch it to arbitrary settings during failure inspection without violating cyclical status checks by the OBSW) the item simply is left out in the mode/sequence table. This feature is also used during platform commissioning, e.g. to test higher ACS devices in Safe Mode without violating any ACS subsystem mode target table mode definitions.

- If the CheckFlag is set, the status of all items is checked against the sequence table and if the check is successful, the next sequence table is processed. In a failure case the fallback is triggered. The target table defines the final state to be reached during the mode transition.
- A transition sequence then can be defined by means of Srv. (202,2) as for example provided in Table 2.1 further below for the satellite system level or Table 6.14 for the power subsystem.

It is important to note that in this processing scheme it may not be a concern from which mode the system switches to a new table. E.g. the **sequence** for the transition to Safe Mode should be designed in a way, that it is of no concern whether the system enters Safe Mode from a higher operational mode by FDIR detected failure, by ground command or by an automatic boot up transition during LEOP.

However, mode transitions may also be designed in such a way that they can only be used in one "direction", i.e. from higher to lower or vice versa. This becomes necessary when devices like GPS and STR are activated, which need a longer time to get a first measurement. A wrong usage of such transitions is avoided by the mission planning system in which the system mode is tracked and the respective Flight Procedures are provided (see Chaps. 14 and 15).

For such system mode transitions a large number of transitions on subsystem, assembly, controller and device level is performed (also in nominal cases) for which the OBSW generates corresponding Event TM. By this means ground can easily track all the transitions occurring on board.

Note that the TTC subsystem is left out of all system mode target tables in Table 2.1. This is because the system is planned to be operated in such a way that system and TTC subsystem modes are commanded separately. This avoids duplicate system mode definitions for Safe, Idle, Coarse Nadir Pointing and other modes, which can be used both with and without ground station contact. Leaving the TTC subsystem out of the target table allows commanding it independently from the system mode.

2.1.13 Spacecraft Equipment Operation Versus Modes

Table 2.2 (part 1–3) contains a detailed illustration of the diverse satellite subsystems and devices being off or in their diverse operational modes related to the spacecraft modes.

Table 2.2 FLP system modes

System	Object ID	None			0x79000000	Boot			0x79010000	Safe			0x79030000
	(hex)	Mode	#	Submode	#	Mode	#	Submode	#	Mode	#	Submode	#
FDIR Controller	**0x43460000**	Off	0	Off	0	Off	0	Default	0	Normal	2	Default	0
ACS Subsystem	**0x01000200**	Off	0x19000000			Off	0x19000000			Safe	0x19020000		
ACS Controller	**0x43410000**	Off	0	Off	0	Off	0	Off	0	Normal	2	Safe	2
MGT Ass.	**0x41000300**	Off	0	Off	0	Off	0	Off	0	Normal	2	Multi	1
MGM Ass.	**0x41000200**	Off	0	Off	0	Off	0	Off	0	Normal	2	Fast	2
RW Ass.	**0x41000100**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
STR Ass.	**0x41000800**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
FOG Ass.	**0x41000400**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
GPS Ass.	**0x41000500**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
CDH Subsystem	**0x01000600**	Default	0x29000000			Default	0x29000000			Default	0x29000000		
I/O-Board Ass.	**0x41010000**	On	1	Single	0	On	1	Single	0	On	1	Single	0
Power Subsystem	**0x01000300**	Off	0x49000000			Off	0x49000000			Default	0x49010000		
Power Controller	**0x43500000**	Off	0	Off	0	Off	0	Off	0	Normal	2	Default	0
PCDU	**0x44003200**	Normal	2	Idle	0	Normal	2	Idle	0	Normal	2	Idle	0
TTC Subsystem	**0x01000400**	Off	0x69000000			Off	0x69000000			Standby	0x69020000		
Communic. Controller	**0x43520000**	Off	0	Off	0	Off	0	Off	0	Normal	2	Standby	1
Rx Assembly	**0x41000600**	On	1	Default	0	On	1	Default	0	Normal	2	Default	0
Tx Assembly	**0x41000700**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
CCSDS-Board Ass.	**0x41100000**	On	1	Passive	0	On	1	Passive	0	On	1	Active	1
TCS Subsystem	**0x01000500**	Off	0x59000000			Off	0x59000000			Default	0x59010000		
TCS Controller	**0x43540000**	Off	0	Off	0	Off	0	Off	0	Normal	2	Auto	0
Payload Subsystem	**0x01005000**	Off	0x39000000			Off	0x39000000			Off	0x39000000		
Payload Controller	**0x43700000**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
PL1	**0x44501000**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
PL2	**0x44500F00**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
PL3	**0x44500B00**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
PL4	**0x44500E00**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
DDS Ass.	**0x41010200**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
PLOC Ass.	**0x41010100**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0

(continued)

Table 2.2 (continued)

System	Object ID	CoarseNadir			0x79040000	Idle			0x79050000	TargetPt_GS			0x79060000
	(hex)	Mode	#	Submode	#	Mode	#	Submode	#	Mode	#	Submode	#
FDIR Controller	**0x43460000**	Normal	2	Default	0	Normal	2	Default	0	Normal	2	Default	0
ACS Subsystem	**0x01000200**	CoarseNadir	0x19030000			Idle	0x19040000			TargetPt	0x19060000		
ACS Controller	**0x43410000**	Normal	2	CoarseNadir	3	Normal	2	Idle	4	Normal	2	TargetPt	6
MGT Ass.	**0x41000300**	Normal	2	Single	0	Normal	2	Single	0	Normal	2	Single	0
MGM Ass.	**0x41000200**	Normal	2	Fast	2	Normal	2	Fast	2	Normal	2	Fast	2
RW Ass.	**0x41000100**	Off	0	Off	0	Normal	2	Torque	1	Normal	2	Torque	1
STR Ass.	**0x41000800**	Normal	2	Default	7	Normal	2	Default	7	Normal	2	Default	7
FOG Ass.	**0x41000400**	Off	0	Off	0	Normal	2	Default	0	Normal	2	Default	0
GPS Ass.	**0x41000500**	On	1	Single	0	On	1	Single	0	On	1	Single	0
CDH Subsystem	**0x01000600**	Default	0x29000000			Default	0x29000000			Default	0x29000000		
I/O-Board Ass.	**0x41010000**	On	1	Single	0	On	1	Single	0	On	1	Single	0
Power Subsystem	**0x01000300**	Default	0x49010000			Default	0x49010000			Default	0x49010000		
Power Controller	**0x43500000**	Normal	2	Default	0	Normal	2	Default	0	Normal	2	Default	0
PCDU	**0x44003200**	Normal	2	Idle	0	Normal	2	Idle	0	Normal	2	Idle	0
TTC Subsystem	**0x01000400**	Standby	0x69020000			Standby	0x69020000			Comm	0x69010000		
Communic. Controller	**0x43520000**	Normal	2	Standby	1	Normal	2	Standby	1	Normal	2	Comm	2
Rx Assembly	**0x41000600**	Normal	2	Default	0	Normal	2	Default	0	Normal	2	Default	0
Tx Assembly	**0x41000700**	Off	0	Off	0	Off	0	Off	0	On	1	Default	0
CCSDS-Board Ass.	**0x41100000**	On	1	Active	1	On	1	Active	1	On	1	Active	1
TCS Subsystem	**0x01000500**	Default	0x59010000			Default	0x59010000			Default	0x59010000		
TCS Controller	**0x43540000**	Normal	2	Auto	0	Normal	2	Auto	0	Normal	2	Auto	0
Payload Subsystem	**0x01005000**	Off	0x39000000			Off	0x39000000			Off	0x39000000		
Payload Controller	**0x43700000**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
PL1	**0x44501000**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
PL2	**0x44500F00**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
PL3	**0x44500B00**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
PL4	**0x44500E00**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
DDS Ass.	**0x41010200**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
PLOC Ass.	**0x41010100**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0

(continued)

Table 2.2 (continued)

System	Object ID	InertialPT			0x79070000	NadirPt			0x790B0000	TargetPt			0x79130000
	(hex)	Mode	#	Submode	#	Mode	#	Submode	#	Mode	#	Submode	#
FDIR Controller	**0x43460000**	Normal	2	Default	0	Normal	2	Default	0	Normal	2	Default	0
ACS Subsystem	**0x01000200**	InertialPt	0x19080000			NadirPt	0x19050000			TargetPt	0x19060000		
ACS Controller	**0x43410000**	Normal	2	InertialPt	7	Normal	2	NadirPt	5	Normal	2	TargetPt	6
MGT Ass.	**0x41000300**	Normal	2	Single	0	Normal	2	Single	0	Normal	2	Single	0
MGM Ass.	**0x41000200**	Normal	2	Fast	2	Normal	2	Fast	2	Normal	2	Fast	2
RW Ass.	**0x41000100**	Normal	2	Torque	1	Normal	2	Torque	1	Normal	2	Torque	1
STR Ass.	**0x41000800**	Normal	2	NEO	71	Normal	2	Default	7	Normal	2	Default	7
FOG Ass.	**0x41000400**	Normal	2	Default	0	Normal	2	Default	0	Normal	2	Default	0
GPS Ass.	**0x41000500**	On	1	Single	0	On	1	Single	0	On	1	Single	0
CDH Subsystem	**0x01000600**	Default	0x29000000			Default	0x29000000			Default	0x29000000		
I/O-Board Ass.	**0x41010000**	On	1	Single	0	On	1	Single	0	On	1	Single	0
Power Subsystem	**0x01000300**	Default	0x49010000			Default	0x49010000			Default	0x49010000		
Power Controller	**0x43500000**	Normal	2	Default	0	Normal	2	Default	0	Normal	2	Default	0
PCDU	**0x44003200**	Normal	2	Idle	0	Normal	2	Idle	0	Normal	2	Idle	0
TTC Subsystem	**0x01000400**	Standby	0x69020000			Standby	0x69020000			Standby	0x69020000		
Communic. Controller	**0x43520000**	Normal	2	Standby	1	Normal	2	Standby	1	Normal	2	Standby	1
Rx Assembly	**0x41000600**	Normal	2	Default	0	Normal	2	Default	0	Normal	2	Default	0
Tx Assembly	**0x41000700**	Off	0	Off	0	Off	0	Off	0	Off	0	Off	0
CCSDS-Board Ass.	**0x41100000**	On	1	Active	1	On	1	Active	1	On	1	Active	1
TCS Subsystem	**0x01000500**	Default	0x59010000			Default	0x59010000			Default	0x59010000		
TCS Controller	**0x43540000**	Normal	2	Auto	0	Normal	2	Auto	0	Normal	2	Auto	0
Payload Subsystem	**0x01005000**	InertialPt	0x39090000			NadirPt	0x39010000			TargetPt	0x39070000		
Payload Controller	**0x43700000**	Normal	2	Default	0	Normal	2	Default	0	Normal	2	Default	0
PL1	**0x44501000**	Off	0	Off	0	Off	0	Default	0	Normal	2	Default	0
PL2	**0x44500F00**	Off	0	Off	0	Normal	2	Default	0	Off	0	Default	0
PL3	**0x44500B00**	Off	0	Off	0	Normal	2	Default	0	Off	0	Default	0
PL4	**0x44500E00**	Normal	2	Default	0	Off	0	Default	0	Off	0	Default	0
DDS Ass.	**0x41010200**	Off	0	Off	0	Off	0	Default	0	Off	0	Default	0
PLOC Ass.	**0x41010100**	Off	0	Off	0	Normal	2	Default	0	Normal	2	Default	0

2.2 Spacecraft Telecommand and Telemetry Structure

Figures 2.6 and 2.7 illustrate the structure of the FLP telecommands and telemetry respectively from packet layer to transmission layer according to the CCSDS standards [15–27]. The exact Virtual Channel settings, Packet length settings, CRC usage etc. can be taken from these figures.

2.2.1 CCSDS Protocol Addressing

As defined in the CCSDS standards [18, 27], several different IDs are defined to identify the channel and the origin of a TM packet or the destination of a TC packet (please also refer to the Figs. 2.6 and 2.7). The highest level of the channels is the physical channel, identified by its name. Physical channels are typically used for different ground/space links, i.e. for missions with multiple spacecraft and/or multiple ground stations.

For the FLP mission "Flying Laptop" there is only one spacecraft and only one ground station involved at a time (either Weilheim or Stuttgart) so there is no need to assign a specific ID to the Physical Channel.

The physical channel consists of several Master Channels identified by the Master Channel ID (MCID). The MCID consists of the Transfer Frame Version Number (TFVN, see Sect. 2.5) and the Spacecraft ID (SCID, see Sect. 2.2.2), both of which are given in the Transfer Frame Header. As for the FLP platform the TFVN is always "00" (in binary).

A Master Channel consists of several Virtual Channels, identified by the Global Virtual Channel ID (GVCID). The GVCID consists of the MCID and the Virtual Channel ID (VCID). As mentioned before, the MCID is fix for the "Flying Laptop" mission. Different VCID sets exist for uplink and downlink:

Uplink
For the uplink the VCID determines whether the command is a High Priority Command (HPC) or not and to which OBC CCSDS-Board the command is directed (nominal or redundant). The FLP platform uplink VCID definitions are provided in Table 2.3.

For the uplink only—according to the CCSDS standard—a Virtual Channel can consist of several Multiplexer Access Point (MAP) channels, identified by a Global Multiplexer Access Point ID (GMAPID) according to [27]. The GMAPID consists of the GVCID and the MAP ID.

For the FLP platform, there are no MAPs defined, so the MAPID is fixed to 0 for all commands. The TC routing is performed through the APID mechanism provided by the Packet Utilization Standard (PUS)—see Sect. 2.4. The APID definitions are provided in Sect. 2.3.

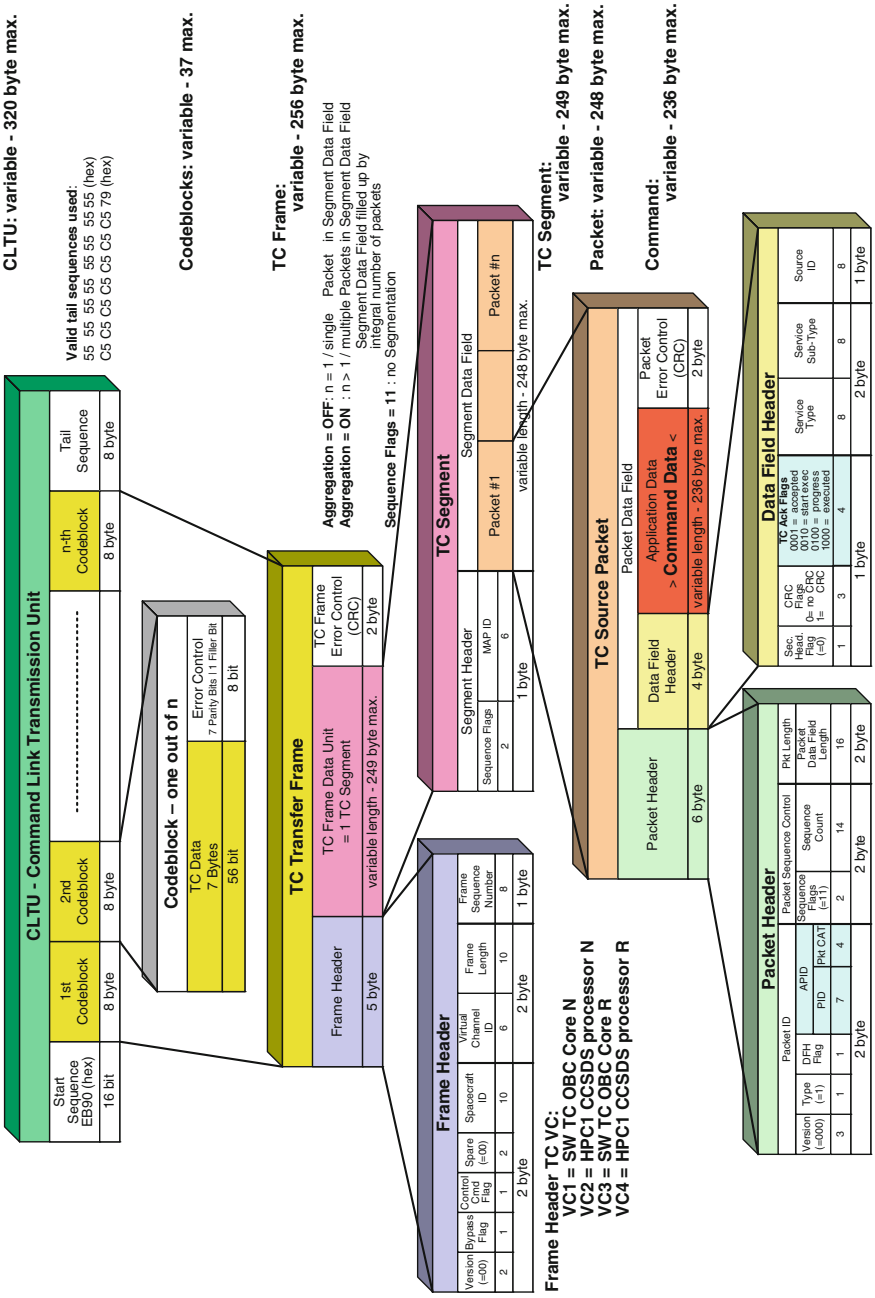


Fig. 2.6 Telecommand packet definition. © IRS, University of Stuttgart, Template © Airbus DS

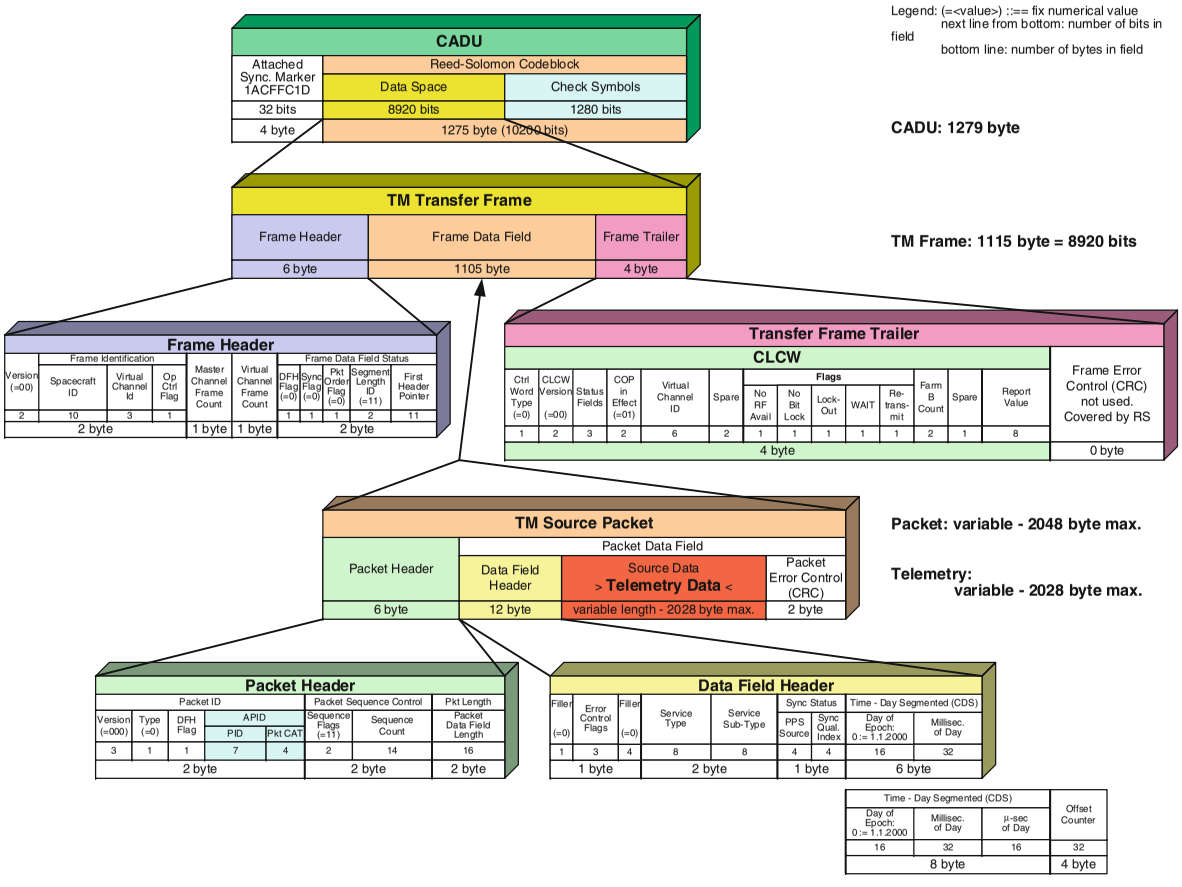


Fig. 2.7 Telemetry packet definition. © IRS, University of Stuttgart, Template © Airbus DS

Table 2.3 Uplink VCIDs overview

TC VCIDs	CCSDS-Board	Command Taype
VC1	CCSDS N	Nominal High Priority TC
VC2		Nominal standard TC
VC3	CCSDS R	Redundant High Priority TC
VC4		Redundant standard TC

Table 2.4 Downlink VCIDs overview

TM VCIDs	Command Type
VC0	Live TM
VC1	Stored TM: Srv 3 Housekeeping/Diagnistic packets
VC2	Stored TM: Srv 5 Event packets
VC3	Stored TM: all other packets

Downlink
For the downlink the VCID determines the origin of the TM, i.e. whether it is live TM or not and if not, from which packet store it originates (see Sects. 2.4.11 and 4.3.2). Thus the VC also determines the priority of the submitted TM frames and thus the routing priority in the CCSDS-Board multiplexer. Four Virtual Channels can be determined according to the origin of the TM, as summarized in Table 2.4.

2.2.2 Spacecraft ID

Each spacecraft transmitting CCSDS protocol in the S-band range reserved for space communication, needs an official registered spacecraft ID number. The official CCSDS Spacecraft ID (SCID) assigned to the FLP mssion “Flying Laptop” is “001001011101” in binary or “25D” in hexadecimal numbers. It will appear in every TM- and TC-Frame Header. It was requested by M. Pilgram of DLR/GSOC and was initially assigned on August 1st, 2011. The entry in the official database of registered spacecraft Space Assigned Number Authority (SANA) [46] is provided in Table 2.5:

Table 2.5 Spacecraft ID overview

Spacecraft Name	Channel	Version	Id (hex)	Requestor Name	Requestor Affiliation	Requestor Affiliation Country	Assigned Date	Assignor	Status
FLYINGLAPTOP	TC, TLM	1	25D	M. Pilgram	DLR	DE	2011-08-01	JGG	Assigned

2.2.3 System Authentication

The FLP core data handling subsystem (Core DHS) provides a simplified authentication concept to prevent satellite hijacking. This is developer intellectual property and is not subject to this published FOM release.

2.3 Application Process ID Definitions (APIDs)

The CCSDS Space Packet Protocol Standard [20] defines the lower protocol structure, namely the TM or TC packets (see Sect. 2.5). Within this standard, the Application Process ID (APID) is defined as the main ID to define the origin of a TM packet respectively the target of a TC packet. The APID range 2040–2047 is however reserved by certain standards (see [21]).

The APID can be used to address different parts of the OBSW or different devices. It needs to be noted that a device which has its own APID must be a PUS terminal. Thus it has to be able to receive/interprete and to send PUS compatible packets. This requires a certain integrated "intelligence" in such a device. For the FLP only the STR has this capability and thus its own APID. All other devices are controlled by the OBSW and addressed using the OBSW APID.

Inside the FLP OBSW the diverse processes do not have individual APIDs—so there exist no diverse Process IDs (PRID) and Categories (PCAT). The entire OBSW has a single APID. For the defined APIDs see Table 2.6:

Table 2.6 FLP APIDs definition

Process/Hardware	APID (dez)	APID (bin)	Remarks
Idle Packet	2047	11111111111	Defined in [20]
OBSW general	53	00000110101	
Star Tracker	1937	11110010001	
Simulator	1804	11100001100	Fixed, as compiled in simulator kernel (source code IP of Airbus DS)
TMTCFE Remote Control	1540	11000000100	Configurable

2.4 PUS Tailoring Concept

The ECSS Standard ECSS-E-70-41A "Ground systems and operations—Telemetry and telecommand packet utilization" [13] defines the so-called "Packet Utilization Standard" (PUS) Services. Within these PUS Services generic functionalities are defined on how telecommands and telemetry can be utilized for spacecraft control, and for on-board and on-ground data handling, telecommand verification etc. Each service includes several subservices which can either be a request to a service provider, i.e. an on-board application process, or a report from the provider. Often a report is defined as a "reply" to a service request, the request being a telecommand (TC) which is uplinked and the report being a telemetry (TM) packet which is downlinked.

The ECSS standard defined services are numbered from 1 to 19, with the services 7, 10 and 16 currently being reserved but unused [13]. Not all services and subservices defined by the ECSS need to be implemented for a dedicated mission. Additional services may be defined by a spacecraft manufacturer for special mission purposes with service numbers above 127. For self-defined services and for standard services also additional subservices may be defined by a spacecraft manufacturer. Several of these additional services are defined for the FLP. For each service a "minimum capability" set of subservices is defined, which are considered mandatory if the service itself is implemented.

Due to the generic nature of the PUS standard, there are no dedicated services for specific equipment operations, but standard PUS Services can be extended for such purposes. Examples for such extensions are specific payload functions, specific failure handling functions or specific onboard data handling functions.

For these reasons of mission specific services and mission specific additional subservices the PUS standard needs to be tailored for each mission. In the following sections all services and all corresponding subservices are described which are implemented in the FLP Core Data Handling Subystem (Core DHS). For each service used on the FLP, a table of all subservices is provided at the end of each section. The implementation of the telecommand and telemetry packets and variables in the Flight Control System's database—the so-called Mission Information Base (MIB)—is covered in detail in Chap. 16.

The FLP Onboard Software by default supports the following standard PUS Services:

- Service 1: Acknowledge Service
- Service 2: Device Commanding Service
- Service 3: Housekeeping Service
- Service 5: Event Reporting Service
- Service 6: Memory Management Service
- Service 8: Function Service
- Service 9: Time Management Service
- Service 11: On-board Operations Scheduling Service
- Service 12: On-Board Monitoring Service

- Service 15: On-board Storage and Retrieval Service
- Service 17: Test Service
- Service 20x: Self-defined services for mode commanding.

The following sections describe the FLP PUS implementation. For the detailed TC/TM/Event lists refer to the annex Sect. 17.3.

2.4.1 PUS-Service 1—Telecommand Verification

Service 1 defines the reporting of telecommand reception and execution in various stages (see also Table 2.7). In general, telecommands shall be verified upon reception on board the satellite. When a command is received, a Telecommand Acceptance Report is sent to Ground—Service (1,1) for successful reception—respectively (1,2) in case of failed reception or corrupted command. After a command has been successfully executed or in case where an error occurred during the execution, a Telecommand Execution Completed Report—Service (1,7) respectively (1,8)—is be generated. The execution of more complex commands can be acknowledged by sending execution start and execution progress success and failure reports—Service (1,3)—(1,6). All of these report types are implemented in the FLP Core DHS.

Table 2.7 PUS-Service 1—telecommand verification

Nr.	Description	TM/TC
(1,1)	Telecommand Acceptance Report—Success	TM
(1,2)	Telecommand Acceptance Report—Failure	TM
(1,3)	Telecommand Execution Started Report—Success	TM
(1,4)	Telecommand Execution Started Report—Failure	TM
(1,5)	Telecommand Execution Progress Report—Success	TM
(1,6)	Telecommand Execution Progress Report—Failure	TM
(1,7)	Telecommand Execution Completed Report—Success	TM
(1,8)	Telecommand Execution Completed Report—Failure	TM

2.4.2 PUS-Service 2—Device Command Distribution

PUS-Service 2—see Table 2.8—is used for direct commanding of on-board devices (equipment—like a star tracker, a reaction wheel etc.). It is used mainly for testing, internal OBSW commanding and contingency situations. During nominal operations, Service 2 commands can be used e.g. to request and report settings or other data from devices, which is not part of the housekeeping TM or to trigger certain rarely used functions:

Table 2.8 PUS-Service 2—device command distribution

Nr.	Description	TM/TC
(2,128)	Distribute Raw Device Commands	TC
(2,129)	Raw Device Command Report	TM
(2,130)	Distribute Direct Device Commands	TC
(2,132)	Distribute Channel Switch Command	TC
(2,133)	Device Wiretapping Request	TC
(2,134)	Device Wiretapping TC Report	TM
(2,135)	Direct Device Command Report	TM

Subservice (2,128) is used to command a device (on-board equipment) using a command structure which is understood by the device itself ("raw commanding"). The command is directly routed to the device by the device handler, if the handler is in raw mode (see Sect. 3.2.2). Subservice (2,129) is the report answer to the (2,128) request, containing the actual answer packet from the commanded device. It is also directly routed to the ground by the device handler without any interpretation.

Due to the complexity of the PCDU, this functionality is mainly used during the PCDU verification, which is why only the Service (2,129) reports from the PCDU are interpreted by the Mission Control System (MCS).

Subservice (2,130) is used to command a device by a simpler command, which is "known" by the OBSW ("direct commanding"). Thus, it is not in a format understood by the target device, but is interpreted by the device handler in the OBSW. Furthermore, the reply packet of the device is also interpreted by the device handler. To use this Subservice, the device handler must be in On or Normal mode (see Sect. 3.2.2). Note that not all commands of a given device are necessarily implemented in the OBSW so that they can be sent by Service (2,130). For the other commands, Service (2,128) must be used.

TM answers to direct commands of type (2,130) containing requests of TM from devices are downlinked as Subservice (2,135). This can be used to downlink TM parameters, which are not requested by the device handlers on a regular basis. This includes mainly status variables which do not change frequently (as for example bi-stable relays states in the PCDU) and which are not used by any on-board controller. All values requested by service (2,130) are not written into the OBSW *datapool* (see Sect. 4.3) and cannot be downlinked using PUS Service 3 (Housekeeping).

In general, Service 2 is mainly used for troubleshooting during testing and possibly on orbit. However, using Service (2,130) commands to receive (2,135) TM is part of nominal operations.

Subservice (2,133) is used to enable and disable a so-called "wiretapping mode" of a device handler. In this mode, the device handler downlinks all communication between a device and the device handler. Commands from the handler to the device are downlinked as Service (2,134) packets and answer packets from the device as Service (2,129) (see above). This service is intended to be used for debugging purposes during tests and possibly in orbit. Wiretapping can be activated in any

device handler mode (see Sect. 3.2.2). However, it needs to be used carefully as large amounts of data might be generated, possibly spamming the downlink channel, especially when the handler is in on or normal mode. It can be useful to enable the service shortly before a to be debugged command is sent, and to disable it shortly after command execution and error recording. Generally, the packets are however not interpreted within the Mission Control System but their TM (see service (2,135) explained below) needs to be interpreted manually.

Service (2,3) is a subservice which is used on board of standard spacecraft for emergency commanding. They are used to send pulse commands to the so-called Command Pulse Distribution Unit (CPDU) to switch units on or off, directly from ground (see also [140]). Due to the special Combined Data and Power Management Infrastructure (CDPI) on the FLP here the High Priority Commands (HPC) here are distinguished by their virtual channels, provide more sophisticated features and are not PUS compliant. Thus there is no need for implementation of Service (2,3).

2.4.3 PUS-Service 3—Housekeeping and Diagnostic Reporting

This service provides the possibility to report housekeeping (HK) and diagnostic data to ground. The term HK data refers to nominal telemetry in the PUS, whereas diagnostic data is telemetry sent in case of anomalies occurring within the HK data.

Both the nominal HK and diagnostic data can be downlinked in periodic or filtered mode. In periodic mode, the packets are generated at a predefined sampling rate, whereas in filtered mode, packets are only generated if predefined thresholds are exceeded. However, the filtered mode is not used on the FLP.

Within the service a list of HK and diagnostic parameter reports is maintained (see Table 2.9), in which sampling rates and parameters to be sampled are defined. The first eight subservices of this service will be used on the FLP in order to define new (3,1)/(3,2) or to erase old (3,3)/(3,4) HK or diagnostic parameter reports and to enable (3,5)/(3,7) or to disable (3,6)/(3,8) the generation of HK and diagnostic parameter reports.

Subservices (3,9) to (3,12) provide the possibility to downlink the HK and diagnostic parameter report definitions in order to double check the definitions actually used on board.

Subservices (3,25)/(3,26) define reports for HK and diagnostic parameters. These subservices are the minimum capability of the service.

There exist predefined housekeeping packets for the FLP, containing the *datapool* parameters of a single subsystem or device. Every *datapool* variable can be found in at least one (3,25) packet.

Table 2.9 PUS Service 3—housekeeping and diagnostics data handling

(3,1)	Define new Housekeeping Parameter Report	TC
(3,2)	Define New Diagnostic Parameter Report	TC
(3,3)	Clear Housekeeping Parameter Report Definitions	TC
(3,4)	Clear Diagnostic Parameter Report Definitions	TC
(3,5)	Enable Housekeeping Parameter Report Generation	TC
(3,6)	Disable Housekeeping Parameter Report Generation	TC
(3,7)	Enable Diagnostic Parameter Report Generation	TC
(3,8)	Disable Diagnostic Parameter Report Generation	TC
(3,9)	Report Housekeeping Parameter Report Definitions	TC
(3,10)	Housekeeping Parameter Report Definitions Report	TM
(3,11)	Report Diagnostic Parameter Report Definitions	TC
(3,12)	Diagnostic Parameter Report Definitions Report	TM
(3,25)	Housekeeping Parameter Report	TM
(3,26)	Diagnostic Parameter Report	TM

2.4.4 PUS-Service 4—Parameter Statistics Reporting

This service provides the possibility to report only statistics of certain TM (minimum, maximum, mean values, standard deviation) in order to reduce the amount of data to be downlinked. This service is not implemented in the Core DHS of the FLP.

2.4.5 PUS-Service 5—Event Reporting

This service is used to report event TM to Ground. Event TM can be generated in case of anomalies and errors or if certain autonomous on-board actions have been carried out or milestones have been reached. There are four different kinds of event reports defined—subservices (5,1) to (5,4) for

- normal/progress reports for autonomous on-board actions or milestones—Service (5,1) and for
- error/anomaly reports with three different levels of severity, namely low, medium and high—Services (5,2) to (5,4).

Subservices (5,5) and (5,6) are used to activate/deactivate the Event generation on board in flight—individually for the diverse report types by their Report-ID (RID). This is useful in case of a device sending corrupted TM or when priority of certain reports change during flight.

For the FLP it needs to be noted that when an Event is disabled by its RID, it is disabled for all on-board objects, which are able to generate it.

Table 2.10 PUS Service 5—Event reporting service

(5,1)	Normal/Progress Report	TM
(5,2)	Error/Anomaly Report - Low Severity	TM
(5,3)	Error/Anomaly Report - Medium Severity	TM
(5,4)	Error/Anomaly Report - High Severity	TM
(5,5)	Enable Event Report Generation	TC
(5,6)	Disable Event Report Generation	TC
(5,128)	Inject Event for Test Purposes	TC

All Service 5 subservices are used on the FLP (see Table 2.10).

The private subservice Service (5,128) can be used to trigger reporting of a certain event. This is used exclusively for test purposes.

2.4.6 PUS-Service 6—Memory Management Service

This service is used to directly access different parts of the on-board memory. It is commonly used in case of OBSW updates/patches or STR star-map patches or in contingency situations, in which certain parts of the memory of a PUS commandable device/software needs to be dumped or overwritten directly.

However, for the FLP this service was extended to enable higher level data access. For example, it is possible to set configuration variables of controllers—like the *ACS_Controller* with its control laws for steering the satellite or the power supply subsystem's *PSS_Controller* with its variables for battery State-of-Charge computation. For more background on the controllers see Sect. 3.2.8.

This access to variables is done by using the controller object-ID as a memory ID and the ID of the parameter as the memory address. This allows performing high level data access operations within the standard PUS framework.

In general, there are two different ways to access the memory, either by an absolute memory address or by using a base address plus an offset. All subservices are defined for both ways. For FLP, only the absolute addressing is used.

The handling of memory management is somewhat similar to the access of device specific information via Service 2. Service 6 however is not used for accessing the OBSW device handlers and the devices, but the assemblies and controllers. The similarity is depicted below (see also Table 2.11):

- TC Service to set a value of a device: Srv. (2,130)
- TC Service to query a value of a device: Srv. (2,130)
- TM Service with the requested values of a device: Srv. (2,135)
- TC Service to set a value of a controller: Srv. (6,132)
- TC Service to query a value of a controller: Srv. (6,135)
- TM Service with the requested values of a controller: Srv. (6,136)

Parameter setting in Service 6 is for one parameter at a time. Setting of multiple parameters requires multiple individual Service 6 TCs.

Table 2.11 PUS-Service 6—memory management service

(6,2)	Load Memory using Absolute Addresses	TC
(6,5)	Dump Memory using Absolute Addresses	TC
(6,6)	Memory Dump using Absolute Addresses Report	TM
(6,9)	Check Memory using Absolute Addresses	TC
(6,10)	Memory Check using Absolute Addresses Report	TM
(6,132)	Load Memory using Absolute Addresses customized	TC
(6,135)	Dump Memory using Absolute Addresses customized	TC
(6,136)	Memory Dump using Absolute Addresses Report customized	TM

2.4.7 PUS-Service 8—Function Management Service

This service has only one subservice defined in the PUS called "Perform Function" (see Table 2.12). It is used to invoke all sorts of non-standard functions, e.g. payload commanding, mode switches etc.

Table 2.12 PUS-Service 8—function management service

(8,1)	Perform Function	TC
(8,129)	Request Data Function Report	TM

Service (8,1) is used to set groups of often used parameters in controllers, e.g. all parameters for the battery State-of-Charge estimation with a single command, and to query the data. The answers to request commands are downlinked using Service (8,129).

As the FLP uses the mode switching concept based on the mode tables and sequence tables described in Sect. 2.1.12 it does not need a full implementation of the classic PUS Service 8.

2.4.8 PUS-Service 9—Time Management Service

This service can be used to report the on-board time to the ground to coordinate it with UTC (9,2) and to change the rate with which this report is generated (9,1).

On the FLP a clock supplied by the OBC operating system counts the seconds from a defined starting point (OBC boot). This service is used to link this seconds counter with UTC by uplinking the current UTC as Service (9,128) TC—see Table 2.13. Once the GPS system is on, an on-board time in UTC format is

Table 2.13 PUS Service 9—time management service

(9,1)	Change Time Report Generation Rate	TC
(9,128)	Set Time Request	TC

available from the GPS time packets on board and GPS time is used. This time code is more precise than any OBC OS seconds counter based on-board time.

When the GPS system is switched off (safe mode, detumble mode), the correlation between the on-board time counter and UTC can still be done using any live TM packet containing a time stamp. Thus the standard subservices (9,1) and (9,2) are not essential and are not implemented in the FLP Generation 1 OBSW. In the PUS it is explicitly stated that the concepts for missions using a GPS system (like FLP spacecraft) are not covered.

2.4.9 PUS-Service 11—On-board Operations Scheduling Service

Within this PUS Service a command schedule is maintained, which consists of loadable telecommands to be executed at a specified time. According to the ECSS standard [14] such a schedule can be divided into subschedules, which each can contain a dedicated commands or Flight Procedures for example. For the first FLP mission at the University of Stuttgart, however subschedules are not supported by the older version of the Mission Control System (MCS).

Via PUS Service 11 it is also possible to downlink the contents of the schedule to double check its integrity. Within the standard, the possibilities for summarized and detailed reports of the contents of the schedule are provided.

The service—as implemented for the FLP—consists of subservices to enable (11,1) and to disable (11,2) the release of commands from the schedule, to reset the entire schedule (11,3) and to insert (11,4) commands into the schedule. These are all part of the minimum capability set and are available on the FLP—see Table 2.14.

It is only possible to enable the schedule if the system time has been set. When inserting new telecommands into the mission timeline, the following rules apply:

- New commands may be inserted both when the schedule is enabled and disabled.
- Telecommands do not need to be ordered according time tag during uplink/load. The scheduling service orders them by time-tag.
- If a system time is set, the service rejects TCs which are already outdated at time of insertion. Otherwise, insertion is possible.

Table 2.14 PUS-Service 11—On-board operations scheduling service

(11,1)	Enable Release of Telecommands	TC
(11,2)	Disable Release of Telecommands	TC
(11,3)	Reset Command Schedule	TC
(11,4)	Insert Telecommands in Command Schedule	TC

In the FLP Generation 1 OBSW the schedule is stored in the OBC RAM, therefore it is lost in case of a system reboot. Resetting the schedule (11,3) works as specified in the ECSS standard. The service disables the schedule in case of a time jump, as defined in the PUS standard.

More sophisticated subservices for reporting, time shifting of commands etc. are not yet implemented—also due to the fact that the payload management in FLP Generation 1 via the PLOC is somewhat limited compared to the architecture later targeted for FLP Generation 2—see Sect. 9.3.

2.4.10 PUS-Service 12—On-board Monitoring Service

This Service is used to monitor on-board variables in order to generate Event TM or to notify the *FDIR* in the OBSW if specified parameter limits are exceeded. The generated Monitoring Reports contain details on all limit transitions since the last time the same Report was issued, including how long and how often limits were exceeded. The service also maintains a list of parameters to be monitored and the applicable limits.

PUS Service 12 implementation is also minimized and tailored for its implementation on FLP Generation 1. As the parameter monitoring is performed already in the device handlers and controllers of the object-oriented OBSW (see Sect. 3.2), it is sufficient to provide the modification service and transition report (12,12)—see Table 2.15. The only other standard service is the definition of the maximum reporting delay via Service (12,3).

Besides the PUS Service to modify the parameter checking information (=set limits) (12,7), there is a custom subservice (12,137) to be used for specific objects with additional limits, e.g. the battery strings which have a maximum spatial gradient limit.

Monitoring of parameters in a specific object, e.g. a controller, can also be enabled and disabled by Services (12,131) and (12,132) respectively.

Table 2.15 PUS-Service 12—On-board monitoring service

(12,3)	Change maximum reporting delay	TC
(12,7)	Modify Parameter Checking Information	TC
(12,12)	Check Transition Report	TM
(12,131)	Enable Monitoring of Parameters within one object	TC
(12,132)	Disable Monitoring of Parameters within one object	TC
(12,137)	Modify Parameter Checking Information for objects with additional limits	TC

2.4.11 PUS-Service 15—On-board Storage and Retrieval Service

This service maintains a list which TM packet is to be stored in which memory area (packet store). There exist three packet stores in the OBC I/O-Board, one for Service 3 HK and diagnostic packets, one for Service 5 event TM and one for all other TM (see Sects. 2.4.11 and 4.3.2). The service is used to organize storage of the packets during times of no ground station access and to retrieve the packets in order to downlink them—see also Table 2.16:

Table 2.16 PUS_Service 15—On-board storage and retrieval service

(15,1)	Enable Storage in Packet Stores	TC
(15,2)	Disable Storage in Packet Stores	TC
(15,3)	Add Packets to Storage Selection Definition	TC
(15,4)	Remove Packets from Storage Selection Definition	TC
(15,5)	Report Storage Selection Definition	TC
(15,6)	Storage Selection Definition Report	TM
(15,7)	Downlink Packet Store Contents for Packet Range	TC
(15,10)	Delete Packet Stores Contents up to Specified Packets	TC
(15,12)	Report Catalogues for Selected Packet Stores	TC
(15,13)	Packet Store Catalogue Report	TM

2.4.12 PUS-Service 17—Test Service

This service is a simple "ping" service, designed to test the connection between the ground station and the spacecraft during tests, during LEOP and potentially at the beginning of each ground station contact. It consists of a request to perform a connection test (17,1) and a connection test report (17,2) as a reply to the request (see Table 2.17). If both packets are received correctly by each side, the connection is established successfully. Both subservices are implemented on in the FLP Core DHS.

Table 2.17 PUS-Service 17—test service

(17,1)	Perform Connection Test ("Ping")	TC
(17,2)	Connection Test Report	TM

2.4.13 PUS-Service 18—On-board Control Procedures Service

This service can be used to define On-board Control Procedures by means of a predefined scripting language interpreter in the OBSW. Such a feature is not directly provided by the FLP OBSW.

A similar functionality as what OBCPs provide w.r.t. sequence control is provided by the custom service 202 Sequences Control Service (Sect. 2.4.17).

2.4.14 PUS-Service 19—Event-Action Service

Due to the object-oriented OBSW design and the encapsulation of functions the reaction chains on certain symptom occurrences are not implemented via Event-Action chains but by handling through the next higher OBSW instance.

On symptom occurrence Events are generated as in a classic procedural software design for information of the flight operators and for function triggering in the next higher object instance.

Therefore the classic Event-Action Service is not needed for the FLP OBSW.

2.4.15 PUS Service 200—Mode Control Service

The PUS service 200 is custom defined for the FLP mode management and thus replaces certain functionalities of a classical PUS Service 8. As shown in Table 2.18, Service (200,1) is used to command any "object" of the satellite system, which has different modes (see Sect. 3.1.2), in one of its modes—identified by a combination of mode and submode.

An object can be any software or hardware component which is identified by an object-ID (see Sect. 3.2) and has different modes, like

- device handlers,
- assemblies,
- controllers,
- subsystems or
- the satellite system object.

Subservices (200,2) and (200,3) are used to query the mode of an object or an object and all of its children recursively, e.g. all ACS objects for the ACS

Table 2.18 PUS-Service 200—mode control service

(200,1)	Set Object Mode	TC
(200,2)	Request Object Mode	TC
(200,3)	Request Object Mode Recursively	TC

subsystem. There are no report answers defined for these subservices, because the replies are provided in the form of Event TM via Service 5 (see Sect. 2.4.5).

2.4.16 PUS-Service 201—Health Flag Control Service

The PUS service 201 is tailored for the FLP to control the health states of the devices on-board the satellite, which have an associated health flag (for more details see Sect. 3.1.2). As shown in Table 2.19, there is a subservice to set an object's health flag (201,1). The subservice (201,4) is used to query the health flag of an object.

The TM of this request is returned in form of Service 5 Event TM (see Sect. 2.4.5), similar to services (200,2) and (200,3). There exist also subservices to request and report the health flags of all objects having a health flag assigned.

Table 2.19 PUS-Service 201—health flag control service

(201,1)	Set Object Health Flag	TC
(201,2)	Request All Object Health Flags	TC
(201,3)	Report All Object Health Flags	TM
(201,4)	Request Single Object Health Flag	TC

2.4.17 PUS-Service 202—Mode Tables and Sequences Control Service

The PUS Service 202 is tailored for managing the FLP spacecraft high level object modes. These comprise the

- satellite *System* object modes (which correspond 1:1 to the spacecraft modes) and
- the subsystem modes of all the spacecraft subsystems. The latter are also represented by a control object in the hierarchy of the OBSW (see later in Sect. 3.1.1).

This custom PUS Service is used to control the tables and sequences that are used to change the S/C system or a subsystem level mode and which were introduced already in Sect. 2.1.12.

The definitions of subsystem/assembly/device modes versus S/C modes—see Table 2.2—are defined in internal "Mode Tables" which are preloaded in the OBSW ROM image and are reloadable/modifiable via this service from ground.

In general, these tables contain several entries of mode commands for invoking the system transition from one mode to the next—similar to Service (200,1) requests. Functional sequences (sequence tables) on board can contain a number of such mode commands in a table to be executed sequentially and to achieve a mode change on subsystem or even system level.

Table 2.20 PUS-Service 202—mode tables and sequences service

(202,1)	Add Mode Change Table	TC
(202,2)	Add Mode Change Sequence	TC
(202,3)	Delete Mode Change Table	TC
(202,4)	Delete Mode Change Sequence	TC
(202,5)	Request Mode Change Tables Available	TC
(202,6)	Report Mode Change Tables Available	TM
(202,7)	Request Mode Change Sequences Available	TC
(202,8)	Report Mode Change Sequences Available	TM
(202,9)	Request Mode Change Table Data	TC
(202,10)	Report Mode Change Table Data	TM
(202,11)	Request Mode Change Sequence Data	TC
(202,12)	Report Mode Change Sequence Data	TM
(202,13)	Request Available Memory Slots for Mode Change Tables and Sequences	TC
(202,14)	Report Available Memory Slots for Mode Change Tables and Sequences	TM
(202,15)	Request Available Memory Slots for Mode Change Tables of an Object	TC
(202,16)	Report Available Memory Slots for Mode Change Tables of an Object	TM
(202,17)	Request Available Memory Slots for Mode Change Sequences of an Object	TC
(202,18)	Report Available Memory Slots for Mode Change Sequences of an Object	TM

As listed in Table 2.20, there are subservices implemented to add such tables[2] and sequences[3] (202,1), (202,2), to delete them (202,3), (202,4), to request and report lists of available tables and sequences (202,5) to (202,8). Furthermore, there exist subservices

- to request and report the data contained within these tables (202,9) to (202,12) and
- to request and report the available memory slots for tables and sequences (202,13) to (202,14)
- and to request and report the available memory slots for tables or sequences of a single object via the subservices (202,15) to (202,18).

2.4.18 PUS-Service 203—Payload Control Service

The FLP OBSW by default provides a dedicated PUS Service for interfacing a payload controller if implemented in the spacecraft. In the case of the first FLP

[2]target tables in the context described in Sect. 2.1.12.

[3]sequence tables in the context described in Sect. 2.1.12.

based satellite based—the "Flying Laptop"—this custom Service is used for controlling the Payload Control Computer (PLOC).

Details on the specific implementation of payload commands please are to be defined in a mission specific Payload Flight Operations Manual.

2.5 Spacecraft Commandability and Observability

As mentioned the communication between spacecraft and ground station applies the CCSDS standards. Telecommand packets are segmented and framed and converted to Command Link Transmission Units (CLTU) in the ground segment. The CLTUs are modulated, transferred to the spacecraft and demodulated on board by the receivers. The demodulated bitstream is forwarded to the CCSDS-Boards. The CCSDS-Boards extract the TC frames from the CLTUs by performing all the necessary decoding and unpacking. The on-board software cyclically acquires the TC frames from the locked CCSDS-Board (see Chap. 6) and unpacks the TC packets to process them.

As explained in more detail in Sect. 6.1 the FLP is equipped with two CCSDS-Boards, a nominal and a redundant one. The CCSDS-Boards are operated in hot redundancy. When ground contact is achieved the first board reporting receiver lock and carrier lock status to the OBSW will be used to downlink TM unless being marked non-healthy by *FDIR* or by ground. The board used to process the uplinked commands is selected by ground via the TC Virtual Channel (see Sect. 2.2.1).

As the transmitter itself is operated in cold redundancy, there will never be two different signals sent at the same time. During nominal operations, the nominal CCSDS-Board is used. When no TM is received on ground, it is possible that the CCSDS-Board in use is marked as non-healthy or is not working correctly. A switchover has to be commanded.

Spacecraft telemetry also follows the CCSDS standards and transmission is again performed via the CCSDS-Boards to the active transmitter (see Fig. 6.1). TM frames including the TM packets are submitted to the CCSDS-Boards by the OBSW via separate TM Virtual Channels for life TM, and different kinds of playback TM (see Sect. 2.2.1).

The active CCSDS-Board performs the diverse encoding down to Channel Acquisition Data Units (CADU) which are forwarded to the active transmitter and which are modulated there. When ground connection is established and currently no TM is downlinked by the OBSW the CCSDS-Board generates Idle Frames to keep the data connection to Ground alive.

The diverse PUS services for TC and TM that are supported by the FLP have been explained in the previous chapter.

High Priority Commands of class 1 (HPC1)—which bypass the OBC and OBSW—can be sent to the spacecraft and are identified in the CCSDS-Board—by Virtual Channel ID, not by MAP-ID (see Sect. 2.2.1). They are sent to the PCDU

directly from the CCSDS-Boards and directly processed by the Combined-Controller in the PCDU—see Sect. 4.2 and Figs. 4.1 and 4.2.

The FLP Generation 1 does not provide High Priority TM replies for these HPC1 commands.

The mode control service Srv.(200) and the mode table management via Srv. (202) allow a flexible adaptation of the spacecraft transition sequences by upload of new mode tables also in flight. This obsoletes a complex OBCP interpreter engine in the OBSW and supports the concept of an easy to use low-cost platform.

With this set of features the commandability and observability of the spacecraft platform in nominal and contingency situations is sufficiently complete.

2.6 Spacecraft On-board Time Management

On-board time management is performed via PUS Service 9 (Sect. 2.4.8). In the FLP platform, the On-Board Computer (OBC), the GPS system and the Star Tracker (STR) share time information.

The FLP platform does not utilize a dedicated real-time clock which keeps time and date information available at any time, but relies on the time information delivered by ground or by the GPS system. There exist two independent pulse-per-second (PPS) lines, one from the GPS to the OBC and one from the OBC to the STR (see Fig. 2.8). In fact electrically they all are redundant but only one of each pair of redundant lines is active at a time.

A *Time_Manager* object in the OBSW is responsible for collection and distribution of timestamps. A time tag is allocated to the incoming pulse from the GPS, which, together with the absolute time information in the GPS time packet, can be used to set the on-board time in the OBC to an absolute value.

Similarly, the Time_Manager forwards the absolute time of the outgoing pulse to the STR unit, which automatically uses this information to compute and report its attitude information with the OBC on-board time.

The Time Manager in the OBC handles two different time domains:

- After reboot, it uses the CPU uptime as time reference. To maintain a monotonic clock, the current uptime value is stored in the OBC PROM (see Sect. 4.3.1) in regular intervals. It is retrieved at boot time.
  By this means a system restart with a runtime smaller than the previous one is avoided.

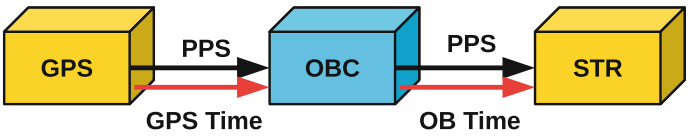


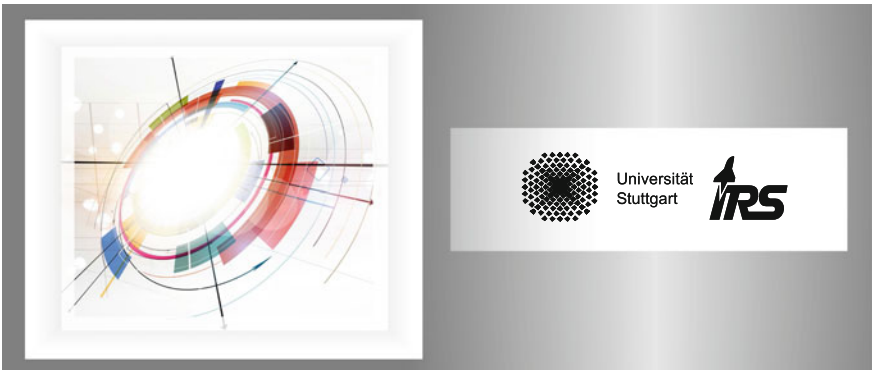
Fig. 2.8 Time distribution in the spacecraft

- An absolute time in (CCSDS CDS format [26]) is used as soon as the time is set by ground command or has been acquired from GPS.

A precise synchronization (in the range of a few ms) to the GPS pulse-per-second is possible with this set-up, but is subject to later implementation in the FLP Generation 2. The current implementation implies jumps in the on-board time when switching from uplinked UTC time to GPS time.

Chapter 3
Data Handling and Control Concept

Ulrich Mohr, Bastian Bätz and Jens Eickhoff



© Fotolia

Abstract This chapter addresses all the fact on the onboard software which an operator needs to know when controlling an FLP-based spacecraft from ground. The implemented onboard software objects tree is explained, from low level device handlers to the top level system object. Furthermore the command/reply flow through the onboard software and the onboard software object types are treated. The object modes and the specifics of controller objects are covered and many further details on object IDs, their addressing in commands and also on the onboard software dynamic architecture like tasks, cycle times and the polling sequence table and crash handling.

Keywords Onboard software objects tree · Command/reply flow through the onboard software · Object types, modes, IDs · Software event management Software dynamic architecture · Tasks and cycle times · Polling sequence table OBSW death report

U. Mohr (✉) · B. Bätz
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: mohr@irs.uni-stuttgart.de

B. Bätz
e-mail: baetz@irs.uni-stuttgart.de

J. Eickhoff
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_3

3.1 Onboard Software Architecture

For spacecraft software the key issue is to meet real-time requirements. This means to fulfill strict timing conditions and to react to occurring events—nominal ones like a telecommand, or non-nominal ones like a failure—instantaneously. Developing a complete real-time operating system was beyond the scope and capacity of the FLP project. Thus, an already existing product, the open-source software RTEMS[1] was selected as a basis for the platform OBSW. Figure 3.1 taken from [140] depicts the basic elements of a standard satellite on-board software. A detailed explanation of the figure is not provided here. The reader is referred to the cited reference. RTEMS provides all operating system (OS) kernel functions plus time managing functions, task scheduling and inter-process communication [81]. These elements are included in the "RTOS" block in Fig. 3.1 and are accessible via an RTEMS C-language programming Application Programming Interface (API). The FLP OBSW pursues this design with some deviations which are due to its implementation in object-oriented design, coded in the programming language C++.

For device communication, I/O *Line Drivers* are implemented, which in this case are represented by an RMAP-stack managing the communication over the SpaceWire (SpW) data links. Equipment handlers contain information about how the devices communicate regarding their protocols and CRC mechanisms. They extract measurement data from incoming device data packets and forward them into a software *datapool*. Further information about the equipment handlers (called *deviceHandler* in the FLP software) is provided in Sect. 3.2.2.

The mission specific applications, including control algorithms for power management, thermal control and attitude/orbit control, use the Onboard Software (OBSW) *datapool* parameter values to evaluate the current state of the spacecraft and command actuators to reach or keep a target state (see Fig. 3.1). However in the FLP OBSW design these control applications are encapsulated in software objects which will be explained later—see also Fig. 3.3 for a quick overview.

On all levels there are failure detection functions that report to a system-wide Failure Detection, Isolation and Recovery (FDIR) object the failure Events that are detected. If possible, the *system FDIR* will take measures to compensate for the failure and allow continuing the current operation (*system FDIR* hereafter only called *FDIR* for short). The FDIR functionality for all subsystems is covered in detail in Chap. 10.

For standardized access from a ground station to the spacecraft, a technique called Packet Utilization Standard (PUS) has been developed by ESA. This standard recommends certain service functions that should be available on board a spacecraft, and how the OBSW is to be configured and commanded and controlled

[1]Real-Time Executive for Multiprocessor Systems—developed by NASA in 1988. The Real-Time Executive for Multiprocessor Systems or RTEMS is an open source fully featured Real Time Operating System (RTOS) that supports a variety of open standard Application Programming Interfaces (API) and interface standards—see [81].

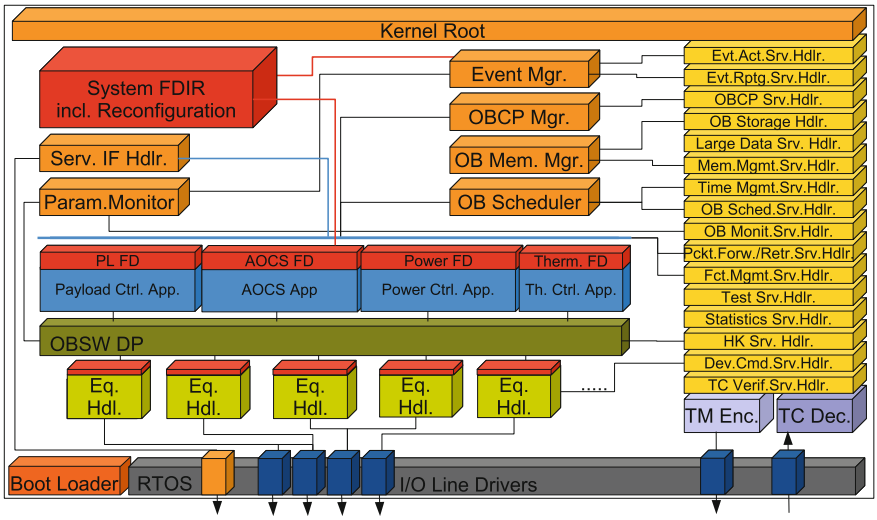


Fig. 3.1 OBSW block diagram. © Jens Eickhoff—from [140]

via these PUS Services from ground in a standardized manner. This allows for compatibility with different, commercial ground stations that operate conform to this PUS standard. For a university satellite, this offers an advantageous usage of commercial ground stations as support.

This chapter briefly describes the principles of the FLP OBSW as it was designed by the FLP OBSW team and as far as they are relevant for spacecraft operations. It introduces the software objects that are important for operations, device communications and failure management. It also covers basic information handling and data flow through the software as far as required and used by the mechanisms specified in the FDIR chapters after this. The dynamic architecture aspects like tasking and timings are also covered. More details about the OBSW framework and the application for FLP will be provided in [134].

The failure management concept for FLP makes use of these basic principles. They are described in Sect. 10.1 and following. However, the details of the fault management functions for the different domains have been designed independent from the software design (see [132]) and served as requirements for the OBSW team.

3.1.1 OBSW Hierarchy Concept

The FLP software is entirely designed following the object-oriented paradigm, not only with respect to coding, but also with respect to modeling system aspects. This is realized by encapsulation of system functionality in independent software objects. Each of these objects covers certain system functionality such as handling a device or performing control operations. There are aggregation objects that group

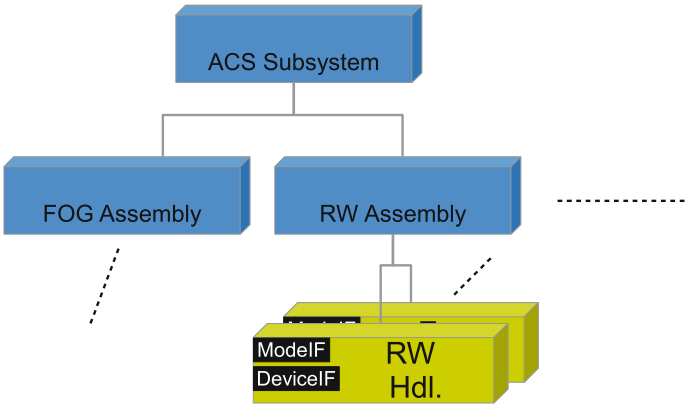


Fig. 3.2 OBSW handler concept. © IRS, University of Stuttgart

objects into subsystems or assemblies. The general attempt is to store certain knowledge only in one dedicated object and to provide interfaces for other objects to access that information.

For example the *ACS_*Subsystem of the satellite is modeled as one so-called "handler" and is equipped with sub-handlers for the reaction wheel *RW_*, the *FOG-Assembly* and others. The *RW_Assembly* holds the sub-handlers for the individual reaction wheels. This is illustrated in Fig. 3.2. The lowest level of handlers are the device handlers (DH). They are equivalent to the *Equipment Handlers* in Fig. 3.1.

On *RW_Assembly* level the information is available about the health status of the individual wheels and default selection. And the *RW_Assembly* then will command the individual *RW_Handler* to start up the wheels (including control parameters such as speed). The process of wheel startup with first requesting power supply from PCDU, RW startup, monitoring of tacho sensor signal and currents etc. up to final success report of startup success up to the *RW_Assembly* is entirely encapsulated on this *RW_Handler* level.

So far the lower level in the object hierarchy. On the other end, the top level is represented in Fig. 3.3 with the *System* object (Object-ID 0x1000100) on the highest level of the hierarchy. The *System* object modes correspond 1:1 to the spacecraft operational modes as depicted in Fig. 2.1. Commanding a spacecraft mode change in software is represented by commanding a mode change of the *System* object.

Below the *System* the next layer in the tree are the subsystem objects, one for each of the main spacecraft subsystems—Core DHS, Power, TTC, ACS, Thermal and Payloads.[2] Below these subsystems the assemblies and devices are located which in Fig. 3.3 only can be imagined. In the subsequent chapters on each subsystem, they will be revisited with individual figures in higher detail and resolution.

[2]The latter being specific for each mission and not treated here in this platform Flight Operations Manual.

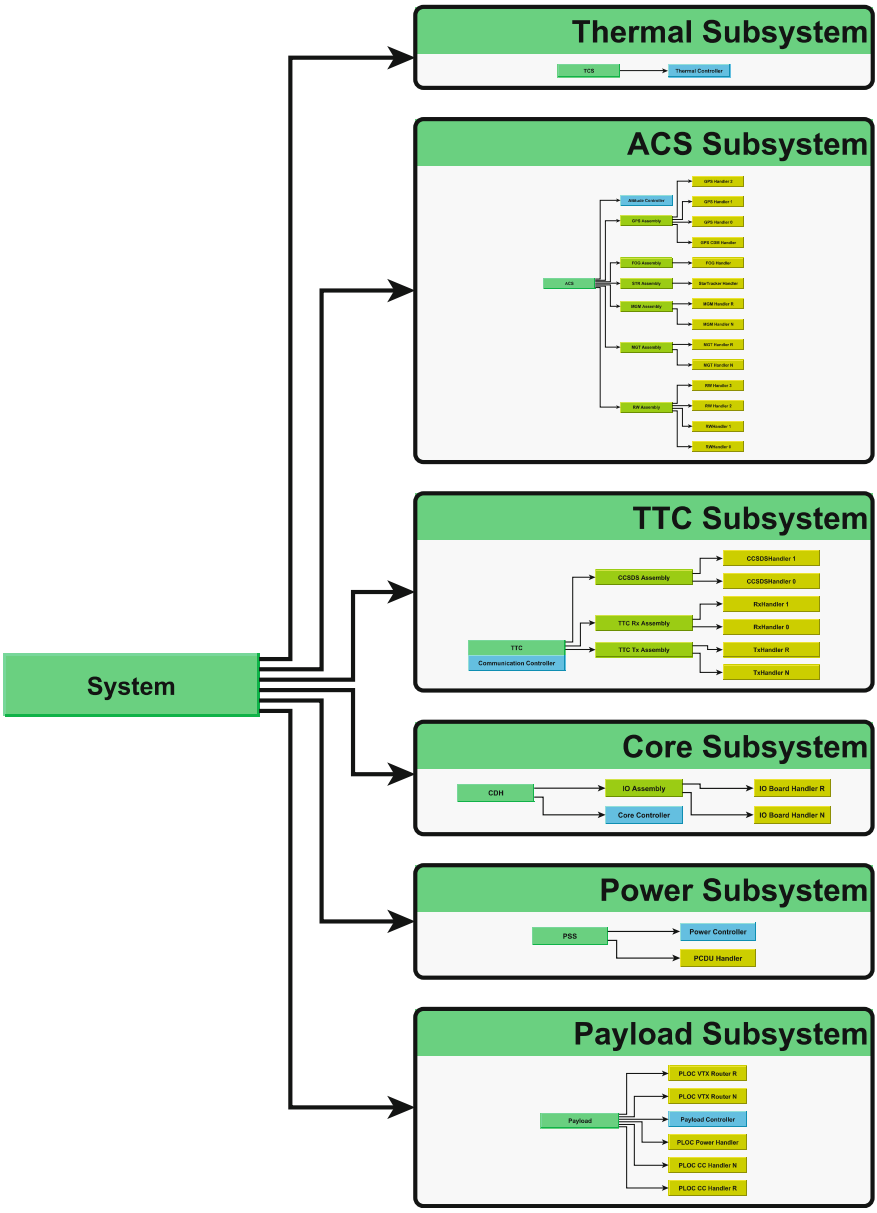


Fig. 3.3 The implemented OBSW subsystem objects tree. © IRS, University of Stuttgart

Each of these handlers on whatever level can be commanded into dedicated modes. A handler being commanded a mode change cares for all its subobjects being mode-changed accordingly.

For the low level objects like device handlers (see more in Sect. 3.2.2) and assemblies (more in Sect. 3.2.3) this is rather straight forward. For example a reaction wheel assembly transiting from 4 wheel mode to 3 wheel mode as result of a defective wheel just shuts down the affected wheel via its device handler. The standard modes of the low level handler objects are Off, On and Normal. Exceptions and extensions will be covered in the individual subsystem Chaps. 4–8.

For higher level objects like *System* a mode change implies an entire change process through the entire hierarchy chain. Therefore on system and subsystem level the mode changes are controllable by loaded mode switch tables—as it was explained in Sect. 2.1.12 and is visualized in Fig. 2.5 and Table 2.1.

An additional aspect are the subsystem controllers. Each subsystem has a dedicated controller (marked blue in Fig. 3.3). The controller contain the algorithms or control laws for the individual modes. E.g. the control laws for all ACS modes (Idle, Nadir-pointing, Target-pointing) are implemented inside the *ACS_Controller*. If ACS is switched from for example Idle Mode to Nadir-pointing Mode the *ACS_Controller* is commanded by the *ACS_Subsystem* object into the new mode and applies the corresponding numerics.

These controller objects (see Sect. 3.2.4) themselves are low level handler objects as the device handlers and assemblies. They have the standard modes of the low level handlers (Off, On and Normal) or, like in case of the *ACS_Controller* have modes matching 1:1 to their subsystem object's modes (*ACS_Subsystem* modes in this example). So for the example of the Idle Mode (see Fig. 2.1), there exists a

- *System* Idle Mode of the spacecraft
- *ACS_Subsystem* Idle Mode
- *ACS_Controller* Idle Mode

This section is only intended to explain the basic principle. Special cases like direct device commanding, failure management and sequence chains of inter-depending actions will be treated in later sections.

3.1.2 Command/Reply Flow Through the OBSW

The ESA Packet Utilization Standard (PUS) covers two aspects—the protocol aspect and the service functionality aspect. The protocol aspect describes how packets are to be structured and interpreted, while the functionality aspect defines which actions are triggered by individual commands. The distinction between those aspects allows to distribute their implementation to different parts of the software.

This service concept is useful as the FLP software framework anyway organizes all functionality into encapsulated software objects. Software objects can be the equipment handlers, assemblies or controllers as cited above, but also the PUS services themselves are modeled as objects. See therefore the service handlers depicted in Fig. 3.1 on the right—and similarly in Fig. 3.4.

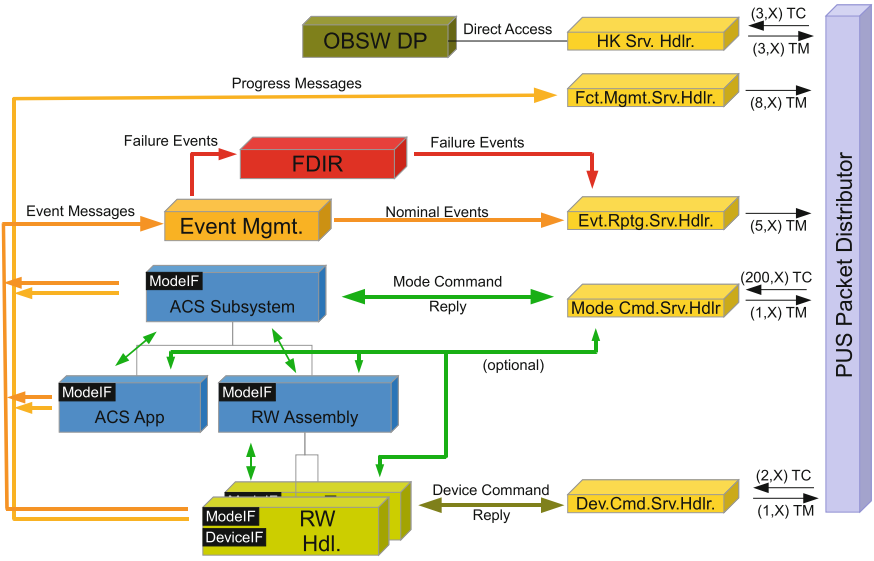


Fig. 3.4 Command/reply flow through the OBSW. © IRS, University of Stuttgart

The PUS services are implemented in the FLP on-board software such that each service is assigned with an interface which defines the functions needed for this specific service. An object can now provide one or more interfaces and thus can offer the functions required by the services (see Fig. 3.4).

The protocol aspect for a service is implemented in a dedicated PUS service object class. Incoming PUS telecommands are routed to the derived PUS service objects implementing the dedicated service handler. Note that messages sent internally between a PUS service handler object and an addressed target object—like e.g. the depicted *ACS_Subsystem* in Fig. 3.4—are no longer representing PUS, but an internal format. One advantage of translating the commands to an internal format is that by this means objects can command other objects internally as well, without any telecommand. In this case the target does not see any difference and no separate functions need to be implemented.

In some cases the object executing the commanded request does not need to be a separate object. One example would be the housekeeping service. Here, the functionality is directly implemented in the service handler object as no other objects need to be involved and, hence, no internal messages are being sent.

Another important aspect is that a dedicated service can be implemented based on multiple objects. For example a mode command can be accepted by a “Device Handler”-object, which takes care of the equipment mode transition from “On” to “Off” and vice versa, which may include power-up, configuration and simple functionality tests. The accepted mode command is generic, so a configurable subsystem object (Fig. 3.3), which has a number of “mode objects” attached can change their modes with the help of predefined mode tables (as it was explained in

Sect. 2.1.12 and visualized in Fig. 2.5 and Table 2.1). Execution of a sequence of mode tables can also be initiated by a mode command, so that subsystems can be commanded by other subsystems, allowing hierarchical grouping of the objects (see Fig. 3.4 and Table 2.1). As a result, most typical standard Flight Procedures (Chap. 15)—like for transiting from the satellite Idle Mode to a Target-pointing Mode—only require a single mode command to the top-level system object.

The OBSW is designed in UML, coded in C++. Some of its code objects are "visible" to the operator—meaning commandable from ground—while others are performing their functionality out of the operator's accessibility. The latter class of objects serve to build the functional framework of the code, supply inherited features to subclasses or control the software dynamics. Such inaccessible elements of the code will not be treated further in the frame of this Flight Operations Manual but in [134]. Each object has a unique identifier, the "*objectID*". The visible objects are listed in Sect. 3.2.8.

Objects receive commands which trigger various actions. Commands are grouped together into interfaces. If an object implements an interface, it is able to receive all of the commands that are defined in this interface. Objects can implement an arbitrary number of interfaces.

Some of the most important interfaces are:

- *HasModesIF* This interfaces declares that an object has a mode and a submode. It receives commands to change and read the current mode and submode. The mode is used to change the behavior of a periodically working object, while the submode optionally specifies parameters for the mode.
- *HasHealthIF* This interface declares that an object has a health state. It receives commands to set and read the health state.
- *DeviceHandlerIF* This interface declares an object that handles communications with a device connected to the I/O-Board of the OBC. It receives various commands related to the monitoring, interception and insertion of commands from and to the device.
- *HasMemoryIF* This interface declares an object that manages some sort of memory (e.g. TM packet stores). It receives commands to read, write and verify this memory.
- *AcceptsTelecommandsIF* This interface declares an object that is able to parse PUS Telecommands. It receives a commands to parse it.

3.2 OBSW Object Types

The objects of the OBSW can be grouped into types according to common interfaces they implement and the function that they perform. While all objects of a certain type implement the common interfaces, it is possible that individual objects implement additional extended interfaces (Fig. 3.5).

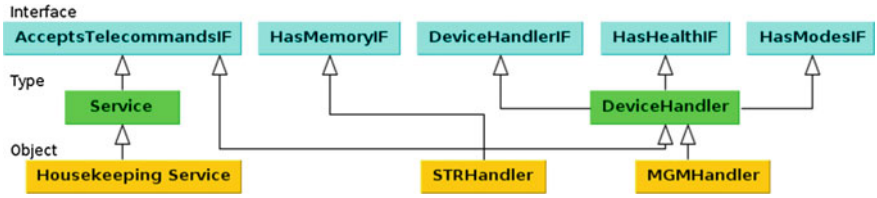


Fig. 3.5 Interfaces, types and objects. © IRS, University of Stuttgart

One example is the device handler of the star tracker. It is of the type device handler. As the selected star tracker type is able to directly command via PUS telecommands and telemetry [61], its device handler additionally implements the *HasMemoryIF* and *AcceptsTelecommandsIF* to allow changing of configuration variables and sending PUS commands directly to the device.

3.2.1 Service Objects

Service objects provide the interface to the ground. As example refer to the PUS Service 3 Housekeeping Service Handler "HK Srv. Hdlr."—colored in orange—in the upper right of Fig. 3.4. Inside the OBSW this object is called PUS_HK_SERVICE. These service objects accept telecommands and generate telemetry replies which are forwarded to ground. Each PUS Service type implementation on board is represented by one service object.

These service objects extract the data from the PUS packets, check the recipient, identified by its object-ID and forward the request to the addressed target object. The target executes the request and reports all significant execution steps, which includes failure or completion, to the service object. Depending on the service either the same or another service object transforms the reports into PUS verification or housekeeping messages. Service objects implement the *AcceptsTelecommandsIF* to be able to receive telecommands.

3.2.2 Device Handler Objects

Device handler (DH) objects handle the communication with devices attached to the OBC. During nominal operations, the handlers are commanded from ground instead of the devices themselves so that the actual protocol used by the devices is hidden from the operator through a common commanding approach. Device handlers also collect telemetry from the devices on a regular basis and write it to the common *datapool*.

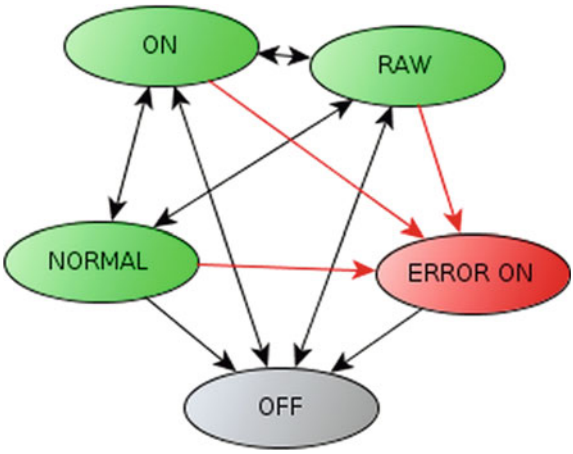


Fig. 3.6 Device handler modes. © IRS, University of Stuttgart

Device handlers implement the *HasModesIF* and by standard offer the five modes, described in Fig. 3.6. Submodes are specific to individual devices. Device handler also implement the *DeviceHandlerIF* with a set of available commands to send a specific command to the device or to read specific telemetry from the device which is then stored in the common *datapool*. There exists also a set of commands to monitor the connection between the device handler and its device.

For the PUS services used for commanding dedicated device mode transitions please refer to the detailed explanations in Sect. 2.4.

On Mode

In this mode a device (e.g. a reaction wheel) is switched on, fully configured and ready to perform its tasks. The device is commanded to be passive, which means that actuators are commanded to zero torque and sensors are not polled. Also, no routine communication is performed.

- This mode can be commanded from all modes from except Error On.
- If this mode is commanded from Off mode, the handler will command the PCDU to switch on the switches necessary to power the device and to perform the boot-sequence necessary to make the device ready for full operation.
- If this mode is commanded from any other mode, the device will be set to be passive.
- In On mode, service (2,130) direct TCs (see Sect. 2.4) are accepted and executed.
- If a device handler has reached the On mode successfully, there has always been a basic successful communication between the handler and the device. Thus commanding a device handler from Off to On mode and checking for success, can be used as a simple functional test of the device.

Off Mode
In this mode, the device is not powered. The Off mode can be commanded from all other modes.

- If a device handler is set to off mode, it shuts down the device nominally in case where the device is still On, and it commands the PCDU to deactivate the device's power supply.
- Only service (2,131) raw device TCs (see Sect. 2.4) are accepted in this mode.

Normal Mode
In Normal mode, the device handler continuously submits commands to the device. ACS actuators are commanded to the torque requested by the controller and both sensors and actuators are polled on a regular basis and their telemetry is written to the OBSW *datapool*.

- This mode can only be commanded from On mode and Raw mode.
- In Normal mode, service (2,130) TC (see Sect. 2.4) are accepted and executed.

Error On Mode
The Error-On mode is entered if the device handler is unable to switch off the device during the transition to Off mode.

- The only way to leave this mode is to switch off the device by telecommand and to send send an Off mode command.
- If a switch can not be switched off anymore but the device handler still must be set to Off mode, the switch state in the *datapool* can be overwritten temporarily as this is the value checked by the device handler to get the switch state. This can be done by Service (2,131) raw TCs.

Raw Mode
The Raw mode is used for direct device commanding from ground. The device handler only serves as a router. Raw device commands of Service (2,128) are sent to the device and the command replies are sent back to ground directly as Service (2,129) telemetry (see Sect. 2.4). This functionality can be used for all devices in contingency situations or for test purposes.

- This mode can be commanded from On, Off and Normal mode.
- If Raw mode is commanded from On or Normal mode, the handler does not perform any operation but setting actuators to zero torque so that the state of the device is left unchanged.
- If Raw mode is commanded from Off mode, the device handler does not perform any operation without dedicated telecommand (also no power-up the device). This is intended for contingency cases requiring manual boot-up of a device.
- The handler does not accept any (2,130) TCs in Raw mode.

Device Handler Internal Data
A *deviceHandler* stores knowledge about the physical device it represents. This includes:

- communication protocol used by the device—start bit, stop bit, message length, CRC
- calibration functions for measurement values of the device
- the specific number of its power switch and fuse in the PCDU
- frequency of HK data polling
- boot up sequence of the device
- boot up duration of the device
- device modes
- device health status.

By storing the ID of the device's power switch in the PCDU, the process of how to switch the device on/off is stored in this device handler. The mechanism of powering off a device by triggering the switch in the PCDU is such that a command is sent to the corresponding device handler object. The device handler performs the device preparation for shutdown and handshakes the electric power-line opening with the PCDU.

This also works in failure cases. For example *FDIR* can send a *healthCommand()* with the target health state to the device handler. Figure 3.7 provides an illustration for this mechanism. The currently implemented health states (*healthFlag*) are:

- HEALTHY—nominal operation is possible if activated.
- PASSIVE—the device has a passive state and is currently in this state.
- DEACTIVATED—the device is healthy, but is switched off.
- EXTERNAL_CONTROL—the device is under control of the ground operator. It ignores commands from the S/C system. The ground operator is responsible for the device.
- FAULTY—there have been malfunctions (might be one or more) of a device within a certain period of time for *FDIR* to decide to initiate a reboot of the device.
- PERMANENTLY_FAULTY—there have been malfunctions (might be one or more) of the device within a certain period of time for *FDIR* to decide that the device must not be used any further. Recovery is only possible through ground interaction.

For every device, a *healthFlag* is stored in the OBSW *datapool* (see Fig. 3.1)—called *datapool* in the SW—and in the Spacecraft State Vector on the I/O-Boards—see later in Sects. 4.3.2 and 10.2.11. This is required so that after a software reboot the system can be made aware of which device has been marked as PERMANENTLY_FAULTY before, to avoid unsing this device again. Devices that have been marked as PERMANENTLY_FAULTY by *FDIR* or ground can only be reset to HEALTHY through ground interaction.

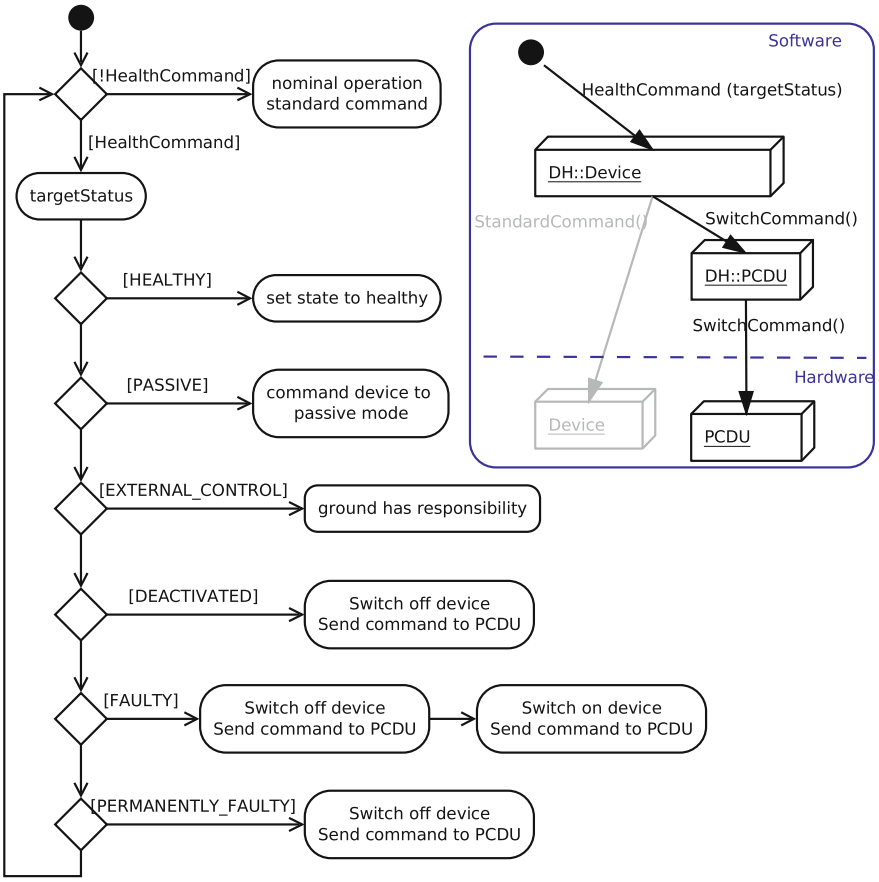


Fig. 3.7 Device health command—controlling of devices. © IRS, University of Stuttgart

Every device handler has an interface to the PCDU device handler. This interface allows for a device handler to initiate a *switchCommand()* to the PCDU that contains the device's switch ID. This technique is also depicted in Fig. 3.7. During nominal operation a standard command is sent to the device for HK polling. Only when the health state changes, a *switchCommand()* is sent to the PCDU. The command is sent to the PCDU device handler to forward the command to the PCDU and initiate the change of the switch's status On → Off or vice versa.

As consequence of a mode transition, several devices might be switched On or Off at the same time. The according device handlers will all send a separate command to the PCDU device handler. However, the PCDU can only process one command per cycle. Therefore, it will take several cycles until all desired devices are switched On/Off.

It is intended that a device handler does not store information about the power consumption of its device. All information about power consumption throughout the system is stored in the power controller object.

3.2.3 Assembly Objects

Assembly objects implement the *HasModesIF*. The mode of an assembly reflects the mode of the underlying device handlers. Thus, assemblies offer all of the modes of device handlers except for the Error-On mode.

Assemblies store information about the redundancy approach of their devices and handle /hide redundancy switching operations, which is encapsulated in the assembly. In case where one device fails and is switched off by *FDIR* or ground, the corresponding *assembly* checks whether its current mode can be kept with the remaining devices, and if so the current operation is continued. Otherwise, it signals that it cannot keep the mode which will lead to the system switching off the *assembly* and the devices. This might lead to a system mode switch as well. The mode of an *assembly* reflects the mode of the underlying device handlers. Thus, assemblies offer all the modes of device handlers.

Figure 3.8 is an example for the behavior and function of two assemblies illustrating their redundancy handling. The RW system is based on a 3-out-of-4 redundancy concept. Four wheels are physically assembled in a tetrahedron configuration. This allows for the compensation of one RW in case of a failure.

The assembly is in nominal operational mode when all four RWs are available. The availability of the RWs is verified in every ACS cycle. If one RW is marked as non-healthy and is deactivated, the assembly will verify whether there are still three RWs available to continue the current operation. The S/C *System* object itself is not allowed to reactivate a device marked as non-healthy. But during the operation with three RWs, ground interaction could lead to resetting the health state of the faulty one to HEALTHY.

If another RW fails, the assembly will check again whether at least three RWs are available. If this is not the case, the assembly will report that the current mode cannot be continued and will automatically deactivates its devices, namely all wheels.

In Sect. 3.2.2 and Fig. 3.7 the status EXTERNAL_CONTROL has been introduced. This means that the device is ignored by all automatic processes in the system. This state can be applied by ground for device tests and could lead to resetting its *healthFlag* to HEALTHY.

Depending on the mode, only one working device is required for the *GPS_Assembly* in the example. So the assembly steps through the devices until finding a healthy and activated one. If no healthy device is found, it signals that it cannot keep the current mode and the assembly will be switched off.

Note: *The event CANT_KEEP_MODE can be generated by different objects. Separation between the events, or determination of a reaction, can be achieved by examining the contained event parameters, for example the ID of the reporting object.*

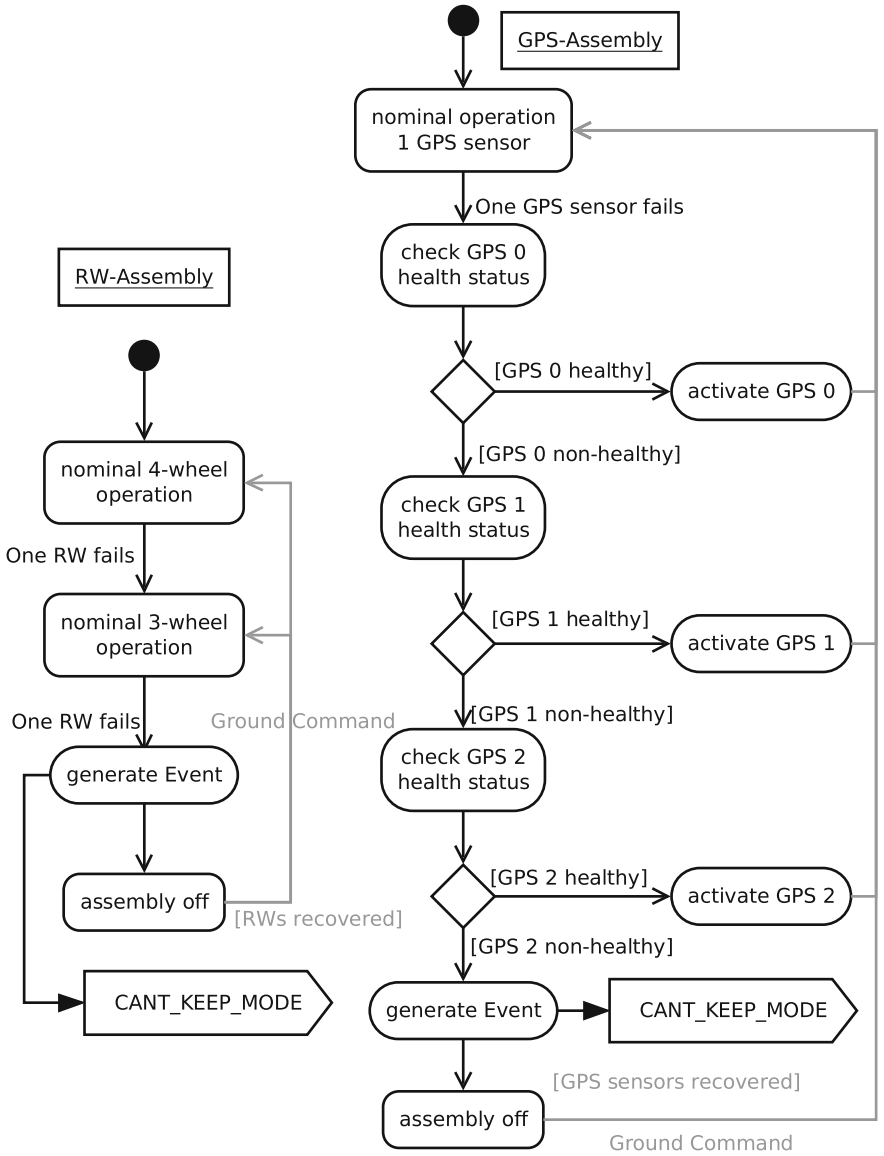


Fig. 3.8 Example for assembly behavior—GPS and RW system. © IRS, University of Stuttgart

3.2.4 Controller Objects

These objects are on a similar level as assemblies and monitor and control the behavior of subsystems. The monitoring and processing of TM parameters is

performed within the controllers and control algorithms like the attitude control or the battery State-of-Charge estimation are executed here.

Controller objects implement the *HasModesIF* and offer three modes:

- Off: The controller does not perform any action
- On: The controller performs its nominal operation but does not give any commanding output
- Normal: The controller performs its nominal operation

There might be additional modes for some of the controller objects. Controller objects implement the *HasMemoryIF* to allow access to configuration values.

Controller (CTR) objects contain knowledge about the (mathematical) mechanisms that are required to keep the satellite in operation. Basically, they make control decisions and initiate commands based on sensor measurement inputs.

From software perspective, the controller objects are on a similar level as *assemblies* and monitor/control the behavior of subsystems. In Fig. 3.9 an abstract example for a controller object is shown with the three steps that are executed by these objects.

As a first step, input from sensors is monitored and checked for validity. This action will be detailed in the FDIR mechanisms of the different subsystems in Sects. 10.5–10.8. Some sensor input will need calibration before further usage, e.g. temperatures are provided in raw value, calibrated Centigrade units, or even as electrical resistance. Equal measurements from redundant sensors are correlated (sensor fusion) so that all further steps have one common set of input values to work with.

In a second step, the controller verifies whether it has valid input for all calculations that its current mode comprises. The execution of the control algorithms will be performed and torque, speed or On/Off settings will be obtained and stored in the controller. Depending on the required visibility of these data in the software, they will be stored in the central *datapool,* or internally in the controller. They are accessible from ground by requesting the according TM packet. If the availability of valid sensor information is not sufficient to execute the functions, the controller will trigger a CANT_KEEP_MODE event.

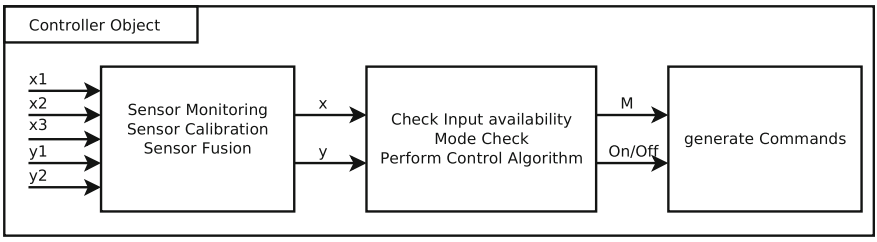


Fig. 3.9 Example content of a controller object. © IRS, University of Stuttgart

The final step of a controller uses the results of the control algorithm and generates the required commands for the actuator devices. The commands are stored in the *datapool* where actuator device handlers have access to them.

3.2.5 Subsystem Objects

Subsystem objects (see also Fig. 3.3) are abstract objects used to aggregate other objects into a hierarchical tree. Subsystem objects can group controllers, assemblies, device handlers (where no assembly exists) and other subsystems as their children.

- On this level, mode management for the spacecraft subsystem modes is performed.
- *Subsystem* objects are one level above the subsystem controller or assembly objects. E.g. the *TCS_Subsystem* object is in the hierarchy above the *TCS_Controller*.

Subsystem objects will be commanded by one central system object.

3.2.6 Top-Level System Object

In the OBSW there exists one top-level system object with the ID = 0x1000100 which represents the mode interface of the software (see also Fig. 3.3). It is called S*ystem* object. It is the root element of the hierarchical mode tree and handles mode transitions of the complete satellite. It corresponds to the "Kernel Root" layer in Fig. 3.1.

Commands or mode transitions are hereafter often given in diagrams as *System::function()*. *System::Boot()* commands *System* to execute the *Boot()*-function.

Boot() in this case does not mean that the software is booted. This particular function attempts to convert the satellite system into a known and reasonable starting state, that the software is aware and in control of. The system object will be commanded from ground or by *FDIR* if a mode transition is to be performed.

3.2.7 Object Execution

In a procedural real-time software, different functions of a software are usually encapsulated in separate processes (threads or tasks—depending on the technical realization), which are executed quasi-parallel (or truly parallel in multiprocessor systems).

For the object-oriented FLP OBSW, this concept is used as well, but with the variation that the objects themselves provide a standardized function, called *performOperation(),* which is executed periodically. Underlying, standard RTEMS tasks are used, as further described later in Sect. 3.4.

3.2.8 OBSW Object-IDs and PRIDs

OBSW *objectID* variables are used to identify each object (devices, OBSW elements or controllers) of the satellite system. object-IDs are four byte values, typically given in a hexadecimal form. The first byte represents the type/level of the object (device handler, subsystem,…). The second and third byte provide information to identify the specific device. The last byte is used as a reference counter for child objects of a given parent object, e.g. for temperature sensors in the TCS. The object-IDs are listed in Table 3.1.

Note: *The possible range of objectID is limited by the identification mechanism for TM packets in the mission control system. The maximum possible ID is* $2^{31} - 1 = 0x7FFFFFFF$. *The maximum value used in the current state is* 0x55FFFFFF.

To facilitate the reading of object-IDs for an operator, the numbers are selected according to the name of the objects and the ASCII number of the object's first letter if possible. Temperature sensors and heaters are considered as child instances of the *TCS_Controller*. They will not be represented by their own objects since they do not offer any complicated functionality. The *TCS Controller* will report in their stead. They need particular object-IDs, however, because they are assigned with a health state. The same is true for fuses and switches being considered as children of the Power Supply System (PSS).

For the IO- and CCSDS-Boards there exist two 'Handler' objects and two 'RMAP' objects. The reason for this is that the 'Handler' instance is derived from the common *deviceHandler* base class, while the SpW communication between OBC Processor-Board and peripheral OBC boards requires additional functions. The base class stores the PCDU switch number and mode information as well as the definitions of basic failure events. For the communication over SpW and RMAP protocol, a separate object is defined, containing the necessary information and report functions about the hardware links, the latter being encapsulated in a further object. PUS-Services have their own ID range as well as system objects.

In commercial spacecraft missions, oftentimes process identification numbers (PRIDs) are used to address distinct processes. For FLP, PRIDs will not be used. Processes or services can be addressed by their respective object-ID.

Table 3.1 OBSW object-ID ranges

Object	Instance	First Byte			Last Byte
Assembly		0x41 (=65d=ASCII 'A')			
	RW	0x41	0x00	0x01	0x00
	MGM	0x41	0x00	0x02	0x00
	MGT	0x41	0x00	0x03	0x00
	FOG	0x41	0x00	0x04	0x00
	GPS	0x41	0x00	0x05	0x00
	TTC_RX	0x41	0x00	0x06	0x00
	TTC_TX	0x41	0x00	0x07	0x00
	STR	0x41	0x00	0x08	0x00
	IO	0x41	0x01	0x00	0x00
	PLOC	0x41	0x01	0x01	0x00
	DDS	0x41	0x01	0x02	0x00
	CCSDS	0x41	0x01	0x03	0x00
Controller		0x43 (=67d='C')			
	ACS	0x43	0x41('A')	0x00	0x00
	Core	0x43	0x43 ('C')	0x00	0x00
	FDIR	0x43	0x46 ('F')	0x00	0x00
	PSS	0x43	0x50 ('P')	0x00	0x00
	Fuses	0x43	0x50	0x01	0x00.. 0x1A
	Solar Panels	0x43	0x50	0x02	0x00 … 0x03
	Bus Power Sensor	0x43	0x50	0x03	0x00
	Batt. String and Sensors	0x43	0x50	0x04 … 0x06	0x00 … 0x01
	TTC	0x43	0x52 ('R') (Radio)	0x00	0x00
	TCS	0x43	0x54 ('T')	0x00	0x00
	Temp. Sensors	0x43	0x54	0x00	0x01 … 0x64
	Heater	0x43	0x54	0x00	0x71 … 0x76
Device Handler		0x44 (=68d='D')			
	PST	0x44	0x00	0x00	0x00
	TTC RX N	0x44	0x00	0x06	0x00
	TTC RX R	0x44	0x00	0x07	0x00
	TTC TX N	0x44	0x00	0x08	0x00
	TTC TX R	0x44	0x00	0x09	0x00
	STR	0x44	0x00	0x10	0x00

(continued)

Table 3.1 (continued)

Object	Instance	First Byte			Last Byte
	FOG	0x44	0x00	0x14	0x00
	RW 0 … 3	0x44	0x00	0x15 … 0x18	0x00
	MGM 0 & 1	0x44	0x00	0x1A & 0x1B	0x00
	MGT 0 & 1	0x44	0x00	0x1C & 0x1D	0x00
	GPS 0 … 2	0x44	0x00	0x1F … 0x21	0x00
	PCDU	0x44	0x00	0x32	0x00
	I/O-Board	0x44	0x45	0x00 & 0x01	0x00
	IO RMAP	0x44	0x46	0x00 & 0x01	0x00
	CCSDS- Board	0x44	0x47	0x00 & 0x01	0x00
	CCSDS RMAP	0x44	0x48	0x00 & 0x01	0x00
PL Device Handler		0x44	0x50	0x00 … 0xFF	0x00 … 0xFF
	PLOC	0x44	0x50	0x00 … 0x06	0x00
	DDS	0x44	0x50	0x00 … 0x0A	0x00
	OSIRIS	0x44	0x50	0x0B	0x00 … 0x02
	AIS	0x44	0x50	0x0E	0x00 & 0x01
	PAMCAM	0x44	0x50	0x0F	
	MICS	0x44	0x50	0x10	0x00 … 0x03
Deploy-ments		0x44	0x50	0x11… 0x12	
	DOM	0x44	0x50	0x11	
	Deployment Mechanism	0x44	0x50	0x12	0x00 & 0x01
PUS-Services		0x50.. 0x52	0x00	0x00 … 0xFF	0x00 … 0xFF
System Objects		0x53.. 0x55	0x00 … 0xFF	0x00 … 0xFF	0x00 … 0xFF

3.3 Observability Functions Provided by the OBSW

There are three default ways in which the FLP software provides observability to the ground operator. These are:

- Command verification packets
- Cyclic housekeeping (HK) packets
- Event packets

The first type indicates successful or failed execution of individual telecommands. HK packets are generated to observe periodically changing sensor and control data and event packets report aperiodic activities and state changes.

3.3.1 Command Verification

The PUS standard defines three stages of command verification:

- Acceptance
- Execution started
- Execution step (n-times)
- Completion

In the FLP software, each telecommand is formally checked before being routed to the destination service object:

- During the step of formal checking, the acceptance report is generated.
- All service objects that forward commands to other objects generate the "execution started" report after checking the existence and availability of the object.
- Depending on the interface, an object needs to report a certain number of intermittent steps when commanded. The service objects forward this information as "step" reports and take care of proper step counting.
- Every object reports successful or failed completion to the commanding one, which is also forwarded as "completion" report.
- Thus, ground operators can precisely monitor execution of telecommands by the destination object.
- In case execution failed, a failure code is forwarded with the report which indicates the precise failure root cause.

3.3.2 Cyclic HK

Cyclic data reporting is required only for values that change over time and whose change is measured with an on-board sensor or calculated from sensor data

on-board. Such data is stored in the *datapool*. Therefore, it is sufficient to make the content available to the HK Service.

Every element of the *datapool*—identified by a unique parameter-ID—can be reported in configurable housekeeping packets. For details on the housekeeping service, please refer to Sect. 2.4.3.

3.3.3 Event Messages

In the OBSW, Events—in the sense of PUS Events—are used to inform the system that a failure has been detected or that a decision has been reached. Event Messages have the format:

```
EventMessage(EventId_t event,EventSeverity_t severity, object_id_t
reporter, uint32_t parameter1, uint32_t parameter2 = 0);
```

That means one event message contains a unique event identifier, the severity level of this Event (see below), the object-ID (see Sect. 3.2.8) of the reporting software object, and two parameters to specify more details required for event handling. The reporting software object might be the device handler representing the faulty device itself, but also a software object that reports for a different device (for example if the *TCS_Controller* object reports a limit violation of a temperature sensor). In some of the diagrams later in this book, Events are displayed. The typing convention is CAPITAL_LETTERS with underscore, and they might include additional arguments in square brackets, for example EVENT [additionalInformation]. a*dditionalInformation* in most cases will be the object-ID of the reporting object. Events are either informational events, marked in black in flow diagrams of this document, or are failure events, marked in red.

The Packet Utilization Standard [13] defines four different levels of on-board Events, namely normal progress Events, and low, medium and high severity warnings. Event packets are generated by on-board processes identified by their object-ID. If the Event telemetry is activated in the Event reporting service (see Sect. 2.4.5), the Events are also downlinked as Service 5 telemetry with the sub-service indicating the Event severity. The Event severity levels are used on the FLP platform as follows:

Severity Level 0 (Normal Progress) Events shall indicate an information message initiated through nominal actions happening on board. They are generated to provide operational progress visibility for ground. Possible scenarios are:

- Boot Event from software
- Successful powering of the payload computer

- Successful mode change of the system or subsystems
- Heater activation and deactivation for thermal control

Severity Level 1 (Low Severity) Events shall indicate that somewhere in the system a failure has been detected. The failure will be transmitted to *FDIR* for further investigations or isolation and recovery actions. This type of Event can be reported from any level of OBSW objects (device handlers, assemblies, subsystems, etc.). Possible scenarios are:

- communication failures with equipment
- malfunction of heaters, sensors, etc.

Severity Level 2 (Medium Severity) Events will be generated by FDIR usually after processing of (several) Severity Level 1 Events. Possible scenarios are:

- A device has been switched off due to failures
- A mode cannot be kept
- Devices have been switched off due to low available power

Severity Level 3 (High Severity) Events will be generated by FDIR and indicate Events of the highest possible severity. Possible scenarios are:

- Battery State-of-Charge is low,
- Safe Mode needs to be activated,
- FDIR cannot handle the Event and/or
- Event is mission critical.

There will be Events, that will be processed, but cannot be handled by the on-board FDIR. However, these will be classified again after processing. If they are deemed not to endanger the system, they will not be processed any further on board. They will be visible for the ground operator and according recovery steps are to be conducted by ground. If they are critical, they will be classified according to severity and be reported to the ground separately. This will signal the significance of the Event to the operator.

3.4 FLP Software Dynamic Architecture

This chapter outlines how CPU time is shared among the individual objects. Task management was not reinvented, but has it's foundations in normal procedural task management of the applied RTEMS operating system. On top of this, a framework to periodically execute individual objects is constructed.

3.4.1 Process/Task Management

Different functions of a software are usually encapsulated in separate processes—scheduled in parallel or pseudo-parallel, especially regarding multiprocessor platforms as of today. In RTEMS, these different processes are called tasks [81]. In the FLP OBSW logically separate processes have their own dedicated task. RTEMS offers several mechanisms to manage tasks. For scheduling, RTEMS utilizes the "Rate Monotonic Scheduling" method.

In the FLP flight software, all tasks are executed periodically (see also Table 3.2) and have a priority proportional to their activation frequency, in accordance with the rules for Rate Monotonic Scheduling (RMS). RTEMS also offers task preemption control to interrupt tasks through a task with higher priority. For inter-process communication via *datapool*, RTEMS semaphores are used to lock the *datapool* while a task is accessing it. Details about task management, thread-safety and prioritization have been determined in [134].

Table 3.2 Task timings of the relevant single object tasks

Object NAME	Task Id	Priority [0-99]	Period [ms]	Stack size [kB]
PUS_VERIFICATION_SERVICE	s01	50	400	2
PUS_DEVICE_COMMAND_SERVICE	s02	40	200	2
PUS_HK_SERVICE	s03	40	200	3
PUS_EVENT_REPORT_SERVICE	s05	25	200	3
PUS_FUNCTION_SERVICE	s08	25	800	2
PUS_TIME_SERVICE	s09	50	400	2
PUS_OPERATIONS_SCHEDULING	s11	25	1000	2
PUS_MONITORING_SERVICE	s12	25	1200	2
PUS_STORAGE_AND_RETRIEVAL	SrAR	25	1200	2
PUS_TEST_SERVICE	s17	20	2000	1
PUS_MODE_COMMANDING	s200	25	1200	2
PUS_HEALTH	s201	25	1200	2
PUS_MODE_TABLES	s202	25	1200	2
CORE_CONTROLLER	core	60	1000	1
TCS_CONTROLLER	tcsC	50	1000	1
EVENT_MANAGER	evnt	80	200	1
DATA_POOL_ADMIN	DP	40	1600	1
PSS_CONTROLLER	pssC	40	800	2
ACS_CONTROLLER	acsC	40	200	2
COMM_CONTROLLER	ttcC	40	2000	1
PAYLOAD_CONTROLLER	payC	40	2000	1
DDS_CONTROLLER	ddsC	40	5000	1
FDIR_CONTROLLER	fdC	50	100	1
TC_PACKET_DISTRIBUTOR	dist	50	400	1

3.4.2 *Object Tasks*

All software objects provide a *performOperation()* method, which is executed periodically. Execution is managed with special *ObjectTask* classes that handle the interface with the underlying RTEMS task features.

If the order of object execution is relevant, multiple objects may be executed in one task in a pre-defined order (as depicted in Fig. 3.10). Also, if an objects is part of another object, the owner object may execute its part in its own *performOperation()* call.

- In the FLP OBSW it is not intended to quit a running task, i.e. tasks are started at boot and expected to run "forever".
- Also, neither period nor priority are intended to be changed at run-time in orbit, as their values are carefully determined by analysis and verified by tests against spacecraft simulator, EM equipment and FM hardware.
- Safe concurrent access on global objects (e.g. *datapool*, *healthtable*) is ensured by locking the objects for short periods of time.
- No external command activities are required for safe handling of global objects.

At the current state of the FLP Generation 1 OBSW implementation there is a large number of single object tasks, e.g. one for each single PUS Service, for controllers and for several system tasks. A future refinement may reduce the total number of tasks by consequently grouping related elements. Table 3.2 provides an overview of all object tasks in use.

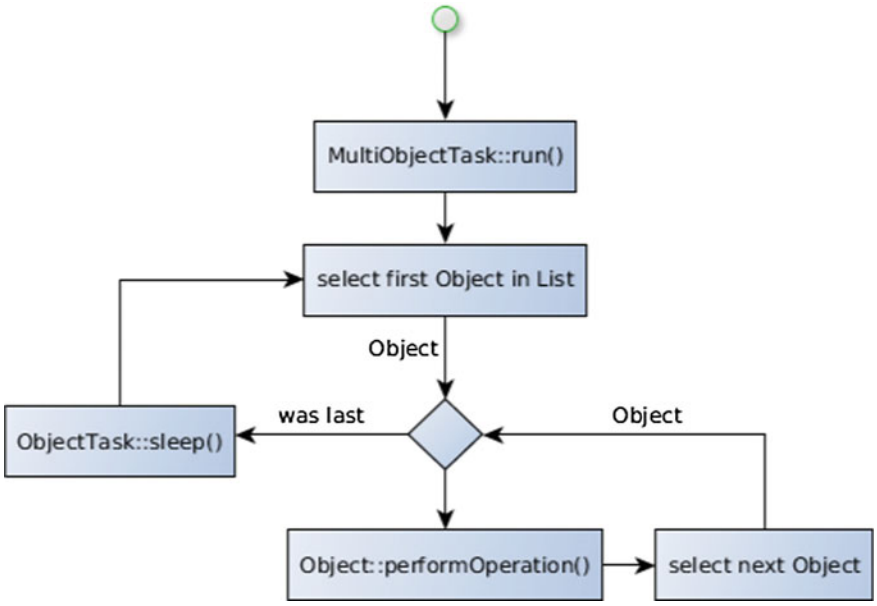


Fig. 3.10 Execution of multiple objects in one task. © IRS, University of Stuttgart

Table 3.3 Task with more than one assigned object

Task Id	Priority [0–99]	Period [ms]	Stack Size [kB]	Objects
MEM	40	200	2	PUS_MEMORY_MANAGEMENT_SERVICE LOCAL_MEMORY_MANAGER
StAF	50	100	3	PUS_PACKET_FORWARDING TM_STORE_EXEC
ccsdA	60	100	10	CCSDS_HANDLER_0 CCSDS_HANDLER_1 CCSDS_ASSEMBLY
ttcA	40	400	1	TTC_TX_ASSEMBLY TTC_RX_ASSEMBLY
assy	40	400	1	FOG_ASSEMBLY GPS_ASSEMBLY MGM_ASSEMBLY MGT_ASSEMBLY RW_ASSEMBLY STR_ASSEMBLY
pAss	40	400	1	PLOC_ASSEMBLY DDS_ASSEMBLY
subs	40	200	1	SYSTEM SUBSYSTEM_ACS SUBSYSTEM_TCS SUBSYSTEM_TTC SUBSYSTEM_PSS SUBSYSTEM_PAYLOAD

Table 3.3 lists all tasks that currently have more than one object assigned. Most of those are set up for performance reasons, i.e. to keep the execution delay between two objects small. Execution order is as listed in the table.

3.4.3 Polling Sequence Table

A special task in S/C software is the *Polling Sequence Table* (PST) which schedules all device communication that goes via the I/O-Board of the OBC. One device after the other it executes the communication functions of the *deviceHandler* objects in a predefined order. For time optimization, device communication is designed such that every device handler is executed twice within a communication cycle. During the first execution a data request is submitted to the device, which in reality is sent to the I/O-Board and then is forwarded electrically from there on to the device. In the meantime, other device handlers and tasks are executed. This allows for the device to have time to execute the command and to return the requested data to the I/O-Board, where it is stored in the corresponding device transmission buffer. Upon

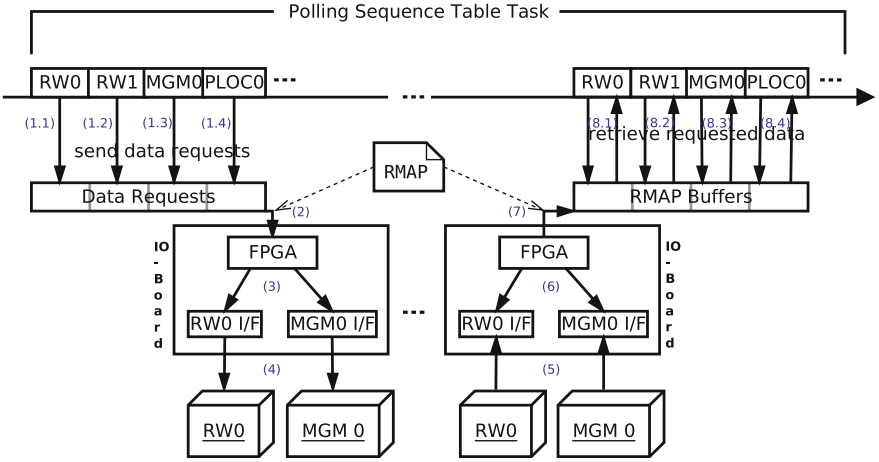


Fig. 3.11 Polling sequence table. © IRS, University of Stuttgart

the second device handler execution the data from the I/O-Board's transmission buffer is polled and for evaluation in the device handler.

Figure 3.11 illustrates the data flow and the timing of the PST in one communication cycle. The process is described in the following steps:

(1) The DHs are executed in the PST and, if in normal mode, generate a data request command that is collected by the RMAP driver. All DHs perform this one after the other. All requests are collected by the RMAP driver.
(2) In a "data burst" the collected data requests are sent via RMAP and SpW interface to the I/O-Board.
(3) On the I/O-Board the requests are forwarded to the corresponding device interface simultaneously, e.g. *RW0 I/F* or *MGM0 I/F*.
(4) The I/F drivers send the requests via their electrical link to the devices. This happens simultaneously, but may take different amounts of time depending on the data rate of the I/F.[3]
(5) After processing the command, the devices send back their response to the OBC.
(6) For every device there is a transmission buffer on the I/O-Board that is used to store the response data from the devices until the OBSW fetches the data.
(7) After a period of 45 ms the OBSW RMAP driver collects the data via SpW and stores them in the RMAP buffer on the Processor-Board. All data

[3]Example—RW interface and MGM interface: Both interfaces are RS-422 standard. The RWs supports a data rate of 9.6 kBit/s whereas the MGMs support 115.2 kBit/s.

interfaces of the devices are fast enough. And the devices are confirmed to be quick enough, to provide a response within the latency period between request send and data retrieval. Otherwise, a failure event will be reported by the corresponding device handler.

(8) After a total of 50 ms the PST executes the DHs again which one after the other retrieves the requested data from the RMAP buffer. Upon reception of the device response, they evaluate the response and perform basic failure detection as described in Sect. 10.4.

3.4.4 Dynamic Scheduling

There is no fixed execution order for tasks. But, due to the nature of Rate Monotonic Scheduling, task execution is automatically ordered by priority, i.e. a high priority task will typically execute before a lower priority task.

Still, passing on requests from the services to the executing objects introduces delays in command execution (see Sect. 3.2.1). Depending on the arrival of a TC, worst case execution delay for a given object is the corresponding task's period, which is typically below **1 s**, but may be up to **5 s** for some slow responding payloads. However, there is no risk of nested delays, as commands are directly forwarded to the executing object.

The flight software supports reporting of task statistics to ground. This is managed by a function of the *Core_Controller* element of the *CoreDHS_Subsystem*.

3.4.5 Modes versus Actions

For the OBSW dynamic architecture two types of dynamic processing must be determined. These are periodic mode execution on the one hand and sporadic action processing on the other (please also refer to Fig. 3.12).

Objects in a given mode are cyclically scheduled and perform their computational task in each cycle. An example are the low-level device handlers which cyclically poll the equipment TM and command TC to the equipment if applicable.

But also the main control applications, such as the thermal control, power control belong to this type. The algorithms computed throughout the loop depend on the mode the object is in, which is controlled by the subsystem mode. E.g the ACS will compute different things in spacecraft Idle Mode compared to Target Pointing Mode. Another example is the thermal control application. In sun-pointing Idle Mode it will control spacecraft heaters differently than in Nadir Pointing Mode.

In contrast to this periodic execution stand the actions. These embed control flows which are triggered on command and then process a sequence of steps. These

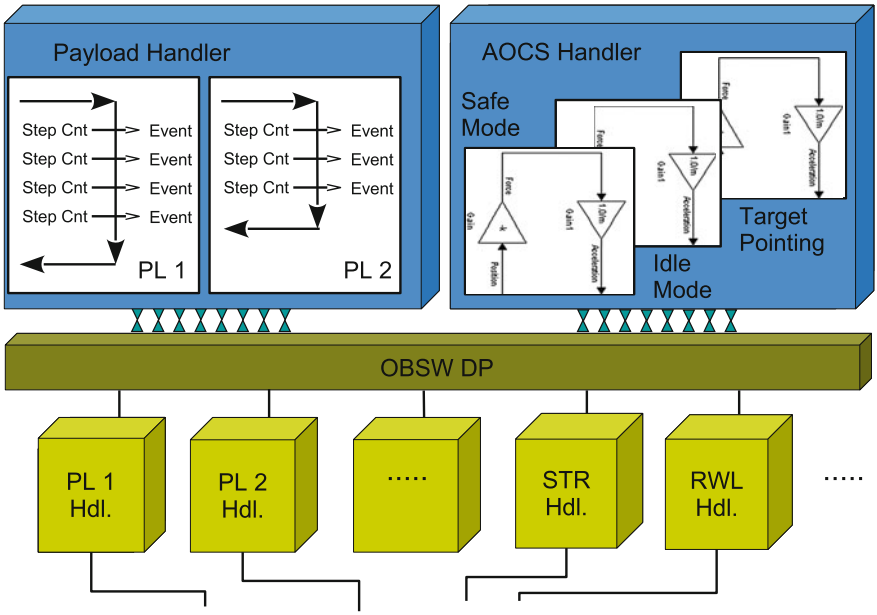


Fig. 3.12 Sequence-control and loop-control objects. © IRS, University of Stuttgart

sequences correspond to "On-board Control Procedures" (OBCP) in a conventional spacecraft. Depending on their individual implementation these sequences can generate start/end Events and can submit step counter TM during their processing.

A typical application for this functionality is the payload control application (see also Fig. 3.1). For each payload such individual control sequences are implemented—for example for acquiring images with a camera payload. In this example a number of steps are necessary until such an image is stored on board.

- Power status must be checked.
- The payload compartment must be temperature stabilized.
- The payload controller must be activated.
- The PCDU must close the LCL to supply the camera with power.
- Payload settings must be adjusted.
- The platform pointing must be initiated and monitored.
- The image acquisition must be started and stopped.
- The raw image data eventually have to be compressed and stored.

Such a control flow can be initiated directly with a single command via ground TC. However such a flow also can be initiated via the sequence start being part of a

mission timeline which includes multiple operations of one or more payloads while the satellite being out of ground visibility.

These actions are not limited to the payload control application. This means they also can be used for other one-shot activities such as establishing ground contact with the payload data downlink system. More details on control sequence features are provided in the individual subsystem control chapters later.

3.5 Onboard Software Death Report

The death report (or in the the FLP terminology "Fatal Error Report") is stored in the OBC PROM to be able to also store it, when no communication with the I/O-Boards is possible any more.

When the OBC boots the next time after a death report was stored, the death report will be reported as a high severity Event (RID 7803) through service (5,4). If this happens during an orbit period without ground station contact, the death report Event packet will be stored in the Event store on the I/O-Board and downloaded during the Event TM dump at the next ground station contact.

Additionally, the last stored death report then is available from the *datapool* through service 3 HK TM. The information in the *datapool* is also available if the software rebooted in between the fatal crash and the current boot.

The parameters in the *datapool* are:

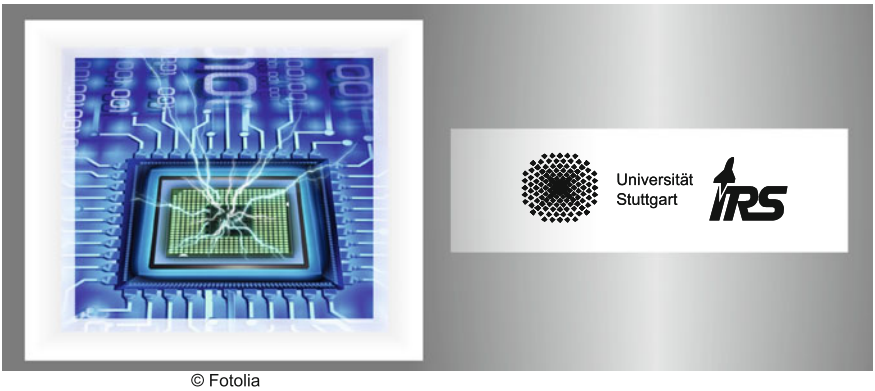
- Error code (the code describing the fatal error which lead to the crash).
- Additional info for some error codes (usually 0).
- A flag indicating whether the report is new. If the flag is "old", the software reported the death report already during an earlier boot and rebooted at least once since the error occurred.
- A flag indicating if the error is internal or external (always internal with the current implementation of FLP Generation 1 OBSW)
- The error source (from the OS or from the OBSW)

As the Fatal Error Report information is then available in the *datapool*, the HK telemetry packet SID 2602 can be activated which transmit the *datapool* internals to ground as normal housekeeping TM.

Note: *This packet normally is deactivated since most of the datapool entries are rather static and the packet downlink consumes considerable bandwidth.*

Chapter 4
Core Data Handling Subsystem

Jens Eickhoff, Rouven Witt and Bastian Bätz



Abstract The first spacecraft subsystem—the Core Data Handling Subsystem (Core DHS)—is described in this chapter. As the FLP platform is based on the very innovative concept of a Combined Data and Power Management Infrastructure (CDPI) instead on the classic 3 unit design of onboard computer, power control and distribution unit and remote interface unit, this CDPI concept and its benefits are explained first. Then the chapter describes the stepwise boot process of an FLP-based spacecraft after its separation from the launcher. The last part comprises the onboard software elements and functions for control of the Core DHS subsystem.

J. Eickhoff (✉)
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

R. Witt · B. Bätz
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: witt@irs.uni-stuttgart.de

B. Bätz
e-mail: baetz@irs.uni-stuttgart.de

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_4

Keywords Combined data and power management infrastructure (CDPI) · CDPI failure detection, isolation and recovery · OBC memories (PROM, RAM, I/O-Board persistent RAM) · The satellite boot process after launcher separation · Core DHS subsystem control

4.1 On-Board Data Handling Subsystem

The FLP platform core data handling subsystem (Core DHS) is based on a specific architecture of OBC and PCDU. As it was mentioned in Sect. 1.8 this infrastructure is called a Combined Data and Power Management Infrastructure (CDPI)—see also Fig. 1.7 and [47].

The overall system control for nominal cases is performed by the OBSW which is described in Chap. 3 and for non nominal cases by the reconfiguration functions and emergency function of the CDPI Combined-Controller for OBC and PCDU subunits—see Sect. 10.2.

The top-level satellite mode control is performed by the *system* object in the OBSW (see Sect. 3.2.6), the control algorithms, control of the equipment assemblies and of the individual devises is performed by corresponding sub-objects all explained in Chap. 3.

The treatment of non-nominal operational cases of the Core DHS—such as OBSW reboots or OBC hardware reconfiguration or OBSW/power related topics—are treated in the Sects. 10.2 and 10.3.

Here the CDPI hardware shall be explained in more detail and subsequently the main OBSW objects which control/monitor the hardware. As it is a Combined Data and Power Management Infrastructure also some topics related to the boot process after launcher separation are treated in this chapter.

4.2 Combined Data and Power Management Infrastructure

A CDPI is a logical merge of a classic Onboard Computer (OBC) and a Power Control and Distribution Unit. The main features are:

- The OBC is designed for single redundancy of all its subunits. Therefore each printed circuit board (PCB) type is available in two instances and all OBC internal data interfaces between its boards are cross-coupled.
- The first OBC board type is the power supply board, which provides the 3.3 V power supply to the individual PCBs and which itself is coupled to the 19–25.5 V satellite power supply from the PCDU. The OBC Power-Board in addition performs the routing of some OBC Processor-Board interfaces to the

outside of the box to the satellite skin connector, namely the SIF, the JTAG-I/F, the line-in of the 1 Hz GPS PPS strobe and the line-out of the star tracker PPS strobe.

- The second board type is the Processor-Board, a so-called Single Board Computer (SBC), since it comprises not only the CPU, but also non-volatile memory for boot loader, operating system and Onboard Software (OBSW) boot image, as well as volatile RAM for running the OBSW. Packet stores for OBSW housekeeping data, for the S/C state vector and the S/C configuration vector are available on the next board type, the so-called I/O-Boards.
- The I/O-Boards are coupled to the Processor-Boards via SpaceWire interfaces and serve for bridging to all types of lower level digital interfaces of S/C onboard equipment. So all onboard units providing interface types like UART, RS422 or others are coupled to the I/O-Boards and the Processor-Board can access their data via its SpaceWire interface (see also Fig. 1.6).
  The I/O-Board design is based on a radiation tolerant FPGA with an according IP Core which is accessed via SpaceWire from the Processor-Board. And secondly it is based on according driver ICs for each type of digital I/O interface to connected spacecraft equipment.
  Since this approach requires some buffers and memory chips on the board anyway, these boards are designed to handle the storage of OBSW housekeeping data, the S/C state vector and the S/C configuration vector via additional memory and according enhanced IP Core functions.
- The fourth type of boards are the CCSDS protocol decoder/encoder boards (or CCSDS-Boars for short) which are also coupled to the Processor-Boards via SpaceWire interfaces and on the other side are interfacing the spacecraft's transceiver units (see also Fig. 1.6). These boards perform the low-level decoding of the uplinked telecommands and the low level encoding of the telemetry to be downlinked during ground station visibility.
  The approach for these boards was to use the identical PCB and FPGA as for the I/O-Boards, and to just equip the CCSDS-Boards with only a limited number of external interfaces, namely those to the transceivers, to the PCDU for High Priority Commands (HPC) and those for cross-coupling of the Command Link Control Word (CLCW) interfaces. Obviously these CCSDS-Boards comprise a different IP Core since they perform a completely different task.
  The CCSDS-Boards subsystem-wise belong to the Telemetry and Telecommand (TTC) subsystem. More details on reading telecommands from the receivers into the OBSW and vice versa writing TM from OBSW to the transmitters therefore can be found in Chap. 6.

The full technical specification of the FLP Generation 1 CDPI is [47]. When looking at the above mentioned design paradigms, a reader who is familiar with classic OBC design immediately will identify some elementary OBC functions to be missing in the so far described board types. These functions have been implemented through a sort of functional merge of OBC and PCDU and they will be explained in the following sections.

4.2.1 The PCDU as Analog RIU

Already at begin of the OBC development a PCDU from Vectronic Aerospace, Berlin, was preselected for satellite power-bus management. This family of PCDUs is equipped with relatively performant microcontrollers and with analog measurement functions for current and voltage measurement and with according functions for TM reporting to the OBC. More details can be found in Sect. 5.5.

Thus part of classic OBC functions such as the tasks of

- analog current measurement,
- voltage measurement and
- parameter analog to digital conversion

for the analog power lines and also for all analog sensor interfaces—as e.g. thermistor and sun-sensor readings—is implemented as PCDU function in this OBC/PCDU co-design. Thus in the CDPI now the PCDU takes over these tasks instead of a Remote Interface Unit (RIU) in classic OBC design (see also Sect. 5.5.4.1).

4.2.2 A Combined-Controller for Power and DHS FDIR

In a classic unit architecture on board of a satellite three controllers implement the Failure Detection, Isolation and Recovery (FDIR) functionality, namely the OBC's Reconfiguration Unit [140] and as subunit the OBC's Command Pulse Decoding Unit (CPDU) [140] as well as the PCDU's internal controller IC. These elements are typically implemented without being based on onboard software but either as pure electronic circuits or as FPGA or ASIC implementation.

This implies that the spacecraft flies three controllers which need to be highly reliable and thus require according design and test effort. The basic idea behind the CDPI patent affiliation [39, 40] is to implement only one FDIR controller for both data handling FDIR and power FDIR, reconfiguration and emergency functions—the patent cites it as a "Combined-Controller". As this controller covers FDIR for both data handling and power is of no relevance, whether it is mounted physically inside the OBC housing or the PCDU housing or even in a common housing of both.

The FLP Generation 1 based "Flying Laptop" satellite from the University of Stuttgart is the first spacecraft to fly such a CDPI architecture with a Combined-Controller. In this first implementation the Combined-Controller physically resides in the PCDU housing and is an implementation of the standard Vectronic Aerospace PCDU controller with enhanced firmware functions defined by the Stuttgart FLP team and implemented by Vectronic Aerospace.

In normal operational cases the OBSW can still command the "PCDU controller"—i.e. the Combined-Controller—to switch on/off spacecraft equipment power lines. The reconfiguration is triggered by the Combined-Controller either via

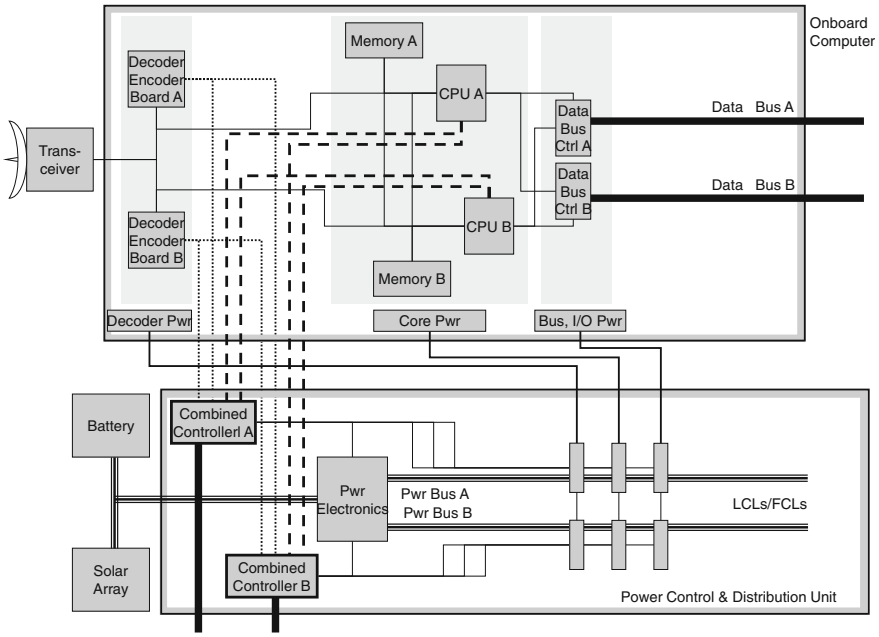


Fig. 4.1 Combined data and power management infrastructure. © Airbus DS—see [39]

soft-reset command lines to the individual OBC subunits (e.g. CPU) or via power resets respectively via power-down of the defective unit and power-up of the component's redundancy.

As example the Combined-Controller (in the PCDU in the current implementation example) can separately power each OBC decoder/encoder board, each Processor-Board with CPU, NV-RAM, RAM, clock module and finally also each OBC I/O-Board (Fig. 4.1).

4.2.3 Completeness of System Architecture

With this system design approach for the Combined Data and Power Management Infrastructure all necessary flight relevant functions are covered (see also [140])—although they are not all allocated to the classic boards or component:

- OBC power supply -> on OBC Power-Boards
- OBC processor and internal data-bus -> on OBC Processor-Boards
- OBC memory—non volatile for boot loader and operating system and OBSW—and volatile as work memory -> on OBC Processor-Boards
- OBC packet stores for Housekeeping Telemetry (HK TM), Spacecraft State Vector -> on I/O-Board

- OBC digital RIU function—coupling of all non SpaceWire digital equipment I/O to the OBC -> on I/O-Board
- Interface to the spacecraft transceivers decoding TCs and encoding TM -> on CCSDS-Boards
- OBC analog RIU functions—coupling of all analog interface control and analog parameter measurements to the OBC -> implemented in PCDU and commanded/controlled from OBC via OBC-PCDU UART interfaces.
- OBC reconfiguration functions -> implemented in the Combined-Controller and in this implementation realized by the PCDU controller with enhanced firmware.
- The OBC High Priority Command (HPC) interface functionality to implement implicitly the functions of an OBC's CPDU -> implemented in the Combined-Controller firmware and accessible through additional UART interfaces via the OBC CCSDS-Boards
- Bus power management functions -> implemented in the PCDU Combined-Controller
- Equipment power supply switching and overvoltage/overcurrent protection -> implemented in the PCDU Combined-Controller
- Power-bus undervoltage FDIR functions (DNEL functions) -> implemented in the PCDU Combined-Controller.

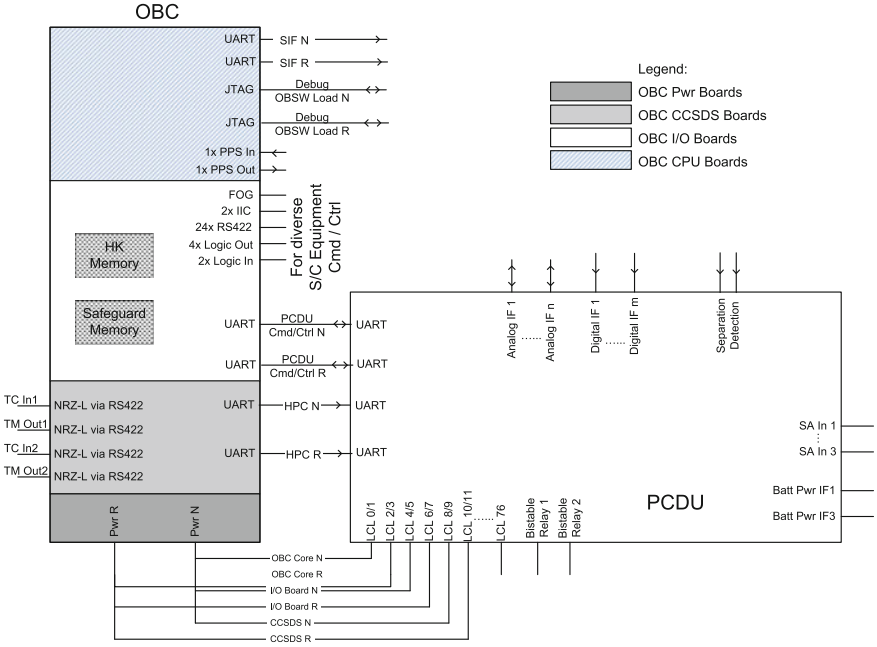


Fig. 4.2 The combined data and power management infrastructure. © Jens Eickhoff—from [47]

An overview of the CDPI with all external I/Os and the interlinks between OBC and PCDU part is included in Fig. 4.2. Box internal cross-couplings—like between Processor-Boards and I/O-Boards—are not included for not overcomplicating the figure here. The same applies for cross-couplings of redundant elements within OBC respectively PCDU.

4.3 Data Management

The FLP software implements different methods to manage different types of data. On the lowest level, data are stored and held in one of the physical memory areas within the FLP On-Board Computer (see Fig. 4.3). Data in the OBC RAM are not manually allocated but are managed by the OBSW application itself. Accessible data are either held in the global OBSW *datapool* (see Fig. 3.1) or in the individual objects. Data areas in the OBC PROM and the I/O-Board persistent RAM are reserved manually.

In the center of on-board communication is the *datapool*, cited in Fig. 3.1 as "OBSW DP". In this *datapool* all variables that need to be accessed by more than one object are stored, such as measurement and control loop data.

datapool is a global object in RAM that stores and delivers the containing elements to other objects. Distribution is performed in a blackboard manner, i.e. new data overwrite the existing ones.

This is in contrast to a controller internal *private pool* which is only accessed by one distinct *controller* object—as Fig. 4.4 illustrates. However, the data of the *private pool* can be accessed by ground to validate the controller's function if required.

All data in *datapool* carry a *validFlag* that indicates the validity of the entry. If a device does not provide data or the CRC of its housekeeping packet is not successful, the device handler still writes the provided data into the *datapool* to allow ground access, but it marks the data as invalid.

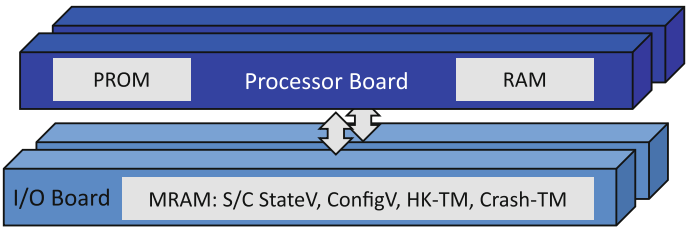


Fig. 4.3 Memory areas in the OBC managed by software. © IRS, University of Stuttgart

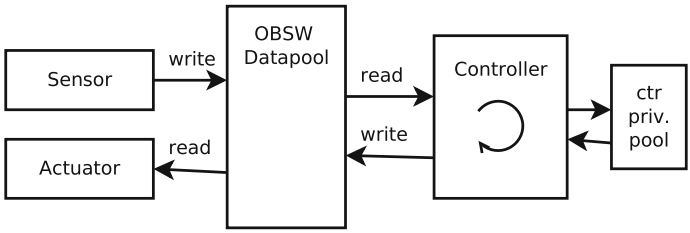


Fig. 4.4 OBSW datapool access. © IRS, University of Stuttgart

4.3.1 PROM Data

The OBC PROM space is allocated manually. It is mainly used to hold the software boot images. In addition, some essential state vector data is stored in the PROM. The PROM in the Processor-Board design is no real PROM in the classic sense but is implemented using NV-RAM chips (see [47]).

- Each OBC Processor-Board has a redundant PROM—called primary and secondary PROM.
- Both PROMs contain a complete OBSW image.
- The boot loader is located at the start address of the primary PROM, which is called each time whenever the Processor-Board is booted.
- At start of the secondary PROM some configuration variables are located which are partly for use by the boot loader and partly for the OBSW itself.
- Both PROMs have a sleep-state, in which the memory chips consume less power and in which the memory is secured against read and write access.
- The sleep-state can be set individually for each PROM (primary and secondary).
- Furthermore the LEON3FT processor provides a control bit for defining the write-access for all PROMs. This is not individually configurable for primary and secondary PROM.
- After the boot process both PROMS are in sleep-state and write-access is disabled.

The boot loader has two functions:

- At each start of a Processor-Board a boot-counter variable is increased to monitor the number of boot processes (e.g. during FDIR operations without ground visibility). The boot-counter is located at the start address of the secondary PROM.
- At each boot of a dedicated Processor-Board, either the OBSW image in the primary or in the secondary PROM is booted. The decision is based on a second counter—another variable at start of the secondary PROM.
  - In case where this second counter is >0 the OBSW image in the secondary PROM is used and the counter is decreased by one.
  - In case where the counter equals zero the OBSW image from the primary PROM is booted.

Further variables in the secondary PROM are the Fatal-Error-Counter and the on-board time which is stored there and which is refreshed in larger cycles to facilitate OBSW and ACS recovery after Core DHS crashes (HW or SW).

In case where the OBSW reaches a point where it cannot continue (Fatal Error) it still tries to dump the last error in the secondary PROM. At next boot after such a Fatal Error the error code is read out and is stored in the *datapool* and is transmitted to ground as failure Event (5,4).

4.3.2 I/O-Board Persistent RAM

Both I/O-Boards provide an extension of non-volatile RAM for the OBC. There are five MRAM banks of 4 MB each available, organized in two regions of 16 and 4 MB. These memory regions are accessible via RMAP calls over SpaceWire. The first region is used to keep telemetry stores for HK, event and miscellaneous telemetry, such as technology demonstration data. The second region, memory range 0x03000000–0x033FFFFF is reserved for use as Spacecraft State Vector storage.

Allocation of RAM to the different telemetry stores is done in the OBSW configuration. Table 4.1 lists the allocation of memory blocks to TM stores. Section 2.4.11 further describes the internal data usage of these data blocks.

Table 4.1 Allocation of memory blocks to TM stores on the I/O-Boards

Tm Store	Address range	Size
Housekeeping Store	0x02000000-0x027FFFFF	8 MB
Event Store	0x02800000-0x02AFFFFF	3 MB
Misc. Store	0x02B00000-0x02FFFFFF	5 MB

4.3.3 OBC RAM and PROM Direct Access

As mentioned above, data in the OBC RAM are allocated by the executing application and the boot loader. Therefore it is not intended (and not recommended) to directly manipulate data in RAM from ground.

In contingency cases however, it may be necessary to check or read elements directly. In addition, loading a new software image to PROM requires direct access to the PROM memory from ground.

This direct memory manipulation of OBC memory is handled with the Memory Management Service and the *LocalMemory* object of the *CoreDHS_Subsystem*, which manages access to all local memory on one Processor-Board.

4.4 System Boot at Launcher Separation

As the FLP platform is suitable for launches in Off Mode, the boot process of the satellite starts with the separation from the launch vehicle (initiated by the launcher itself). The satellite is equipped with a short circuitry in the PCDU which—when opened—causes the PCDU to boot (please also refer to Sect. 11.6.1). The PCDU controlled parts of the LEOP Autosequence are shown in Fig. 4.5.

In the left swim-lane of the diagram, the activities of the PCDU are depicted. These activities include the separation detection and the boot procedure of the PCDU itself (see [57] and Sects. 5.5.3 and 5.6.2.2), during which the battery state of charge is checked to be SoC > 45 % and whether OBC and TTC heater lines are operational. These are activated automatically by thermostat switches in case where the unit temperatures are below the operational minimum of −40 °C.[1] It is not expected that the OBC/TTC become that cold. Thus the PCDU is expected to boot safely into its so-called MiniOps Mode [47, 57] which takes about 20 s. At that point the initially acquired battery SoC value will not yet be reliable until the first battery calibration and will therefore be initialized to an artificially high value (100 % SoC) for safety reasons. From PCDU MiniOps Mode the further operations performed purely by PCDU and the tasks of the OBC start running in parallel—please refer to Fig. 4.5.

After launcher separation detection a preset counter is continuously being decreased inside the PCDU (Deployment Timer 0). The value of the Deployment Timer will be prescribed by the launch supplier to induce some delay between satellite separation and first mechanical activities.

After reaching (Deployment Timer 0) value 0, the PCDU starts to try the solar panels deployment by melting the clampbands of the panel hold down and release mechanisms (see also Sect. 5.3). The circuits for all four deployment mechanisms are closed for **60** s (controlled by another timer—Deployment Timer 1) to deploy the solar panels.

The complete deployment is achieved, when all four Reed sensors detect the opening of the panels (see [57]). In this case, the deployment flag within the PCDU in such case will be set to “1” (Off)—indicating deployment process being completed—and powering of the panel deployment mechanisms is stopped.

In case where the deployment was not successfully completed after a minute (Deployment Timer 1), the timer is reset for another minute until performing a deployment retry—see also Sect. 10.3.3. At maximum five such retry attempts are performed by the PCDU, until the deployment flag is set to “1” (Off) even in case the panels have not deployed correctly. This is a safety feature to avoid that the battery is emptied by the continuously high deployment mechanism currents being consumed in an endless loop.

[1]The latter two constraints are normally of no relevance for the LEOP boot of the satellite but for emergency recovery boots in orbit and thus according checks are included in the PCDU boot sequence.

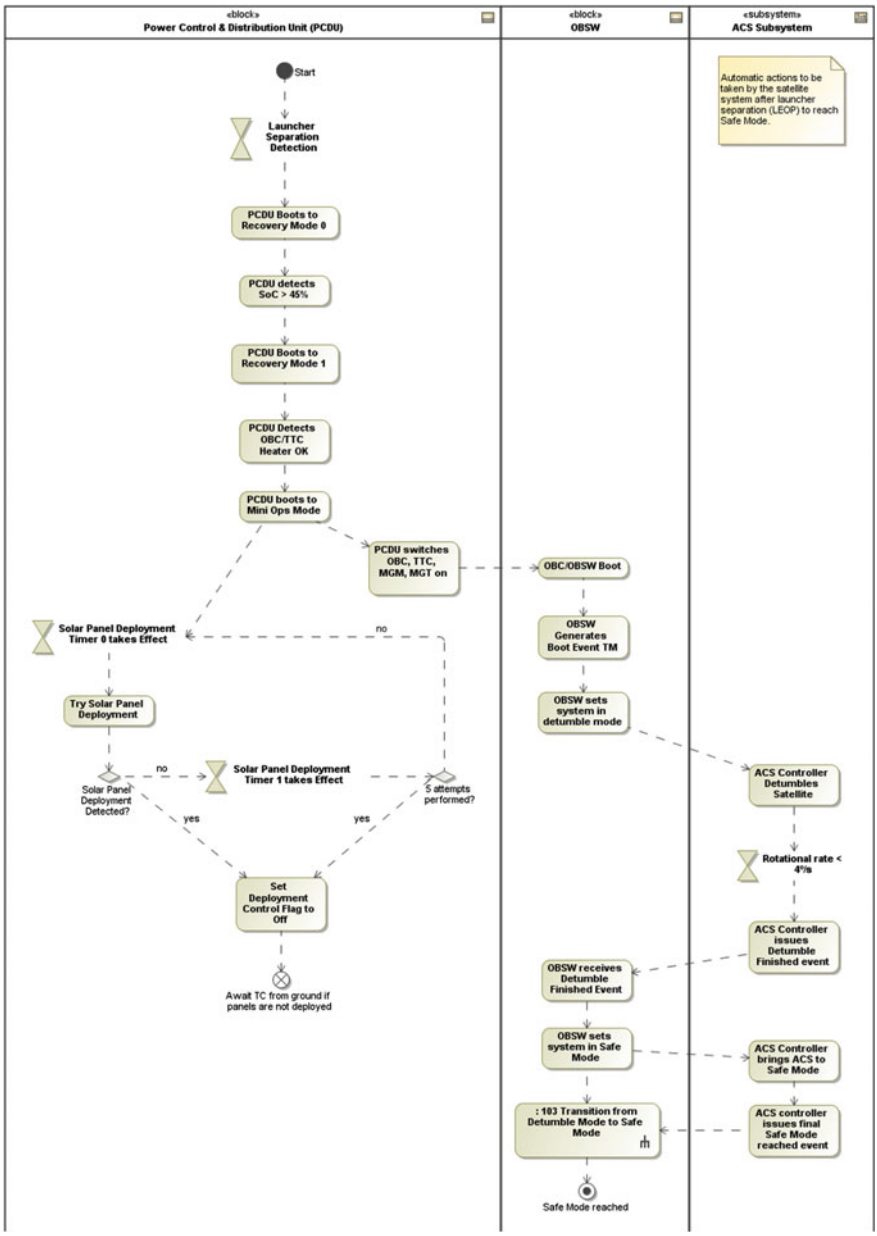


Fig. 4.5 LEOP automatic activities after launcher separation. © IRS, University of Stuttgart

In such case where deployment is stopped and it is left for the operators to handle the problem. The deployment flag can be reset to "0" (On) by TC in order to continue trying to deploy the panels. The entire process is implemented in the

PCDU and is performed automatically. The On-Board Software (OBSW) and the OBC are not needed yet and operate completely independently—as can be seen from Fig. 4.5.

When the PCDU has booted into MiniOps Mode, it activates the power circuits of the OBC, resulting in the OBC booting on its nominal side with Processor-Board N and I/O-Board N being active. The OBSW then detects the last S/C mode prior to boot having been Off Mode and thus starts S/C operation by setting the satellite automatically into Detumble Mode. As the PCDU MiniOps Mode is very limiting, after the OBSW is up and running it will command the PCDU into its Idle Mode.

This implies also setting the "Attitude Control System" (ACS) into Detumble Mode which then starts to reduce rotational rates of the satellite resulting from the launcher separation by means of magnetotorquers, measuring the achieved attitude by means of sun sensors and magnetometers.[2] Furthermore, other vital parts of the OBSW such as the controllers for "power supply subsystem", (PSS), the "telemetry and telecommand subsystem", (TTC), and the "thermal control subsystem", (TCS), are set to their standard modes.

During these initial operational steps of the OBSW it generates the according boot Event TM (being written to the TM packet store on the I/O-Board) for later downlink. The OBSW furthermore collects all relevant TM of initial rotational rates, the deployment status data from the PCDU and the number of deployment attempts and allocates the data in according system TM packets for downlink.

Potential OBSW reboots after software failure, OBC reconfigurations after hardware failure or entire satellite reboots after complete satellite power failure are covered in the diverse subsections of Chap. 10.

4.5 OBSW Controlled Functions

The Core DHS hardware including the running OBSW controls all functions on board the satellite. The overall OBSW controlled functions comprise:

- Spacecraft operational modes and mode transitions
- Subsystem modes and mode transitions
- All management function of the subsystems Power, TTC, ACS, TCS and Payload (including control of science data downlink)
- All control loops of the subsystems Power, ACS and TCS
- Mission timeline driven functions
- Orbit position scheduled functions
- And in the completeness of the CDPI architecture consisting of OBC and PCDU also all FDIR functions of the spacecraft—be it OBSW controlled or purely hardware controlled.

[2]Magnetotorquers and magnetometers are operated in interleaved mode to avoid the magnetotorquers biasing magnetic field measurements.

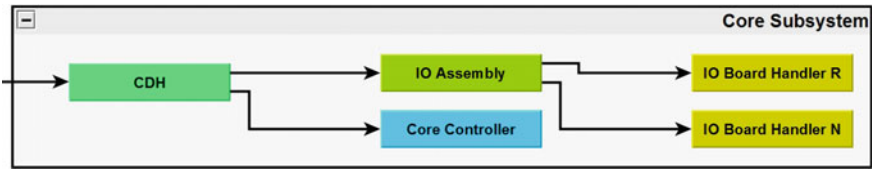


Fig. 4.6 Core DHS subsystem control elements in OBSW. © IRS, University of Stuttgart

These diverse function are explained in detail in the corresponding subsystem chapter and Chap. 10.

Here however an ambiguity has to be clarified: While the entire OBSW covers all the previously listed functions, the *CoreDHS_Subsystem* inside the OBSW is "only" the part which controls the OBC electronics—see also Fig. 4.6. Therefore the Core DHS as subsystem only comprises the device handlers, assemblies etc. which are required for this task.

4.6 Core DHS Subsystem Control

The OBSW and the Core DHS hardware are inherently interwoven as the OBSW runs on the Processor-Boards. The core data handling software concept has already been discussed in Chap. 3. In this chapter now only the software handlers and controllers for the Core DHS hardware are described (see Fig. 4.6).

The Core DHS hardware is handled by four dedicated software objects, which manage the functionality of the Processor-Boards boards. The objects are the I/O-Board handlers and the associated I/O-Board assembly, and a so called *Core_Controller*, which manages functionality of the Processor-Board itself.

The CCSDS-Boards are physically placed in the OBC housing. Logically they belong to the TTC subsystem. Therefore their control assembly and handler description is provided in Sects. 6.3.1.2 and 6.3.2.2.

4.6.1 I/O-Board Handler

The I/O-Board handler has two main objectives:

- Handling I/O-Board modes
- Accessing the I/O-Board memory

The board can be commanded into two different modes (MODE_ON and MODE_OFF), with a number of submodes. These modes are listed in Table 4.2. The additional "No Switch" submode results from the fact that the I/O-Board can be turned off by the PCDU autonomously.

Table 4.2 I/O-Board device handler modes

Object-IDs	Mode	M#	Submode	SM#	Description
I/O-Board N/R 0x44450000 0x44450100	Off	0	Default	0	Off
			No Switch	1	Used to command Board to Off without sending a switch command. Used when no PCDU connection is available. Handler will enter Mode (Off, Default) when successful
	On	1	Passive	0	Board On, memory access possible, communication with external devices not possible
			Active	1	Board On, memory access possible, communication with external devices is possible
			No Switch	255	Used to command Board to On without sending a switch command. Used when no PCDU connection is available. Handler will enter Mode (On, Passive) when successful
	Waiting_for_Board	16	Default	0	Waiting for Space Wire Activation Mode cannot be commanded but is a transition state entered when commanded to On and no active SpaceWire connection exists. This mode is left to (On, Passive) as soon as an active SpaceWire connection is detected

The I/O-Board handlers also mange low-level access to the on-board memory of the I/O-Board. The memory regions and their usage are described in Sect. 4.3.2. All higher level features such as the TM packet store use these access methods. In

addition to accessing the MRAM banks, the handler also allows a memory-like access to the device communication buffers on the I/O-Board. A write command to a given buffer will effectively forward the written data to the associated device, a read command will read data from that device. This feature is not used during nominal operations (device communication is handled directly with RMAP calls), but is provided for contingency operations.

4.6.2 I/O-Board Assembly

The main task of the I/O-Board assembly is to keep device communication active under all circumstances. This is a complex activity, as communication with the PCDU, which is necessary to turn switches on and off, also reconfigures the I/O-Boards in case of failure—see Sect. 10.2. In addition, the PCDU is allowed to reconfigure the I/O-Boards by itself, an activity for which the assembly must be prepared to avoid unnecessary software reboots. Also, the I/O-Boards may be operated in single or dual mode, the latter with one board active doing communication and one board passive, but permitting memory access.

This leads to a complex state machine to check the current state of the boards and to always reach a safe state. Controlling this machine is achieved by the modes available for the assembly and the health flags of the I/O-Board handlers. These may be overwritten by the assembly in certain cases. Table 4.3 describes the available modes of the I/O-Board assembly. The overall OBC reconfiguration workflow—including the I/O-Boards is explained later in the FDIR chapter in Sect. 10.2.3.

The I/O-Board assembly furthermore routes memory commands to the currently active I/O-Board, i.e. if an application—such as a TM store—intends to write data to the active I/O-Board, it may send the request to the I/O-Board assembly, which forwards the request to the active board or triggers an error if no board is available.

Table 4.3 I/O-Board assembly mode table

Object-IDs	Mode	M#	Submode	SM#	Description
I/O-Board 0x41010000	Off	0	Default	0	Both boards are Off, i.e. PCDU Recovery is forced until OBC reboot. Commands are still possible in Off mode
			Recover	1	Forces first step of PCDU Recovery (I/O-Board Off and the other one On), then back to On, Single mode. Commands are still possible in Off mode
	On	1	Single	0	One board active, the other one Off
			Dual	1	One board active, the other one passive

4.6.3 Core DHS Controller

The *Core_Controller* provides a number of functions to handle the Processor-Board hardware and the overall software. They are listed in Table 4.4.

There is no mechanism inside the I/O-Board electronics to directly mirror data from the active I/O-Board to the cold redundant one. Stored data therefore would only be recoverable with direct memory access as a contingency operation or even would be lost in case of a board defect.

Therefore the *Core_Controller* can be commanded to cyclically activate the redundant board in passive mode (i.e. not electrically activating its equipment interface circuitry) and to copy data stored on the nominal board over to the redundant one with the Memory Management Service via the I/O-Board Handler.

The controller uses the general system-wide mode concept (i.e. it can be commanded with default mode commands) to control the execution of the inter-board synchronization:

- If the controller is in OFF mode, it does not synchronize
- If the controller is in ON mode, it will synchronize the two boards every 10 min.

Table 4.4 *Core_Controller* functions

Function Name	ID	Parameters	Information
SET_PROM_SLEEP_STATE	1	PROM ID (0/1) (1B), AWAKE (0/1) (1B)	Sets the sleep line of one individual PROM (see Sect. 4.3.1)
SET_READ_WRITE_STATE	2	ENABLED (0/1)(1B)	Sets the write state of all PROMs (see Sect. 4.3.1)
READ_PROM_STATE	3	–	Returns PROM sleep and write states
WRITE_ON_BOARD_TIME	4	OBT offset (4B)	Set the seconds of OBT already elapsed
RESET_CPU_USAGE_ STATISTICS	10	–	Resets CPU utilization statistics.
READ_CPU_USAGE_STATISTICS	11	–	Dump CPU utilization statistics
SET_SECONDARY_COUNTER	20	New secondary counter (4B)	Sets the boot count for the secondary image
READ_SECONDARY_COUNTER	21	–	Reads the boot count for the secondary image

4.6.4 Core DHS Mode Transitions and Telemetry

The Core DHS subsystem mode commanding is performed via PUS service 200 and the following sequence—see Table 4.5. The Core DHS subsystem itself only has one Default Mode.

The satellite system transitions controlled by the Core DHS have already been listed in Table 2.1 in Chap. 2:

The System, OBC and I/O-Board relevant TC packet definitions are included in the spacecraft TC table in annex Sect. 17.3.1.

The System, OBC and I/O-Board relevant TM packet definitions are included in the spacecraft TM table in annex Sect. 17.3.2.

The System, OBC and I/O-Board relevant Event TM packets are included in the spacecraft Event TM table in annex Sect. 17.3.3.

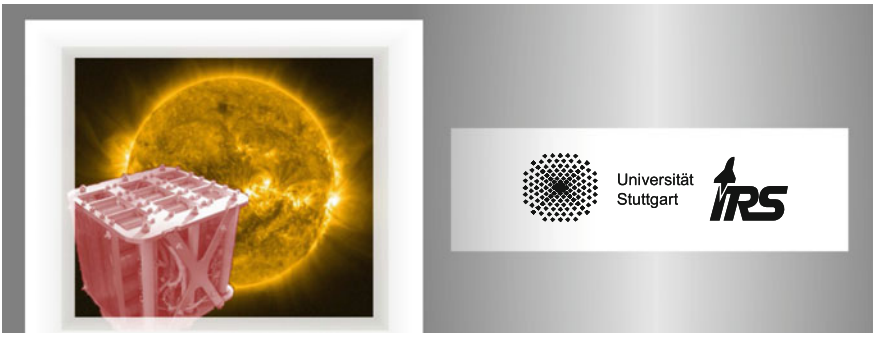
Details on the TM parameter positions in the diverse packets have to be taken from the MIB.

Table 4.5 Core DHS subsystem modes and transition sequences

#	**CDH Modes**	**0x1000600**				
Commanding to Default Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Default	0x29000000	1			
	Name	**Table ID**	**WaitTime**	**Check?**		
	Default_Target	0x29000000	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	I/O-Board Ass.	0x41010000	On	1	Single	0
#						
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Default_Trans	0x29000100	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	I/O-Board Ass.	0x41010000	On	1	Single	0

Chapter 5
Power Supply Subsystem

Kai-Sören Klemich and Bastian Bätz



Abstract The chapter contains first an overview on the power supply subsystem and discusses all the components. These are the solar panels including the deployment mechanism, the battery and in particular the Power Control and Distribution Unit (PCDU). The PCDU modes and its particular functions in the frame of the Combined Data and Power Management Infrastructure (CDPI) are explained. The last part comprises the onboard software elements and functions for control of the power supply subsystem. Additional tables are provided in the book's annexes.

Keywords Power subsystem · Solar panels · Battery · Power Control and Distribution Unit · Specific PCDU functions in the CDPI architecture · Power subsystem control

K.-S. Klemich (✉) · B. Bätz
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: klemich@irs.uni-stuttgart.de

B. Bätz
e-mail: baetz@irs.uni-stuttgart.de

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_5

5.1 Subsystem Overview

The power subsystem of the FLP platform is based on an unregulated power-bus, solar arrays, a Power Control and Distribution Unit (PCDU) and a battery. The power subsystem is marked in red in Figs. 1.6 and 5.1 provides a zoomed view on the main components.

The battery and the solar panels are connected to the PCDU. The power-bus voltage is in the range 20–25.5 V according to the state of charge of the battery and to the battery current.

All power consuming spacecraft equipment is connected to the non-regulated power-bus through power lines (see Fig. 1.6). The power connections are switched and protected by Latching Current Limiters (LCL) inside the PCDU. The LCLs are commanded by the OBC via the RS422 digital link to the PCDU. Obviously the OBC itself is directly connected to the non-regulated bus, without LCL. The protection of the bus for the OBC itself is ensured by the OBC internal Power-Boards (see [47]).

The FLP Generation 1 has two deployable solar panels—panels 0 and 1—and one body mounted paner—panel 2. The platform in this configuration does not provide a solar array drive to reduce complexity. More details are given in Sect. 5.2. The power subsystem is controlled by the *PSS_Controller* in the OBSW.

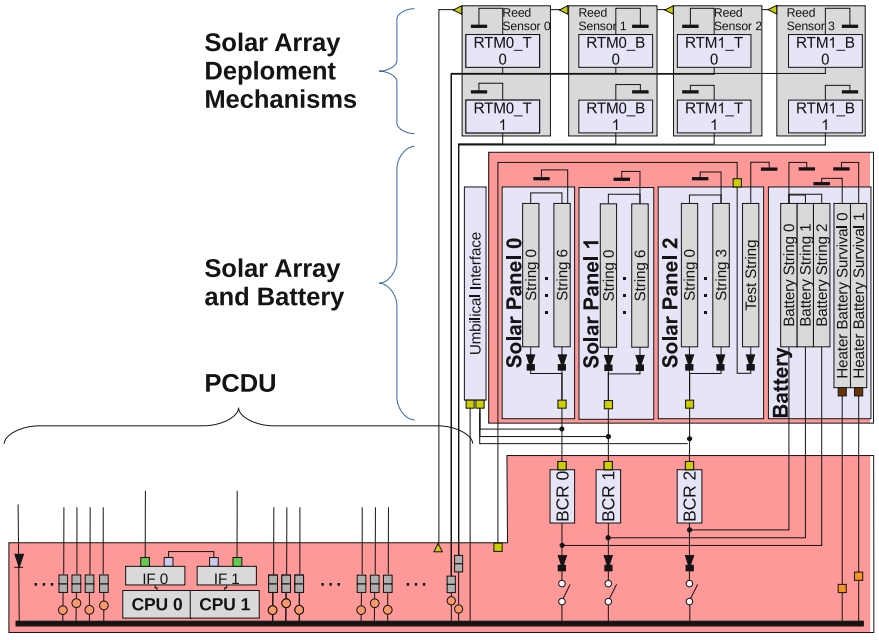


Fig. 5.1 Power supply subsystem overview. © IRS, University of Stuttgart

5.2 Solar Panels

The three solar panels are implemented in two different configurations, a side panel type which is deployed after separation of the spacecraft from the launcher, and a center panel type which is body mounted (please also refer to Figs. 1.2, 1.3, and 1.5).

The GAGET1-ID/160-8040 [59] solar cells from AZUR Space Solar Power are applied as primary energy source (see Table 5.1). For the first FLP mission "Flying Laptop" the center solar panel includes an additional test string with experimental solar cells RWE3G-ID2*/150-8040 from AZUR Space Solar Power [60] with an efficiency of 27.8 % (BoL, 28 °C, AM0). These shall be qualified for space use during the first FLP mission.

The solar panels of the first FLP satellite have been produced by Airbus DS GmbH in Ottobrunn, Germany. The maximum output power is approximately 230 W. The solar array configuration of the first FLP satellite—limited by geometric size to fit into the envelope of a secondary launcher payload—is as follows (see Figs. 5.2 and 5.3):

- Two deployable panels (panel 0 and panel 1) with the following characteristics:
  - 730 × 520 mm
  - 7 strings—15 solar cells each
  - 105 solar cells overall
- One body mounted center panel (panel 2) with:
  - 780 × 680 mm
  - 4 strings—16 solar cells each
  - plus one test string made of 16 RWE3G-ID2*/150-8040 cells
  - 80 solar cells overall

On the body mounted panel, which will be warmer due to increased heat conduction from the satellite's main body, there are four strings of 16 solar cells each in order to achieve a similar operating voltage to the cooler wing panels.

This design with a string length of 15 respectively 16 solar cells leads to a maximum power-bus voltage between 20.0 and 25.5 V for the first FLP based mission "Flying Laptop". The general FLP platform design however allows for a standard 28 V power-bus.

Table 5.1 Overview of the FLP solar cells and battery key characteristics

Solar cells	
Identification	GAGET1-ID/160-8040
Base Material	GaInP2/GaAs/Ge on Ge substrate
Efficiency at BOL, 28 °C, AM0	25.3 %
Maximum power output in FLP configuration	approximately 230 Watts peak

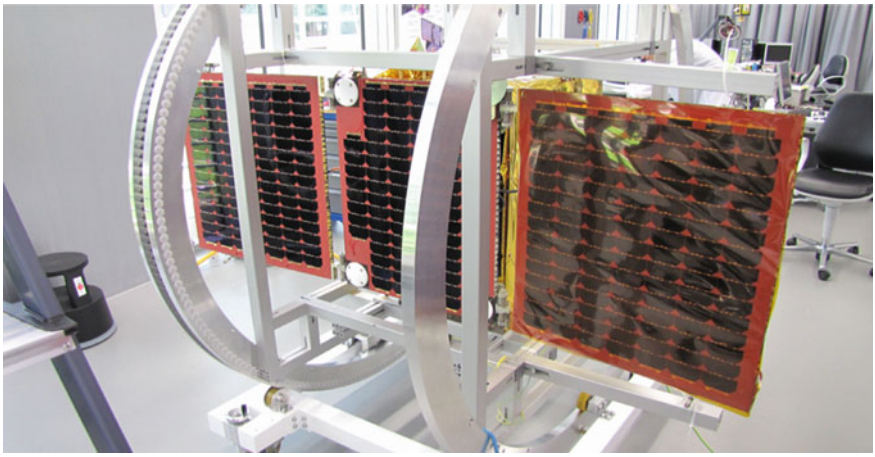


Fig. 5.2 "Flying Laptop" solar array. © IRS, University of Stuttgart

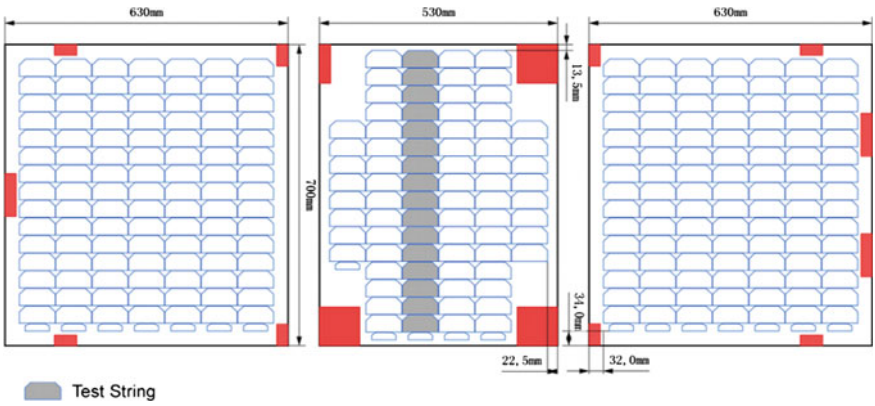


Fig. 5.3 Solar panels configuration. © IRS, University of Stuttgart

The panels feature shunt diodes for every cell so that damaged or shaded cells do not cause an entire solar cell string to fail.

5.3 Solar Array Deployment Mechanism

The solar array deployment is activated via a melting braid mechanism cutting a nylon braid. This material was selected due to its high tensile strength and the relative low melting temperature. The mechanism provides a re-usable, very compact and adapted mechanism for the FLP platform with a very low shock load.

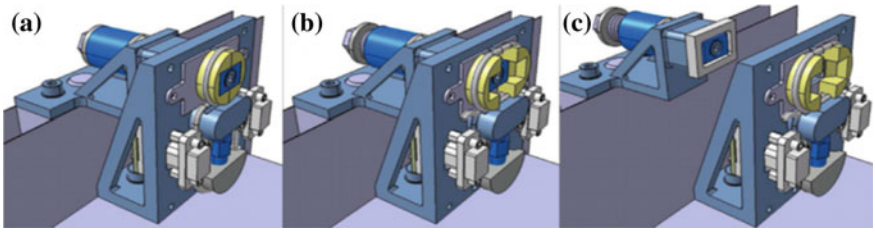


Fig. 5.4 Solar array release steps. © IRS, University of Stuttgart

The retaining mechanism is shown in Fig. 5.4. It is divided into a solar panel bracket mounted to the solar panel and a main body bracket attached to the satellite main body. Both brackets are connected with a retaining bolt, having a wedge on the end. The retaining bolt is preloaded by a spring, located on the solar panel end, trying to pull the wedge out of the main body bracket. The pull-out is prevented by a braid around the divided ferrule that avoids a coming asunder of the two parts by the wedge effect. The braid is pre-stressed by a clamping system consisting of a clamping claw and a bolt with right-hand and left-hand thread. The tightened clamping bolt constitutes a pre-tensioning force to the wire.

The function of the release system can be seen in Fig. 5.4. In Orbit the two pairs of two melting resistors will be activated and heating up each braid on two opposite positions (in the area of the clamping bolt). As melting element ceramic resistors are used. In every resistor pair the melting braid is lead between the two heaters (one on the right hand and one on the left hand). So the braid is heated up on two positions from two sides. By warming up the braid the strength decreases until the pre-tensioning force of the clamping system and the retaining bolt will tear the braid apart. If the melting braid is loose, the divided ferrule comes asunder by the force generated by the wedge system. With enough distance of the divided ferrule the wedge is free and can be released through a hole in the main body bracket. Now the solar panel is free and is deployed by the pretension in the spring of the hinge mechanism—see Fig. 5.6.

To prevent a single point failure in the electrical supply of the melting heater two independent supply lines exist. The electrical configuration is shown in Fig. 5.5. Every retaining mechanism is powered by line 1 and line 2. After activating the power switches in the PCDU the heaters are directly connected with the battery. The generated power of the heaters depends on the voltage of the battery. With a

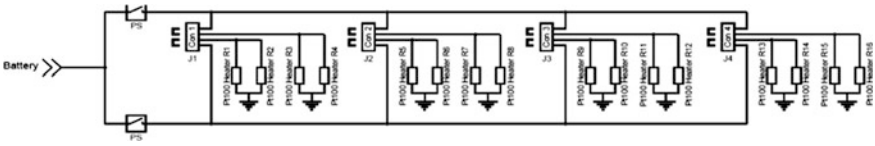


Fig. 5.5 Electrical supply of the clamping band melting heaters. © IRS, University of Stuttgart

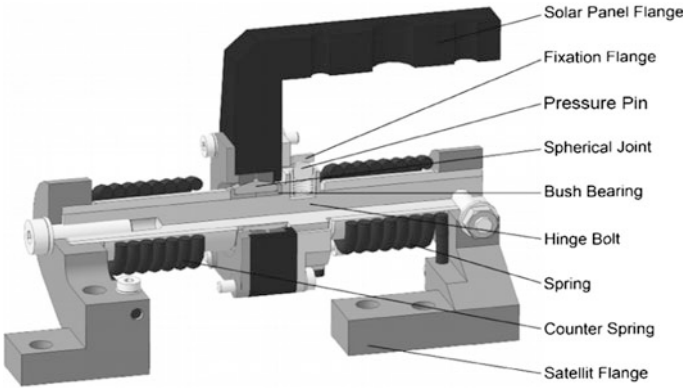


Fig. 5.6 Solar array deployment mechanism. © IRS, University of Stuttgart

battery voltage of 24 V every heater consumes a power of 5.8 W. With all 16 heaters, the power consumption adds up to 92.2 W.

It is foreseen that the heaters are activated by the PCDU after it has has booted in result of the launcher separation. The process is controlled by a timer as the launch supplier will prescribe a delay interval between the launcher separation and the satellite boot. Therefore after launcher separation detection a preset counter in the PCDU is continuously being decreased. It can be adapted to the launch supplier's requests on ground. For the detailed sequence after launcher separation see Sect. 11.6.1.

In order to prevent an activation of the mechanism by accident, a protection connector is foreseen. Only with this additional connector (green-tagged item) applied to each device the two electric circuits are closed and the mechanism can be activated (see also Sect. 17.13).

Furthermore, a deployment detector is installed on each retaining mechanism. A magnet activates a reed sensor in launch configuration. With the solar deployment the reed sensor contact opened and indicates the correct function of the release to the PCDU.

The solar array hinge mechanism (see Fig. 5.6) turns the side panels around by 90° and holds them in their position. In order to meet these requirements at each side of the solar side panels two hinges are foreseen. The hinge systems implement the connection of the side solar panels with the satellite main body. In order to create a compact configuration with a suitable load path a fork fixture is implemented. The solar panel flange is connected via a hinge bolt to the fork shaped satellite flange. The mechanism is mounted to the main satellite structure via a glass-fiber reinforced bracket which insulates the solar panel in an electrical and thermal way from the satellite body. The middle solar panel also is connected via these four brackets.

The lower hinge is designed as fix bearing and the upper hinge as loose bearing. The fix bearing is generated by fixing the movement of the hinge in axis direction

while the loose bearing offers a degree of freedom in the direction of this axis. The rotation of the hinge is created via a bush bearing allowing a rotation between the hinge bolt and the solar panel flange. Furthermore, a ball bearing is integrated into each hinge. With these bearings the hinge system can balance misalignments in the solar panel mounting.

The rotation force is created by torsion springs. For damping and reaching the end position an additional counter torsion spring is installed. So the solar array can "swing" to the release position, can over-swing a bit due to its dynamic movement and is rotated back to the foreseen 90° deployed position.

5.4 Battery

The FLP Generation 1 battery consists of ANR26650M1-B lithium iron phosphate (LiFePO4) cells manufactured by A123 Systems [58]. Each cell has a nominal capacity of 2.5 Ah at a nominal voltage of 3.3 V. The battery consists of dedicated cell strings. Each battery string is directly connected to one of the solar panels.

- The strings for the side panels (string 0 and 1) consist of 35 battery cells each, with blocks of five cells in parallel and seven of these blocks in series, yielding a total capacity of 12.5 Ah at a voltage between 18.9 and 25.2 V.
- For the center panel string (string 2), there are only four battery cells connected in parallel, reducing the nominal capacity to 10 Ah.

As result the battery has a nominal BOL capacity of 35 Ah or 805 Wh at 23 V (Table 5.2).

The battery EM is depicted in Fig. 5.7. Figure 5.8 shows a functional diagram of one string of the battery with the interface to the PCDU, the battery cells, the thermal control devices and the Balancing and Charge Control (BCC) circuitry.

The battery internal thermistors and the battery compartment heaters are covered in the thermal subsystem description in Chap. 8.

The basic battery values measured by the PCDU are voltages, currents and temperatures for the three battery strings.

As the battery cells lose capacity at low temperatures, the battery strings can be heated. The heaters are supplied with power by the PCDU through bi-stable relays, which are usually switched in normal operations. The actual switching of the heaters is done by thermostat switches on the battery bottom plate. If the temperature of the thermostat switches falls below ca. 0 °C, the heaters are activated, if it rises above ca. 5 °C the power supply is interrupted again. There are two redundant heater circuits, each of which consists of three single heaters, one at each string.

Furthermore, each of the battery strings features an electronic board, which performs a basic cell balancing and can send a signal to the PCDU so stop charging of the respective string if one of the cell blocks has reached its maximum voltage. The electronic boards consume approximately 9 mA for battery string 0 and 1 and 8 mA for string 2, depending on the battery voltage. The balancing is achieved by

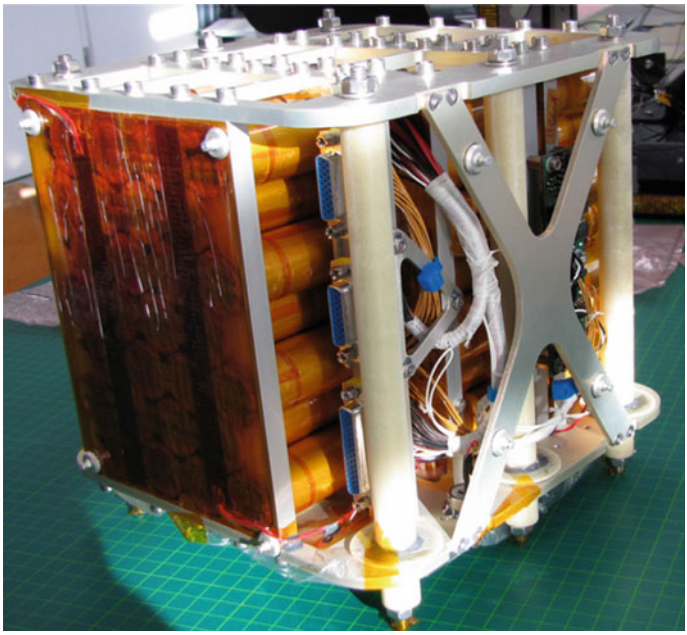


Fig. 5.7 FLP battery engineering model. © IRS, University of Stuttgart

bleed resistors which discharge all cell blocks in a battery string at all times. As the discharge current depends linearly on the cell block voltage, cell blocks with higher voltages are discharged at higher currents. Thus a balancing effect occurs at the cost of a constant bleed current, which is approximately half of the total current consumed by the battery electronic boards [157].

The voltage monitoring of the cell blocks causes the PCDU to stop charging of a string if a maximum cell block voltage of approximately 3.65 V is reached. The charging is interrupted for a configurable time to allow the battery voltage to drop and is then resumed until the charge stop signal is issued again. The response of the PCDU to the charge stop signal can be disabled by setting a flag in the PCDU in case of a failure of the balancing boards.

The SoC of the battery is estimated both in the PCDU and in the power controller (see also details later in Sect. 5.4.2). The algorithm in the PCDU is a simpler version, neglecting any temperature effects on the battery capacity. It is used as a backup and is necessary when the satellite was shut down due to power outage. During power recovery the PCDU waits until the battery SoC is above a threshold of 45 % before booting the OBC. During this time, it must calculate the SoC independently of the OBSW.

In case of the first FLP mission due to the limited power-bus voltage not providing the entire 28 V standard, the PCDU charges the battery with a constant voltage of 25.5 V. Thus, this also defines the maximum battery string voltage. Its minimum voltage is 19.0 V, which equals roughly 0 % SoC, at which the hardware

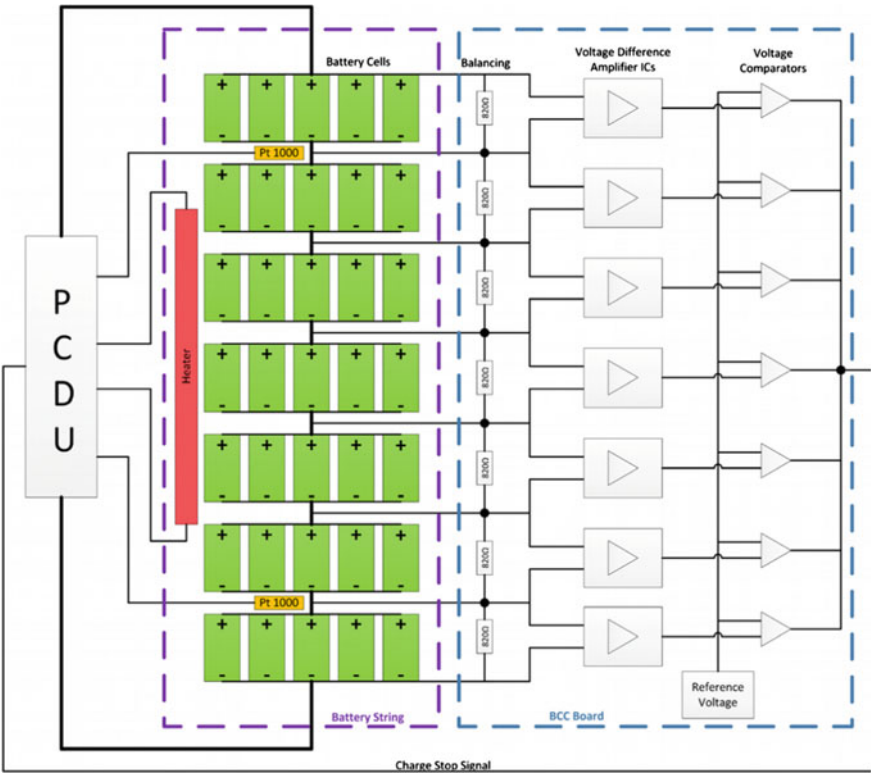


Fig. 5.8 The FLP battery and Battery charge control board layout. © IRS, University of Stuttgart

Table 5.2 Overview of the FLP battery key characteristics

Battery	
Type	Lithium iron phosphate
Identification	ANR26650M1-B
Total capacity for the configuration, BOL	35 Ah

under voltage protection of the PCDU is triggered. Note that the PCDU keeps the three battery strings at very similar voltages, i.e. roughly similar SoC, at all times.

5.4.1 General Monitoring

Similar to the solar panels, the currents and voltages of the three battery strings as well as their temperatures are measured by the PCDU. Note that a positive battery current indicates charging of the string and a negative current indicates discharging.

The charge/discharge power for the battery strings is calculated by the power controller.

There are two Pt1000 thermistors per string, one at the top and one at the bottom of each string (see also [157]), i.e. six temperature values in total. Here "bottom" is referring to the side at which the battery is mounted on the service module plate. Again the PCDU provides the resistances of the thermistors and the *TCS_Controller* calculates the actual temperatures—see Sect. 8.3.

All of these values are part of the *datapool* and can be downlinked using Service (3.25). An overview of the battery TM parameters, their TM packets and the displays they appear on is given in Table 5.3.

Note that as the battery heaters are not supplied with power via fuses but via bi-stable relays, the heater current cannot be measured. The activation of the heaters during nominal operations can only be detected indirectly, e.g. by analyzing the bus and/or battery current and the battery temperature.

Table 5.3 Battery telemetry parameters

Name	Description	Unit	Long Description	Display	SID
PBTSTC00	BAT_STR_CURRENT	A	Battery String Current 0	PBA00000, PBG00001	4000
PBTSTC01	BAT_STR_CURRENT1	A	Battery String Current 1	PBA00000, PBG00001	4000
PBTSTC02	BAT_STR_CURRENT2	A	Battery String Current 2	PBA00000, PBG00001	4000
PBTSTV00	BAT_STR_VOLTAGE	V	Battery String Voltage 0	PBA00000, PBG00001	4000
PBTSTV01	BAT_STR_VOLTAGE1	V	Battery String Voltage 1	PBA00000, PBG00001	4000
PBTSTV02	BAT_STR_VOLTAGE2	V	Battery String Voltage 2	PBA00000, PBG00001	4000
PBTPWR00	BAT_STR_POWER_0	W	Battery String 0 Power	PBA00000, PYA00001	4000
PBTPWR01	BAT_STR_POWER_1	W	Battery String 1 Power	PBA00000, PYA00001	4000
PBTPWR02	BAT_STR_POWER_2	W	Battery String 2 Power	PBA00000, PYA00001	4000
TSTTMP00	TEMP_EXTERNAL	degC	TCS Calibrated PCDU external Temp 00 Battery String 0 bottom	PPA00003	4206
TSTTMP01	TEMP_BATT_S1_BOT	degC	TCS Calibrated PCDU external Temp 01 Battery String 1 bottom	PPA00003	4206

(continued)

Table 5.3 (continued)

Name	Description	Unit	Long Description	Display	SID
TSTTMP02	TEMP_BATT_S2_TOP	degC	TCS Calibrated PCDU external Temp 02 Battery String 2 top	PPA00003	4206
TSTTMP04	TEMP_BATT_S0_TOP	degC	TCS Calibrated PCDU external Temp 04 Battery String 0 top	PPA00003	4206
TSTTMP05	TEMP_BATT_S1_TOP	degC	TCS Calibrated PCDU external Temp 05 Battery String 1 top	PPA00003	4206
TSTTMP06	TEMP_BATT_S2_BOT	degC	TCS Calibrated PCDU external Temp 06 Battery String 2 bottom	PPA00003	4206
PPTRTS00	PCDU_TEMPSENSRES	Ohm	PCDU Temp Sensor Resistance 00 Battery String 0 bottom	TSA00000	5200
PPTRTS01	TSRS_BATT_S1_BOT	Ohm	PCDU Temp Sensor Resistance 01 Battery String 1 bottom	TSA00000	5200
PPTRTS02	TSRS_BATT_S2_TOP	Ohm	PCDU Temp Sensor Resistance 02 Battery String 2 top	TSA00000	5200
PPTRTS04	TSRS_BATT_S0_TOP	Ohm	PCDU Temp Sensor Resistance 04 Battery String 0 top	TSA00000	5200
PPTRTS05	TSRS_BATT_S1_TOP	Ohm	PCDU Temp Sensor Resistance 05 Battery String 1 top	TSA00000	5200
PPTRTS06	TSRS_BATT_S2_BOT	Ohm	PCDU Temp Sensor Resistance 06 Battery String 2 bottom	TSA00000	5200

5.4.2 *Battery SoC Estimation*

The battery State-of-Charge(SoC) is estimated both in the PCDU software as well as in the power controller of the OBSW. In general, it is assumed that the value calculated by the power controller is more reliable, because it takes into account temperature effects. Therefore, in its normal mode, the *PSS_Controller* in the OBSW updates the current SoC information in PCDU *once per hour* and throws an event (RID 4011) when the difference between the two values (OBSW and PCDU) exceeds 0.5 Ah (see Sect. 5.6.2.6).

A simple ampere hour counting algorithm is used for the SoC estimation, which is executed *once a second* both in the PCDU and the OBSW. The currents flowing in and out of the three battery strings measured by the PCDU are added to the discharge currents of the battery electronics. The resulting total string currents are multiplied by a predefined thermal loss factor, depending on the direction of the string current. These current values are then multiplied by one second and the resulting values in ampere hours are added to the previous capacity values. These new capacity values are then divided by the strings' maximum capacity values and lead to a SoC value in percent for each string. The total capacity of the entire battery is calculated by adding up the capacities of the three strings and the total SoC is calculated by dividing the total capacity by the sum of the string maximum capacities.

This estimation algorithm necessarily accumulates an error over time. The error is caused by several factors, including the limited time resolution, i.e.

- the numerical integration of the currents at discrete points in time,
- current measurement errors of the PCDU,
- and most importantly the inaccuracy of the constant thermal loss factors, which in reality are a function of the battery SoC, the actual amperage, battery age and the battery temperature.

Therefore, a calibration functionality exists, which sets the SoC of a string to 100 % if three criteria are met:

- The string current must be between zero and 0.3 A (i.e. the battery is not discharged and the charge current is small, indicating that the string is possibly nearly fully charged),
- the string voltage must exceed 23.5 V (also indicating that the string is nearly fully charged) and
- the SoC must exceed 96 %.

Note that the PCDU uses a constant voltage charge strategy, necessarily resulting in a small residual charge current when 100 % SoC is approached. However, small charge currents can also occur when the solar panels are slowly being oriented away from the sun direction, which is why the current criterion is not sufficient.

Note also that the string capacities can be set by command to both the PCDU and the power controller. This allows to bypass the SoC criterion by the ground in case the battery is determined to be fully charged, but its estimated SoC is too low for the calibration to occur. The *PSS_Controller* throws events for each string when the calibration occurs (Event TM RID 4000) and when the capacity is at 100 %, but no calibration occurs (due to at least one of the other calibration criteria not being met, RID 4032).

To avoid obvious wrong results of the SoC estimation, some safety features are implemented. The string currents and voltages are limit checked before the actual SoC estimation is performed. If a string current is out of the range −12 to 12 A or a string voltage is out of the range 19–26 V (limits here again for the FLP mission

"Flying Laptop"), the SoC estimation is not performed due to obviously wrong measurements so that the SoC value stays the same. The TM parameter calculated by the *PSS_Controller* in such case is also marked as invalid. Events are also thrown by the *PSS_Controller* when these limits are violated (RIDs 4020, 4023, 4026, 4029).

The functionality described so far is the basic functionality of the SoC estimation as implemented in the PCDU. In addition to this basic functionality the *PSS_Controller* takes into account the reduced capacity of the battery at temperatures below 20 °C.

When the temperature of a string (combined from both temperature sensor values) drops below 20 °C a linear reduction of the maximum capacity of the string of 0.05 Ah/K is assumed. Furthermore, the thermal losses become larger at lower temperatures which is accounted for in the SoC estimation by increasing the thermal loss factors linearly by 0.00025 1/K if the string temperature is below 20 °C. In addition to the limit checks described above, there is a lower temperature limit on the string temperature of −45 °C If the string temperature is below this limit, the SoC estimation will not be performed.

All parameters of the SoC estimation mentioned here can be set in the PCDU by direct commands and in the *PSS_Controller* by function commands (see Sects. 5.6.2.3 and 5.6.2.6).

There is also a function of the *PSS_Controller* to write the parameters configured in the controller to the PCDU to ensure consistency.

Note: This function does not set the string maximum capacity values.

5.4.3 Operating the Battery

During in-orbit operations, it shall be made sure that the battery string SoC estimation is calibrated at least *every 2 weeks* to avoid large errors in the SoC estimation. If the calibration is impossible due an accumulated error in the SoC estimation, this can be resolved by setting the current capacity values of the affected strings in the PCDU and the *PSS_Controller* manually. If the calibration does not occur due to one of the other conditions not being met, it is likely that a sensor is defective and the limits may be reset. If a battery string current sensor is defective, the SoC estimation of the affected string cannot be performed anymore.

Due to degradation of the battery cell capacities over time, the maximum string capacities must be reset on a regular basis. As described in [76], a reduction of 0.1 % of the strings' BoL capacity shall be commanded every 2 weeks.

The battery temperature must be monitored permanently. The heater circuitry is designed to be redundant to make sure that the battery temperature does not drop significantly below 0 °C. However, the battery system cannot be actively cooled. Therefore, high battery temperatures close to the cells operational temperature limit of 55 °C constitute a high risk for the mission. If only one sensor or one string shows a high temperature, it may be due to failure of the temperature sensor or due

to a failure in the battery string, e.g. a broken cell. The former problem can be resolved by setting the temperature sensor to faulty if that has not been done by the FDIR (see Sect. 10.3.3.3), the latter problem can be resolved by setting the battery string to faulty (see Sect. 5.6.2.6), which results in the string not being charged anymore. Note that the string can still be discharged in this case. If the temperature of the entire battery is too high, the charge currents of the affected strings are reduced by stopping to charge the affected strings. For details on this problematic scenario see end of Sect. 10.3.3.3.

Setting a battery string to FAULTY may also be used to cope with other failure scenarios, e.g. out-of-limits current or voltage measurements both as a short term contingency measure and as a long term configuration in case TM analysis has shown that a string is actually broken. In the latter case, the maximum capacity of the affected string should be set to 0 so that it is not taken into account anymore by the SoC estimation algorithms.

Low SoC values of the battery trigger Events at prescribed power levels as described later in Sect. 5.6.2.5 and in more detail in Sect. 10.3.1.

- At 50 % SoC, the warning event (RID 4012) is thrown, which causes the FDIR to switch off the payload if it was switched on. At 30 % SoC, the FDIR commands the satellite to safe mode based on the "SOC_CRITICAL" event (RID 4013)—which corresponds to a SW-DNEL event.
- At 20 % SoC the "SOC_SHUT_DOWN" event (RID 4014) is thrown and the satellite is entirely deactivated including ONC and TTC and PCDU and will not reboot before SoC has regained a certain value during free tumbling. For more information on the staggered reaction to power loss refer to Sect. 10.3.1 which is derived from [132].

Both events are thrown by OBSW. So the classic design of one SW-DNEL is refined here to two steps.

5.5 Power Control and Distribution Unit

The FLP follows the standard implementation approach for the primary power source and energy storage device:

Primary power source	Photovoltaic solar cells
Secondary energy storage	Battery cells

The Power Control and Distribution Unit (PCDU), performs the power regulation, control and distribution from the power sources to the consumers in a satellite system. Furthermore, the PCDU is responsible for monitoring and protecting the satellite power-bus. The FLP platform Power Control and Distribution Unit is developed in cooperation with an experienced industrial partner, Vectronic Aerospace GmbH in Berlin, Germany.

The FLP platform features a so-called Combined Data and Power Management Infrastructure (CDPI) where the PCDU itself is equipped with additional functionality as already outlined in Sect. 1.8 and detailed further in Sect. 4.2. This chapter describes both the specific PCDU functionality which enable this CDPI concept from the PCDU perspective as well as the standard PCDU functions themselves in more detail.

5.5.1 Power Control and Distribution Functions

The main task of the PCDU is the distribution and regulation of the electric power on board the satellite. The power handling design is specified in order to safeguard the power supply of the satellite bus as far as possible. Furthermore, specific protection features are implemented in order to prevent damaging of the on-board components or the battery which are essential for accomplishing the mission objectives. Figure 5.9 shows the circuitry of the PCDU and its connections to the satellite bus.

The PCDU does not perform a classic maximum power point tracking of the solar panels and there are no shunts dissipating unused energy. Instead the PCDU adjusts the operating point of the solar panels such that the voltage is optimized to supply the battery and the loads with power. Thus besides the panel temperature and illumination, the actual operating point of the panels depends on the battery voltage and the system's current power consumption. For example when the satellite exits the eclipse with the panels oriented to the sun, the voltage needed to charge the battery and supply power to the system is typically lower than the maximum power point voltage of the solar panels, so the panels are not used to their full potential. However, this loss in power is accepted as the panel voltage is generally in the same range as the necessary charge voltage so the losses are limited and the system design is very simple and robust.

Each solar panel is connected to only one battery string by a Battery Charge Regulator (BCR), in order to prevent a single point failure. The FLP target satellite configuration represents a non-DET system [141] with an unregulated power-bus. The BCR is located in the direct energy path to protect the satellite bus from excessive voltage or current transients.

As explained in Sect. 5.2 the solar arrays of the first FLP based satellite "Flying Laptop" are limited with respect to the maximum voltage and do not provide the full standard of 28 V. Thus for this spacecraft each BCR is adjusted to an upper voltage limit of 25.5 V, which corresponds to the end charge voltage of each battery string.

The three independent power strings are combined before the Main Switch of the PCDU, but are secured with diodes to prevent the current flow of one string into another. In case a battery string or solar panel is broken or short-circuited, the energy of the other two strings can be used to operate the spacecraft.

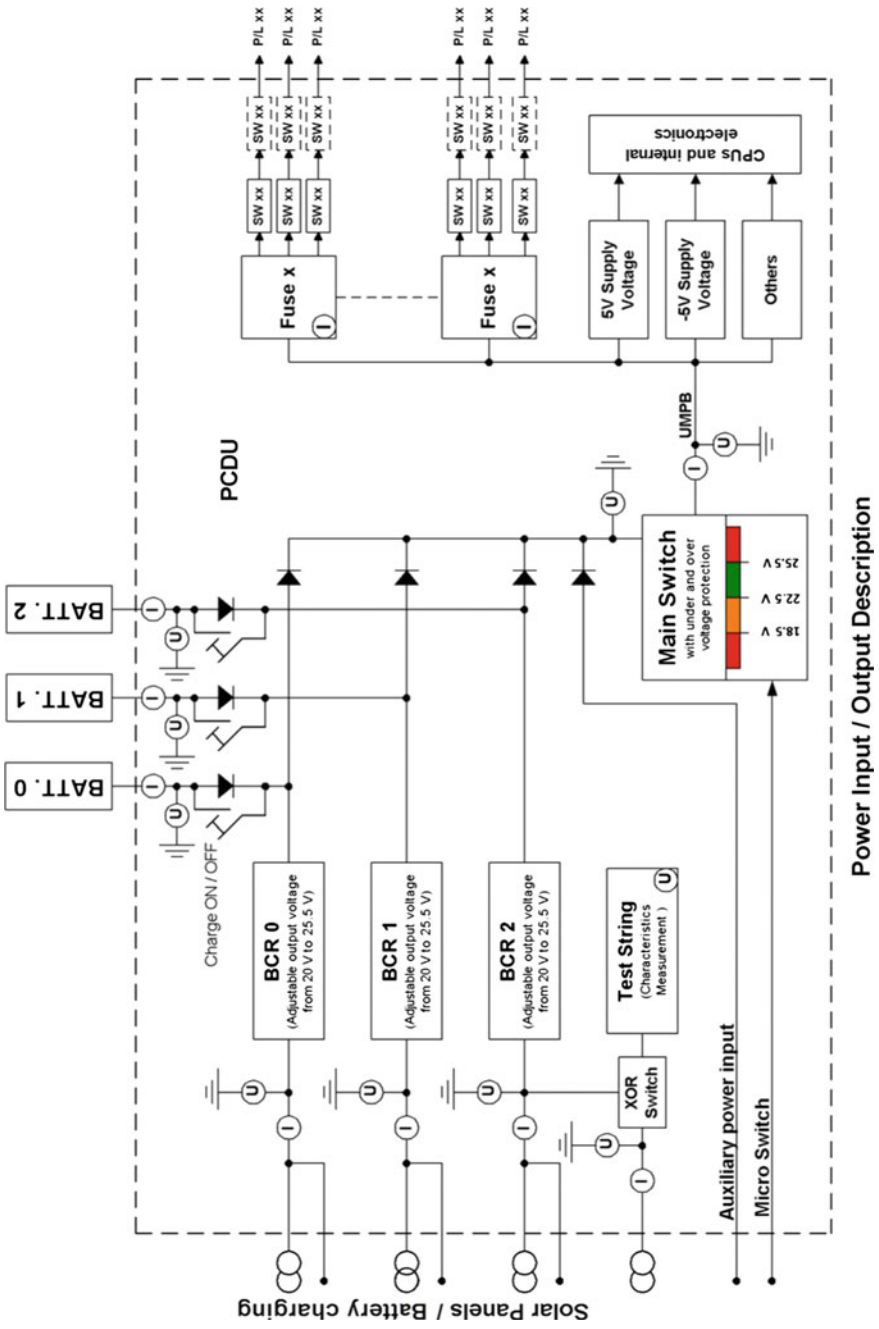


Fig. 5.9 Circuitry and power connections of the PCDU. © IRS, University of Stuttgart

String 0 and 1 represent the energy paths of the deployable solar panels, whereas string 2 represents the path of the center panel and the solar test string. The solar test string is used for the generation of electrical energy by default together with string 2.

The distribution of power to the consumer loads is controlled by the application of a fuse and switch system. The PCDU deactivates the power supply by the respective Latching Current Limiters (LCLs), as soon as an over-current is measured. To minimize volume, mass and cost of the PCDU some power outputs are connected to a shared fuse. However, critical on-board components such as the OBC boards and the TC receivers are implemented as single loads on dedicated separate fuses.

For reliability reasons and due to the combined allocation of multiple loads to one fuse, additional switches are used to regulate power supply of single loads. High-power consuming components are equipped with two switches in series in order to protect the satellite bus. If one switch should stick in open closed position, the second serially connected switch can be opened to deactivate the respective component, if necessary. The LCL fuses can be reactivated after the event of an over-current and the connected consumers are not lost for the mission. A complete list of the component affiliations to fuses and switches can be found in Table 17.10.

The battery heaters with their bi-stable relays already were mentioned in Sect. 5.4.1. In each of the heater circuits—one nominal and one redundant—there are two switches: bi-stable relays in the PCDU and thermostat switches in the battery system. The bi-stable relays can be activated and deactivated by command to the PCDU, the thermostat switches in the battery close the circuit when the battery temperature drops below approximately 0 °C. Note that this limit temperature was selected because below 0 °C the battery capacity decreases significantly and its voltage characteristics change. However, it does not yet constitute a critical temperature for the battery cells. Their lower operating temperature limit is −30 °C as provided by the manufacturer [58].

During nominal operations, the bi-stable relays are activated at all times so that the battery is heated whenever its temperature drops below 0 °C. Due to the relatively low power consumption of the heaters (up to 8 W), this state is not changed as long as the OBSW is active, i.e. there are no FDIR actions which deactivate the bi-stable relays in the PCDU directly. However, the bi-stable relays are deactivated as soon as the PCDU is commanded to shut down mode in case of a severe battery under-voltage or low SoC, which constitutes a severe FDIR case (see Sect. 10.3.1).

Figure 5.10 shows the connections between the PCDU and a single battery string. Charge and discharge of the battery is managed by the power interface. By default, the switch is closed in order to allow charging of the battery string. If the charge process shall be interrupted the switch can be opened. The energy path with the diode still allows energy extraction from the battery.

Since the PCDU only monitors the voltage level of a complete battery string, single cells would not be protected against overcharging by this simple approach. If the voltages of single, serially connected cells diverge too much, single cells could

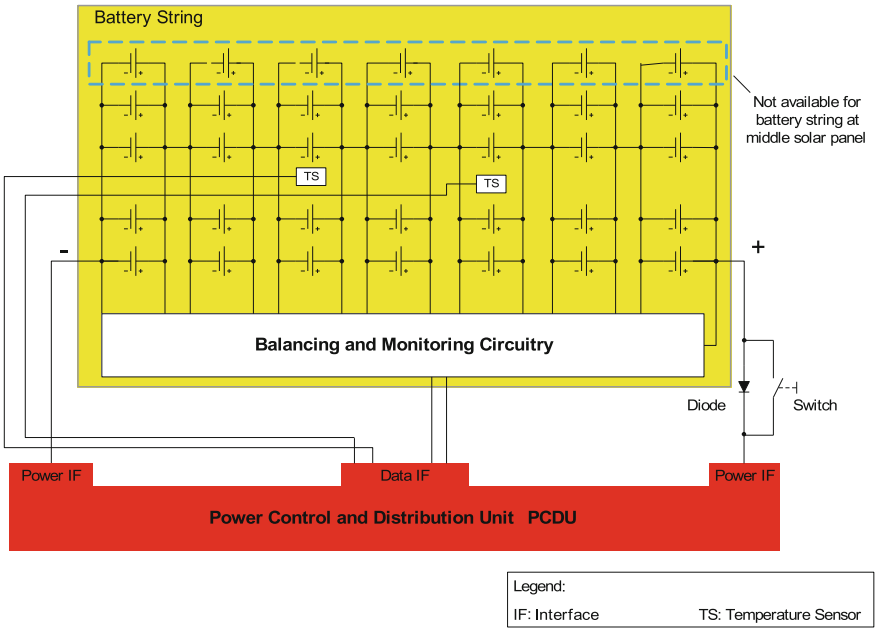


Fig. 5.10 Connection of the PCDU and a battery string. © IRS, University of Stuttgart

be overcharged before the combined charge limit of 25.2 V is reached. Therefore to prevent overcharging of an individual cell, battery charge control boards are implemented inside the battery which monitors the respective cell voltage. The PCDU provides corresponding interfaces for dedicated the signals from the monitoring circuitry inside the battery. As soon as the PCDU receives the interrupt signal, battery charging is stopped by opening the respective switch in the energy path for a specified time. In case of a fault event there exists the possibility to command the PCDU to ignore the interrupt signal.

Each battery string is equipped with two temperature sensors for thermal monitoring. In case the temperature limits for a stable energy output are violated this is detected by the *TCS_Controller*. In such case FDIR disables the string charging to prevent long-term damaging of the cells—see Sect. 10.3.3.3.

The battery's State-of-Charge (SoC), is usually calculated by the OBSW and additionally in the PCDU for those cases where the OBC is not operational—see Sects. 5.4.2 and 10.3.1. The available under-voltage protection feature preserves the batteries, if the voltage level falls below 19 V (HW protection in PCDU) and 20 V (SW protection in OBSW).

The measurement point is located at the main switch of the PCDU. A staggered scheme of undervoltage handling is implemented in the platform FDIR. For more details see Sect. 10.3.1. The final stage is the one where the PCDU and hence the complete satellite is deactivated. In this extreme under-voltage case the satellite system is set into the PCDU Shutdown Mode, where the available power charges

only the batteries and survival is assured with a sufficient margin before the minimum voltage level is reached. In order to avoid a meta-stable state, in which the satellite turns on and off, the PCDU reactivation threshold is specified to a higher voltage level of 21.5 V.

A further implication of the combined loads at one fuse is a current monitoring feature by S/W through a Current State Table. The Current State Table contains reference values of the permitted current levels for all on-board components. Additionally, the PCDU records and monitors which components are active/powered at each fuse with the help of the configuration list. As soon as an LCL determines a current flow exceeding the referenced value in the Current State Table, the PCDU deactivates the respective LCL to avoid corruption of the connected components. This monitoring functionality is performed with a repetition rate of higher than 20 Hz to increase the protection potential.

5.5.2 PCDU Design Overview

The PCDU (see Fig. 5.11 and Table 5.4) features two SH7045 32-bit high-speed, single-chip microcontrollers [56] from RENESAS Electronics for the operational tasks. Those controllers have already been successfully applied for multiple space missions by Vectronic Aerospace.

The steady state power consumption of the unit lies at approximately 8 W. By design, heat emitting parts like fuses, switches or the CPUs are placed by Vectronic on PCBs near the baseplate, which is connected to the structure for thermal conductance reasons. The remaining surface sections are anodized in a matt black color

Fig. 5.11 Engineering model of the PCDU (from [47]). © Vectronic Aerospace GmbH

Table 5.4 PCDU characteristics

Parameter	Min	Typical	Max	Unit
Supply current at 25 V Auxiliary Supply (standby)	90	100	120	mA
Power consumption at 25 V Auxiliary Supply (standby)		2.5		W
Bus voltage	18.5	–	25.5	V
Reset recovery time	–	10	20	sec

to increase the thermal balancing by radiation. A PCB internal heating for the CPU PCBs facilitates a fast warm-up to −20 °C in order to prevent damaging of electronic parts due to thermal stress due to high temperature gradients. Moreover, the PCDU is qualified to a lower limit of −40 °C for operational use to increase the availability of the PCDU and thus satellite system reliability (see also Sect. 8.2). The thermal conditions are monitored by 5 internal temperature sensors inside the PCDU.

According to FLP platform design regulations the PCDU is designed single failure tolerant. This means that a specific functionality is covered by a redundant unit or functional path in case the nominal unit fails.

The PCDU is equipped with a number of interfaces for connecting digital and analog equipment plus the serial interconnection to the OBC. Furthermore, the PCDU provides electrical interfaces for power generation, storage and distribution. Besides that interfaces exist for satellite system monitoring and for all tasks of OBC monitoring and reconfiguration in the frame of the overall CDPI architecture. The listing provided below comprises all interfaces that are implemented for FLP use.

- Interface to solar arrays
- Interface to batteries
- Power supply for all components
- Battery overcharge protection
- Interface to the launcher
- RS422 communication interfaces to the OBC Processor-Board through I/O-Board
- RS422 communication interface for High Priority Commands
- Interface for temperature sensors
- Interface for sun sensors
- Interface for solar panel deployment detection

The PCDU is commanded by so-called PCDU Common Commands for control of the fuses, switches and PCDU modes and to request sensor data as well as PCDU status. The detailed command format between OBC and PCDU is described in [47], but it is transparent for the ground operator. The PCDU is commanded via TC by standard commands for fuse switching, power switch activation etc. as defined in the Mission Information Base (MIB).

5.5.3 PCDU Boot-up Sequence and PCDU Modes

The FLP platform is designed for compatibility with launch supplier requirements for secondary payload satellites. For these usually the requirement exists that no electronic units in the satellite are permitted to be active before launcher separation to avoid electromagnetic interference with the launcher and/or its main payload. Thus the FLP platform based satellites are designed to be launched with all avionics off—including OBC and PCDU.

The PCDU is in Launch/Shutdown mode before separation of the satellite from the launcher (see Fig. 5.12 and [57]). The only active circuitry is the analog detection circuit for launcher separation inside the PCDU plus the battery charge control boards inside the battery.

After separation of the satellite from the launcher—or after recovery from spacecraft overall power failure—the PCDU is the first unit on board to boot. A safe and reliable step-by-step boot-up sequence of the PCDU and thus of the entire satellite system is implemented to facilitate the completion of the first stable satellite mode, the System Safe Mode.

The boot-up procedure includes specific prerequisites before the OBC boards are powered by the PCDU and take over control of the satellite. Thereby, following actions are performed to prevent the damaging of critical satellite units:

1. PCDU internal heaters warm up the unit to its operational temperature limit
2. The power level of the battery is checked to complete the entire PCDU boot-up procedure up to initiate the equipment power-up steps for the satellite Safe Mode—see Fig. 5.12.
3. Check of the temperature level of the OBC unit and the TT&C transceivers. If temperature is below the operational limit, the PCDU activates the power switches for the redundant internal heater design of both units. These heaters are regulated via PCDU internal thermistors to facilitate the heating up to the specified operating temperature.

   Alternatively, a timer condition is implemented which is set according to the results of thermal simulations.

   As soon as the timer condition is met, the PCDU continues the boot-up process.
4. The last step concludes the boot-up procedure to the PCDU MiniOps Mode.

From this point onwards, the OBC can command every other defined operations mode of the satellite. The detailed co-operation between PCDU and then booted OBC and OBSW during the Launch and Early Orbit Phase (LEOP) is described in more detail later in Sect. 4.4. Since the initial spacecraft power-up sequence does not involve any spacecraft commanding from ground it will not be detailed here any further.

It is possible for the system to re-enter the Shutdown mode when the battery SoC drops severely (see Sect. 10.3.1). There exist two PCDU Recovery Modes that

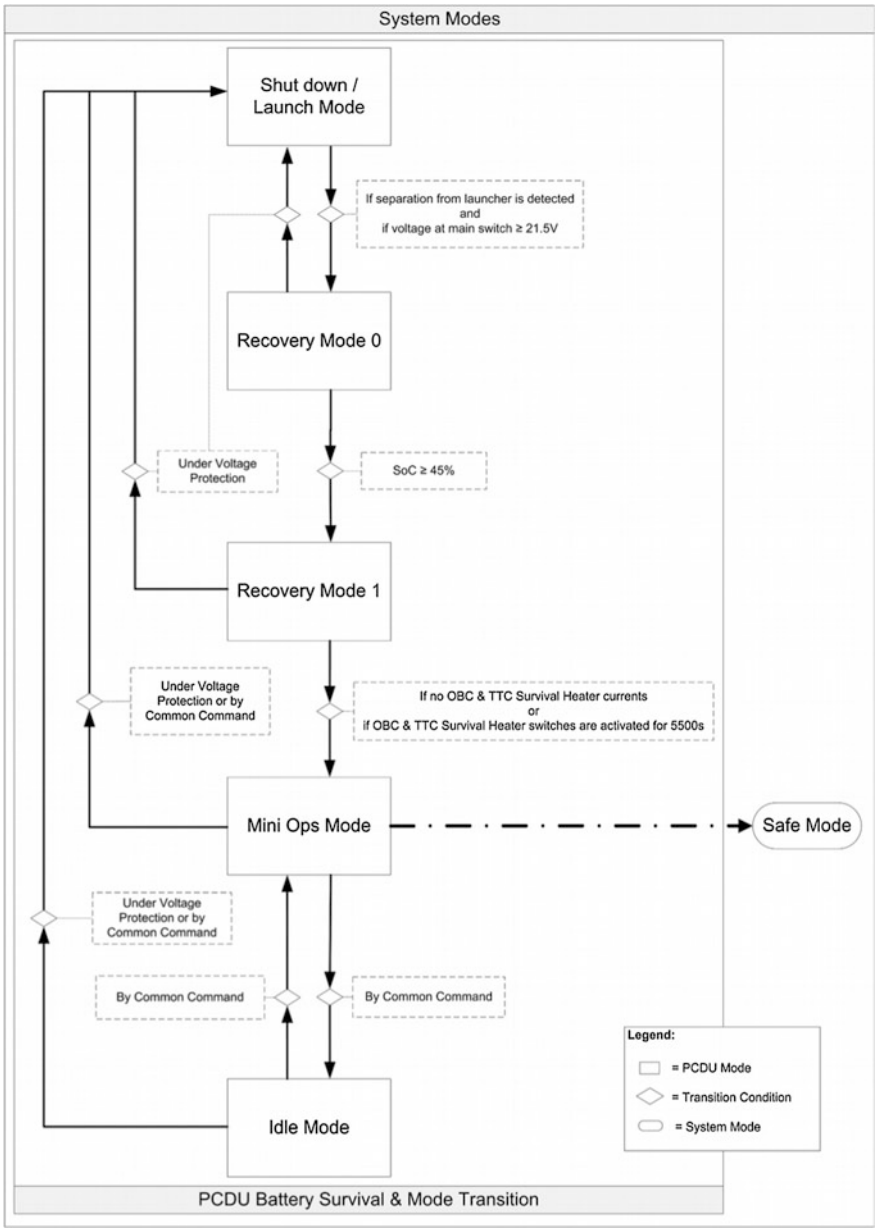


Fig. 5.12 PCDU modes and transitions. © Vectronic Aerospace GmbH

prepare the satellite for the system boot. Main activities are heating of PCDU and OBC/TTC to their operational temperature range. These steps will be run through after launch or emergency shutdown.

The PCDU MiniOps Mode is a very minimalistic mode and does not even allow the *PSS_Controller* of the OBSW to command any fuse switching. Therefore when the PCDU has booted OBC and TTC subsystem at system startup, and the OBSW is commanding the spacecraft into Safe Mode, it also commands the PCDU up to its Idle Mode. In PCDU Idle Mode, all other spacecraft components can be powered via common command. The entire PCDU mode transition process is depicted in Fig. 5.12.

5.5.4 Specific PCDU Functions in the CDPI Architecture

5.5.4.1 Analog Data Handling Concept

One of the special characteristics which exceeds the scope of duties of a standard PCDU is the different approach for the on-board data reception. Usually, the collection of all data is conducted by a separate unit in an industrial satellite, sometimes referred to as Remote Interface Unit [140].

For the FLP, digital and analog data interfaces are separated in the command chain. Most of the digital IFs to the satellite units are realized by the OBC I/O-Boards. Making use of synergies, the PCDU contains all analog on-board interfaces. Since the PCDU contains analog/digital converters (ADCs), for the measurement of voltages and currents anyway, this step was a reasonable decision.

According to this interface control concept, the PCDU collects all analog sensor data on board the satellite. Some digital sensor data which is required for the PCDU tasks is collected as well. The following sensor data is collected by the PCDU (Table 5.5):

Table 5.5 Overview of sensor data collected by PCDU

Data Information	Quantity of functional connections	Data Type
Temperature sensors (resistance)	32	analog
Sun sensors (current)	16	analog
Solar panel and battery condition (voltage/current)	3 (for solar panels)/3 (for batteries)	analog
Characterization of an experimental test string (voltage/current)	1	analog
Reed sensors for deployment status of two solar wing panels	4	digital
Separation detection from upper stage of the launcher	1	digital
Input for monitoring signals for battery overcharge protection	3	digital

This sensor data is not processed inside the PCDU. Whereas the analog sensor data is converted to digital data by the ADCs, all sensor data is forwarded to the

OBC. The handling is conducted in the OBC, utilizing its processing power, where the relevant data is also distributed to the respective subsystem control module. For example the temperature sensor data are handled by the *TCS_Controller* in the OBSW, the Sun Sensor data by the *ACS_Controller*.

5.5.4.2 PCDU Functions for OBC and S/C FDIR

As already mentioned in Sect. 1.8 the PCDU internal Combined-Controller provides reconfiguration functions for both PCDU internal board reconfigurations and for OBC board reconfigurations. Details on these functions are provided in the FDIR Sect. 10.2.2 of this book.

Furthermore the PCDU can be commanded directly via High Priority Commands (HPCs) which bypass the OBSW in case of emergency situations where OBC and OBSW are failed. The HPC based S/C recovery functions are described in Sect. 10.2.4.

5.5.5 Diverse PCDU Functions

The PCDU furthermore features a number of functionalities which partly are standard for PCDUs, partly are resulting from the PCDU's role in the overall CDPI architecture and some also are implemented as specifics of the FLP target satellite. These functions are explained very briefly here only. For further information please refer to [57, 123].

5.5.5.1 Deployment Control for De-Orbiting Mechanism

For faster deorbiting an FLP based satellite can be equipped with a De-Orbiting-Mechanism (DOM) which is a deployable Kapton sail to increase the aerodrag. For control of the DOM release the PCDU firmware contains an internal DOM deployment counter which is starting to count down one tick per second as soon as the PCDU is booted to Mini-Ops mode or higher. For cyclic timer reset during nominal mission and for the mission end of life scenario description refer to Sect. 11.9.

5.5.5.2 Launcher Separation Detection

Due to the PCDU taking over in the CDPI architecture some functions of a classical OBC Remote Interface Unit (RIU) it features an arming switch for detection of spacecraft separation from the launcher by opening the according circuits. This prerequisite together with a sufficient level of solar array input power is required to startup the PCDU operations. More details on the process during LEOP sequence are provided in Sect. 11.6.

5.5.5.3 Control and Monitoring of Solar Panel Deployment

The control and the monitoring of the solar panels deployment procedure by the PCDU is based on both implemented Deployment Timers (timer 0 and timer 1) and on the activation flag. The control is performed if the activation flag is enabled (default).

After the timer 0 becomes active (time after launcher separation) the PCDU activates the fuses and the switches of the heaters for the deployment mechanism and checks its status. The default value for the time delay between launcher separation and start of deployment is 300 s. The value is programmable on ground before launch to comply to the requirements of the launch supplier.

As soon as the deployment mechanism signals a successful deployment of the solar panels, the PCDU switches off the heaters and disables the activation flag. If the deployment mechanism does not signal a successful deployment of the solar panels and the timeout value of 180 s is exceeded, the PCDU will switch off the heaters without disabling the deployment process. A total of 5 attempts will be made (with a wait interval in between) to release the solar panels by switching the deployment device heaters. After 5 unsuccessful attempts the autosequence is finally deactivated in order to save power, and FDIR from ground has to take over. More details on the process during LEOP sequence are provided in Sect. 11.6.

5.5.5.4 History Log Function

The PCDU software includes a history log functionality for commands, Events and configuration of working components. The history log functionality is introduced in order to establish a means to trace activities inside the unit in case of operational issues. Each of the recorded parameters are identified by a dedicated ID and a time stamp.

5.5.5.5 Time Synchronization Between Internal Controllers

The PCDU features a time synchronization mechanism between the current operating PCDU controller and the hot redundant one. The synchronization is performed every 5 min.

5.5.5.6 Overvoltage Protection

In addition to the under voltage protection feature for the batteries, the PCDU features an overvoltage protection for itself. The PCDU is switched off automatically via its main switch as soon as the bus voltage exceeds 28.5 V. This case may apply during tests on ground, when the PCDU is powered through the auxiliary power input but is unlikely during flight.

5.5.5.7 Measurement of Test-String Characteristics

The PCDU of the first FLP mission "Flying Laptop" features a measurement circuitry based on an analog/digital converter (ADC) for recording the characteristic line of the test string of the satellite's center solar array. The measurement is initiated by command. The PCDU sets the current flow through a shunt resistor and records the values of the current and of the associated voltage.

5.6 Power Subsystem Control

The power balancing and battery charge/discharge control in the FLP power subsystem (PSS) is almost completely automated by the PCDU. For communication with the PCDU and for managing the PCDU the power subsystem inside the OBSW is equipped with a PCDU handler object.

Notwithstanding the autonomy in electric power control through the PCDU, it is required to perform certain enhanced surveillance and control activities from within the OBSW to maintain a consistent system overview. The OBSW therefore features a *PSS_Controller* application (see Fig. 5.13), which is composed of a number of system objects as shown in Fig. 5.14.

The controller manages a "Battery String" object for each physical string, which is responsible for battery power and state of charge calculation. Each string has a "String Power Sensor" object, which checks sanity of the voltage and current sensors of the respective battery string in the PCDU.

The controller also has a "Solar Panel" object for each solar panel (and the test string), which monitors the panel's voltage and current sensors and calculates the power being generated by each panel.

There exists also a "Fuse" object for each fuse, which first checks the health of the fuse's current sensor, calculates a power value and monitors this power value with respect to the powered devices. The device's power information is found in the so called "Device Power Component", which represents the information related to power management of a fuse connected device. Depending on the number of devices attached to one fuse, each Fuse object has one to four Device Power Components.

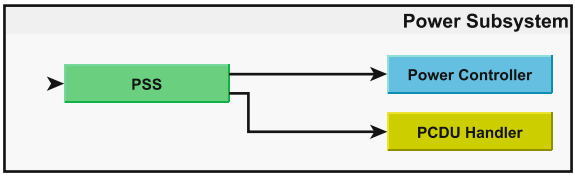


Fig. 5.13 Power-subsystem control elements in OBSW. © IRS, University of Stuttgart

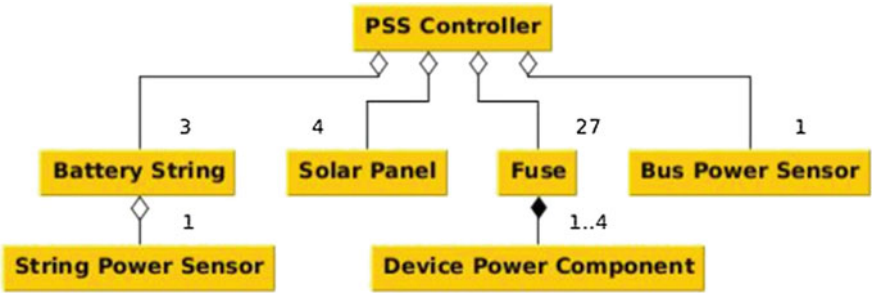


Fig. 5.14 PSS class diagram. © IRS, University of Stuttgart

Finally, the controller has a "Bus Power Sensor" object, which monitors the bus current and voltage sensor sanity and calculates the bus power.

The *PSS_Controller* object itself is responsible for value aggregation, such as for the calculation of the combined state of charge, or the power running through all fuses, as well as higher-level monitoring and control activities, which are described below in more detail.

Figure 5.15 shows all calculations and their localization in the *PSS_Controller*. In addition, it shows the d *datapool* in- and output as well as the dependencies of the calculations. Dotted arrows indicate that the originating calculation needs to be performed for the target one to be calculated, even though the result is not used. A green arrow indicates, that there is an alternative strategy in case the value is not valid.

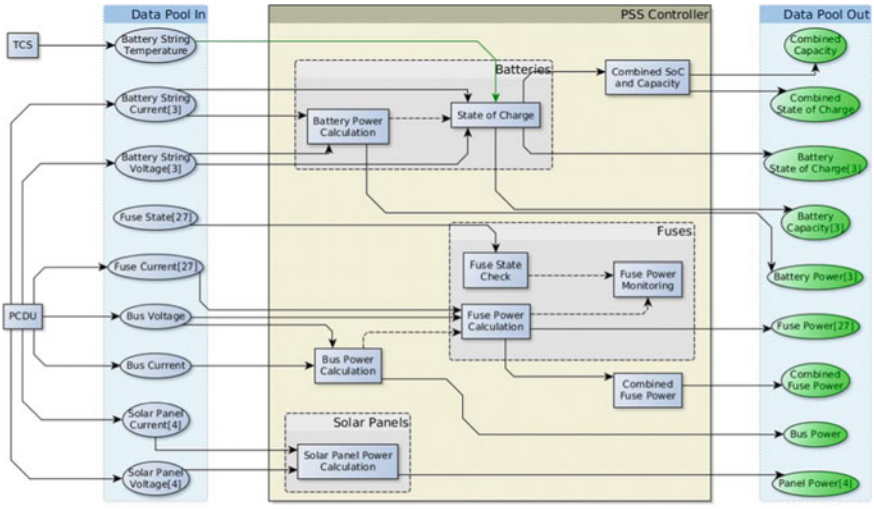


Fig. 5.15 PSS calculations and data pool dependencies. © IRS, University of Stuttgart

5.6.1 PCDU Device Handler

The PCDU device handler is used to command the PCDU through the OBSW. It offers the following modes (Table 5.6):

Table 5.6 PCDU device handler modes

Handler	Mode	M#	Submode	SM#	Description
PCDU	Off	0	Default	0	No PCDU Commanding. DO NOT USE!
	On	1	Default	0	No automatic PCDU Commanding, direct commanding possible. DO NOT USE!
	Normal	2	MiniOps	0	PCDU MiniOps Mode
			Idle	1	PCDU Idle Mode
	Raw	3	Default	0	Raw Mode unchanged from before

- During normal flight operation of the satellite the PCDU will be in mode Normal, submode Idle.
- In MiniOps Mode the PCDU does not allow switching fuse statuses, therefore it is not used for flight operations.
- Commanding the PCDU down to On mode will cut the communication between OBC and PCDU as the PCDU will not listen anymore to OBC TM requests. This will initiate a severe OBC reconfiguration sequence as described in Sect. 10.2.3.
- Commanding the PCDU to Off mode will shutdown the PCDU and cut the power for all satellite equipment. Presuming the power-bus shows a sufficient voltage level the PCDU will reboot automatically. Nevertheless this measure is strictly reserved for severe FDIR cases—see Sect. 10.3.

5.6.2 PSS Controller

The main tasks of the *PSS_Controller* are to monitor the PCDU's behavior, to check the system's power consumption, and to calculate the battery state of charge.

- The monitoring actions are:
  - Monitor the solar panels and calculate the generated power (see Sect. 5.6.2.7).
  - Check solar panel deployment after separation (see Sect. 5.6.2.7).

  - Monitor the battery strings and calculate a battery power, their independent and a combined capacity and state of charge (see Sect. 5.6.2.6).
  - Check the power flowing through each fuse and compare with expected power of devices (see Sect. 5.6.2.8).
  - Check the sanity of the bus voltage and current sensors (see Sect. 5.6.2.9).
- In addition, some control activities are performed.
  - Update the battery capacity values estimated in the PCDU regularly.
  - Control activation and deactivation of BCRs (see).
- As an extra activity, the controller should bring the system into a well-defined state after OBC boot.

To manage all these activities, the controller features a number of modes, functions, *private pool* elements and monitors, which are explained below.

The controller is executed at a constant frequency. In each step all control activities above are executed and all output variables calculated, both for the controller itself and all sub-objects. The power subsystem cycle rate is provided in Sect. 3.4.2.

5.6.2.1 PSS Controller Modes

The power subsystem applies the general system-wide mode concept (i.e. it can be commanded with default mode commands). The *PSS_Controller* directly mirrors the PSS modes and performs the required operational tasks as listed in Table 5.7.

Table 5.7 *PSS_Controller* mode table

Ctrler	Mode	M#	Submode	SM#	Description
PSS	Off	0	Default	0	Off
	On	1	Default	0	On, but no capacity updates to the PCDU
	Normal	2	Default	0	On, with capacity updates to the PCDU and health state checks of the BCRs (if all are Off, it turns all On again)
			Only SoC Updates	1	On, with capacity updates to the PCDU, no health state checks of the BCRs (if all are Off, it turns all On again)
			Only BCR state check	2	On, with health state checks of the BCRs (if all are Off, it turns all On again), no capacity updates to the PCDU
	Boot	20	Default	0	Performs boot procedure: relays On, fuses On, switches Off, get initial capacity values from PCDU, thereafter Off

- There exists an Off mode, where the controller does not perform any monitoring and control activity,
- an On mode, where all monitoring calculations are performed, but no control activities,
- and a Normal mode, where controlling is scheduled as stated above. To turn off specific control activities, the Normal mode is defined with three submodes. Table 5.7 lists all available modes.

As mentioned in the PCDU subsection, the controller will automatically switch the PCDU to its Idle Mode system boot and will turn on all healthy fuses.

If the controller is commanded to Off mode, it sets all *datapool* variables to invalid and all monitors to unchecked.

If the controller was in Off state, it polls the capacity values from the PCDU as initialization values for the state of charge algorithm during a transition to On or to Normal.

5.6.2.2 Boot Mode Procedure

During boot, the *PSS_Controller* shall bring all fuses into a well-defined state for the OBSW to take over system control from the PCDU. Figure 5.16 shows a flow chart of the boot procedure.

For these activities, it is necessary to command the PCDU, which might fail. In this case, the procedure is stopped and either a PSS_TIMEOUT event or a PSS_COMMANDING_PCDU_FAILED event (see Sect. 10.3) is triggered. The former indicates that the PCDU did not respond at all, the latter that either sending the command to the PCDU failed, or the PCDU did not (correctly) execute the command. The controller will switch to the desired mode anyway, but the procedure might not be executed entirely (meaning there might still be some fuses remaining in Off status).

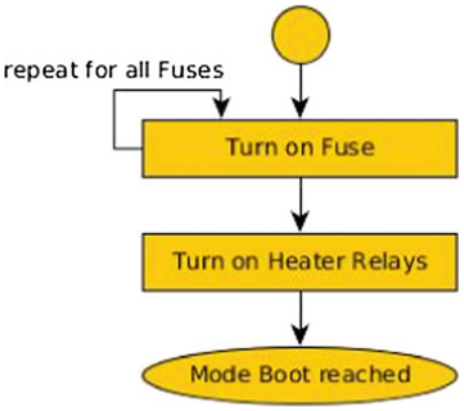


Fig. 5.16 Flow chart of boot procedure. © IRS, University of Stuttgart

5.6.2.3 Controller Functions

The *PSS_Controller* provides functions (Service 8) to simplify handling different battery SoC calculation parameters.

- One command is to simplify setting all maximum battery string capacities at once, which needs to be executed regularly because of cell degradation.
- Furthermore there is one command to update the estimation values in the PCDU and
- one to set all calculation parameters for one string (see Table 5.10).

For details on the algorithms see Sect. 5.6.2.6.

5.6.2.4 Controller Memory

There are some *private pool* parameters to adjust the following values:

- The PCDU update interval: The interval in which the capacity values in the PCDU are updated by the controller.
- The maximum delta capacity: The capacity difference between the combined capacity calculated by the PCDU and that of the PSS above which a CAP_DELTA_VIOLATION event is thrown.
- The panel deployment flag: This flag can be used to manually overwrite the deployment check (see Sect. 5.6.2.7).

5.6.2.5 Controller Monitors

The built-in monitors of the controller check the calculated battery State-of-Charge. The check is performed with a group of three standard limit checks, where the high limit is set to an unrealistic value. The default values are collected in Table 5.10. The three checks throw SOC_WARNING, SOC_CRITICAL and SOC_SHUTDOWN event (see Sect. 10.3) in case the state of charge drops below given limits. SOC_HIGH indicates a calculated state of charge higher than 100 % and should never occur in a bug-free software with correctly set limits.

5.6.2.6 Battery String

Each battery string is represented by a “Battery String” object, the activities of which are shown in Fig. 5.17. The blue squares depict calculations that are performed, the arrows show the necessary *datapool* input and the produced output. Orange ellipses represent standard limit monitors.

The string object checks all input values and calculates the power consumed or distributed by the battery. From that given input and the capacity value from the previous computing step, it also calculates a new capacity and SoC as described

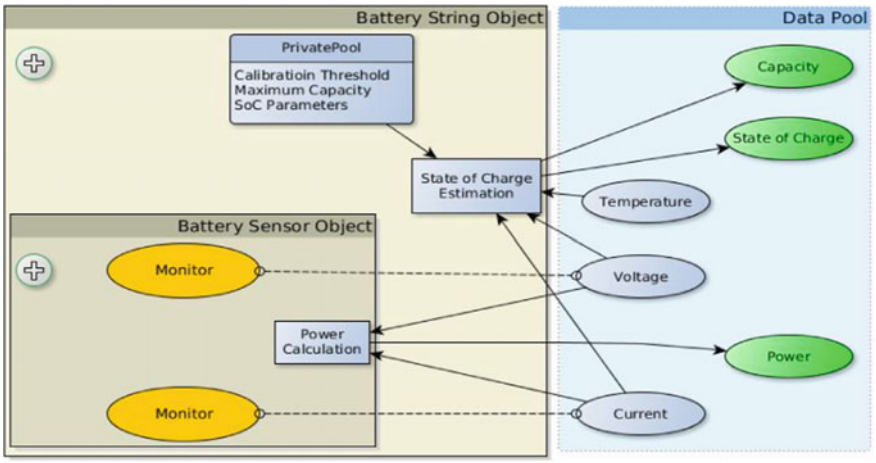


Fig. 5.17 Battery string object. © IRS, University of Stuttgart

further below. The adjustable parameters for this calculation are accessible in the string object's p*rivate pool* memory.

Checking the input values includes monitoring of the string's current and voltage sensor, which is done by a dedicated object.

If the current or voltage input values are out of range or the sensors are marked as faulty, no calculation is performed. If the temperature is invalid, the power value is still calculated and the controller switches to a simplified state of charge estimation.

State of Charge Estimation Algorithm

To avoid drifting of the calculated capacity value, the algorithm has a calibration function, which initially sets the string capacity to its maximum value. To trigger calibration, a voltage, State-of-Charge and current and threshold is used. Assuming the battery is almost full, the voltage as well as the SoC should be above a certain level and the current should be positive but small—see Sects. 5.4.2 and 5.4.3.

If all these conditions are fulfilled, the capacity is set to the calibrated maximum capacity and a SOC_CALIBRATION Event is thrown.

String Memory

The string's parameters tune the State-of-Charge calculation algorithm which was described previously. Settable values are

- the maximum capacity of a string
- the power consumption of the BCC boards
- the thresholds when calibration takes place
- the temperature value above which the state of charge estimation is temperature-corrected
- factors to estimate the temperature losses.

Default values are found in Sect. 5.4.3.

Combined Capacity and State-of-Charge
After calculation of the individual state of charge and capacity values of each string, the *PSS_Controller* collects all valid values and merges them to a combined value. The combined State-of-Charge is the mean value of all valid state of charge values, the capacity the sum. If no values are valid, the combined values are set to invalid.

The calculated capacity is compared with the capacity provided by the PCDU. If the values differ more than the defined threshold (see Sect. 5.6.2.4), the CAP_DELTA_VIOLATION Event (see Sect. 10.3) is thrown.

As there is a certain drift assumed between PCDU and PSS calculation, the capacities in the PCDU are updated at regular, settable intervals by the *PSS_Controller* (see Sect. 5.6.2.4). In normal conditions, the calculated capacities should drift less than 1 % per orbit.

String Health
The battery string itself has a health state which may be "HEALTHY" or "FAULTY". It indicates if the battery itself is defective. The health state of a battery string is monitored by the *PSS_Controller*. The controller turns off the BCR for that battery string to avoid damaging or heating up the system. If the health state becomes "HEALTHY" again, the BCR is re-activated.

As turning off all battery strings is very dangerous for the satellite, the controller will overwrite all health states in case where all strings are marked as "FAULTY" and will try to turn them on again.

The *PSS_Controller* does not stop capacity and state of charge estimation for a string tagged faulty. The algorithm assumes the current and voltage values still being valid and therefore further calculations are possible.

String Power Sensor
Each string has a dedicated sensor object to monitor the health of the battery string's current and voltage sensor and to calculate the power value.

The sensor object checks the current and voltage values for reliability, i.e. it checks if they have a physically reasonable value. If the values are valid and reasonable, a power value is derived.

The sensor can be tagged "FAULTY", which stops the monitor checks and power calculation. The power output value becomes invalid, the monitors are set to "unchecked". In addition, the string's current and SoC calculation stops and these values are invalidated, as the required current value is assumed to be faulty.

The sensor maintains two limit monitors, one for current and one for voltage. In case of a limit violation, the monitors throw appropriate Events (e.g. BATTERY_CURRENT_LOW—see Sect. 10.3).

5.6.2.7 Solar Panel

The "Solar Panel" object monitors one solar panel or the test string and calculates the power value of the panel. Its calculation (blue squares) and monitoring (orange ellipses) activities are shown in Fig. 5.18.

The object checks the current and voltage values for reliability, i.e. it checks if they have a physically reasonable value. If the values are valid and reasonable, a power value is calculated. If a limit is violated, the panel object throws an appropriate event e.g. PANEL_VOLTAGE_HIGH (see Sect. 10.3).

A panel can be tagged "FAULTY", which stops the monitor checks and power calculation. The power output value becomes invalid, the monitors are set to "unchecked". It is not possible to set the power sensor health separately from the panel itself, as the effect is the same (nothing is switched, power calculation is not performed).

The *PSS_Controller* features an autonomous check for solar panel deployment during boot. This is done by reading the reed sensor values periodically (every 30 s). If at least one sensor does not show deployment, a PANEL_NOT_DEPLOYED Event is thrown (see Sect. 10.3). As soon as deployment is detected, the controller throws a PANEL_DEPLOYED Event immediately, stops sending Events and sets the "panels deployed" flag to '1'.

If for some reason the panel was not deployed and the Event needs to be turned off, it is possible to overwrite the "PANELS DEPLOYED" flag in the controller with a memory load command (see Sects. 5.6.2.4 and 10.3).

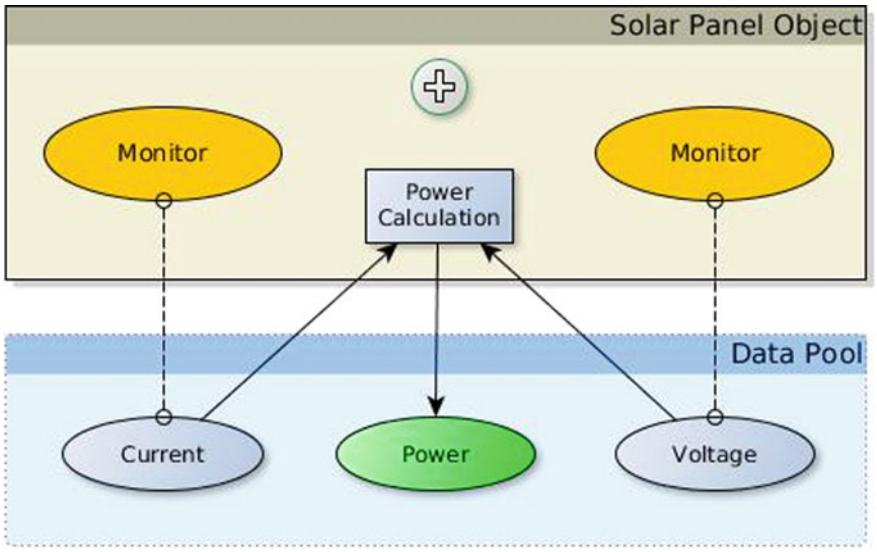


Fig. 5.18 Solar panel object. © IRS, University of Stuttgart

5.6.2.8 Fuse

The "Fuse" object provides an additional check for the power consumption of components and checks state and health of a fuse (see Fig. 5.19). For power checking, it calculates the current power limits (low/high) from the min/max values given by each component attached to the fuse and reports violations of these limits.

In addition, the object checks whether the fuse went off during operation by monitoring the fuse state reported by the PCDU. On detection, a FUSE_WENT_OFF Event is thrown (see Sect. 10.3).

It also features a limit monitor to check the reliability of the current sensor values. The limits have a low limit of 0 A, the upper limit is chosen such that it is above the expected total maximum current of all components of that fuse. A violation indicates either a broken sensor or a very high over current. In this case, a FUSE_CURRENT_HIGH event (see Sect. 10.3) is thrown and no further calculations and checks are performed.

The devices are represented by "Device Power Components" which maintain the limit values for each device. These limit values are settable by a dedicated PUS Service 12 command. Each fuse has a health state, which indicates whether the fuse or its sensor is broken. In such a case, no further checks are performed and the fuse

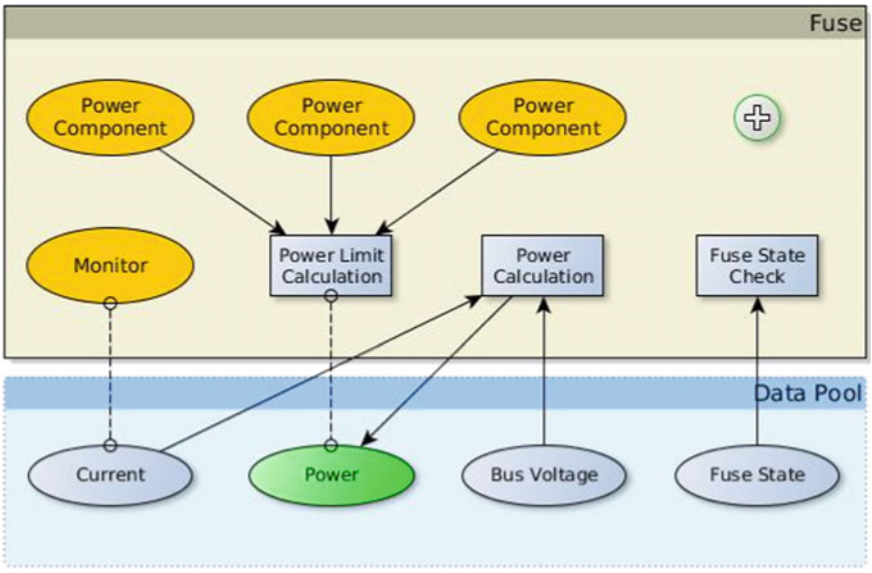


Fig. 5.19 Fuse object. © IRS, University of Stuttgart

is not switched on automatically during boot transition. Power calculation is also not performed anymore. This is also the case if the fuse is turned on manually.

Power Components and Power Monitoring
Each component with at least one switch is monitored by the *PSS_Controller*. As there are up to four components attached to one fuse but only one current sensor per fuse, power monitoring must take into account switch states of the components.

Every component has two limit values which define the upper and lower power limit for component when being active. Components belong to a fuse object, which uses the switch state and the power range to calculate the permitted power range (i.e. upper and lower limit) for the entire fuse. Components might belong to multiple fuses (this is currently only the case for the Star Trackers).

In case where the power is out of this range, all components of that fuse with switches On throw a POWER_ABOVE_HIGH_LIMIT respectively a POWER_BELOW_LOW_LIMIT event (see Sect. 10.3). This is because it is not possible to distinguish which component violated the limit.

Total Fuse Power
To simplify checking of the power budget, the *PSS_Controller* calculates a combined fuse power value, which is the sum of the power values of all fuses. In case where a fuse is off, faulty or the current is above its limit, the power amount contributing to the total power is 0.

5.6.2.9 Bus Power Sensor

The bus voltage and the bus current sensor are critical for evaluation of device power consumption. Its health is monitored by a dedicated "Bus Power Sensor" object. The sensor has a health state which can be set to "HEALTHY" or "FAULTY".

The object checks the current and voltage values for reliability, i.e. it checks if they have a physically reasonable value. If the values are valid and reasonable, a power value is calculated.

Apart from turning off the bus power calculation (power value becomes invalid, monitors unchecked), marking the sensor as "FAULTY" leads to a complete deactivation of fuse monitoring, as there is no reference voltage for power calculation anymore.

5.6.3 Power Subsystem Mode Transitions and Telemetry

The Power-Subsystem mode commanding—according to the Handler and Controller modes defined in Tables 5.6 and 5.7—is performed via PUS service 200 and the following sequences—see Table 5.8:

Table 5.8 Power subsystem modes and transition sequences

	PSS Modes	**0x01000300**				
Commanding to Off-Target Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Off	0x49000000	1			
	Name	**Table ID**	**WaitTime**	**Check?**		
	Off_Target	0x49000000	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Power Controller	0x43500000				
	PCDU	0x44003200				
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Off_Trans	0x49000100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Power Controller	0x43500000	Off	0	Off	0
	PCDU	0x44003200				
Commanding to Default Target Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Default	0x49010000	1	0x49000000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	Default_Target	0x49010000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Power Controller	0x43500000	Normal	2	Default	0
	PCDU	0x44003200	Normal	2	Idle	1
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Default_Trans	0x49010100	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Power Controller	0x43500000	Normal	2	Default	0

(continued)

Table 5.8 (continued)

	PCDU	0x44003200				
Commanding to Boot Target Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Boot	0x49020000	1	0x49000000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	Boot_Target	0x49020000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Power Controller	0x43500000				
	PCDU	0x44003200	Normal	2	Idle	1
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Boot_Trans1	0x49020100	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Power Controller	0x43500000				
	PCDU	0x44003200	Normal	2	Idle	1
	Name	**Table ID**	**WaitTime**	**Check?**		
	Boot_Trans2	0x49020200	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Power Controller	0x43500000	Boot	16	Default	0
	PCDU					
	Name	**Table ID**	**WaitTime**	**Check?**		
	Boot_Trans3	0x49020300	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Power Controller	0x43500000	Off	0	Default	0
	PCDU					

5.6.4 Power Subsystem Object-IDs, Controller Variables and Limits

For commanding the power subsystem objects from ground via PUS Service 201 and Service 6 (setting of SoC parameters), the following Object_IDs are to be used (see Table 5.9):

Table 5.9 OBSW objects controlling powered devices

Object name	Object-ID hex	Object-ID dec	Description
PSS_CONTROLLER	43500000	1129316352	Main controller object
FUSE_00	43500100	1129316608	Fuse object
FUSE_01	43500101	1129316609	Fuse object
FUSE_02	43500102	1129316610	Fuse object
FUSE_03	43500103	1129316611	Fuse object
FUSE_04	43500104	1129316612	Fuse object
FUSE_05	43500105	1129316613	Fuse object
FUSE_06	43500106	1129316614	Fuse object
FUSE_07	43500107	1129316615	Fuse object
FUSE_08	43500108	1129316616	Fuse object
FUSE_09	43500109	1129316617	Fuse object
FUSE_10	4350010A	1129316618	Fuse object
FUSE_11	4350010B	1129316619	Fuse object
FUSE_12	4350010C	1129316620	Fuse object
FUSE_13	4350010D	1129316621	Fuse object
FUSE_14	4350010E	1129316622	Fuse object
FUSE_15	4350010F	1129316623	Fuse object
FUSE_16	43500110	1129316624	Fuse object
FUSE_17	43500111	1129316625	Fuse object
FUSE_18	43500112	1129316626	Fuse object
FUSE_19	43500113	1129316627	Fuse object
FUSE_20	43500114	1129316628	Fuse object
FUSE_21	43500115	1129316629	Fuse object
FUSE_22	43500116	1129316630	Fuse object
FUSE_23	43500117	1129316631	Fuse object
FUSE_24	43500118	1129316632	Fuse object
FUSE_25	43500119	1129316633	Fuse object
FUSE_26	4350011A	1129316634	Fuse object
PANEL_RIGHT	43500200	1129316864	A solar panel representation
PANEL_LEFT	43500201	1129316865	A solar panel representation
PANEL_CENTER	43500202	1129316866	A solar panel representation

(continued)

Table 5.9 (continued)

Object name	Object-ID hex	Object-ID dec	Description
PANEL_TEST	43500203	1129316867	A solar panel representation
BUS_POWER_SENSOR	43500300	1129317120	Represents current and voltage sensor for bus power
BATTERY_STRING_0	43500400	1129317376	Representation of a battery string
BATTERY_SENSOR_0	43500401	1129317377	The power sensor of one battery string
BATTERY_STRING_1	43500500	1129317632	Representation of a battery string
BATTERY_SENSOR_1	43500501	1129317633	The power sensor of one battery string
BATTERY_STRING_2	43500600	1129317888	Representation of a battery string
BATTERY_SENSOR_2	43500601	1129317889	The power sensor of one battery string

5.6.5 Power Subsystem Variables, Limits, and Parameters

The power subsystem internal variables as listed in Table 5.10 are settings and limit parameters and the table lists the default values. The settings can be changed by PUS Service 6 commands (one at a time).

As this mechanism does not provide an entire settings overview, some multi-parameter adjustment functions are implemented into the *PSS_Controller* which are listed in Table 5.11 and which can be set by PUS Service 8. PCDU settings only can be queried from ground by PUS Service 6.

Table 5.12 lists the PSS relevant parameter monitor variables and the Events triggered in case of a limit violation.

The parameters which are managed by the *PSS_Controller* and which can be accessed via PUS Service 6 are provided in Table 5.13.

Table 5.10 Power subsystem internal computation variables and set/get commands

Name in MIB	Memory Id	Address	Name	Description	Type	Unit	Set TC ID	Get TC ID	TM SPID
PYTDCP00	0x43500000	0	MAX_DELTA_CAPACITY	The maximum allowed delta between the current battery capacity calculated in the PCDU and that in the PSS Controller	fioat	Ah	PYC60001	PYC61001	4900
PYTPUI00	0x43500000	1	PCDU_UPDATE_INTERVAL	The rate in which the capacity values in the PCDU are updated by the OBSW	uint32_t	ms	PYC60002	PYC61002	4901
PYTDEP00	0x43500000	2	PANELS_DEPLOYED	The deployment state as assumed by the PSS Controller. 1: Deployed; 0: Not Deployed	uint8_t	–	PYC60003	PYC61003	4902
PBTCTH00	0x43500400	0	Calibration Current	The current below which a calibration is performed	fioat	A	PYC60000	PYC61000	4903
PBTVTH00	0x43500400	1	Calibration Voltage	The voltage above which a calibration is performed	fioat	V	PYC60000	PYC61000	4904
PBTSTH00	0x43500400	2	Calibration State of Charge	The state of charge above which a calibration is performed	fioat	–	PYC60000	PYC61000	4905
PBTTLI00	0x43500400	3	Temperature Threshold	The temperature above which the capacity is calculated with temperature effects	fioat	°C	PYC60000	PYC61000	4906

(continued)

Table 5.10 (continued)

Name in MIB	Memory Id	Address	Name	Description	Type	Unit	Set TC ID	Get TC ID	TM SPID
PBTTCL00	0x43500400	4	Thermal capacity loss	Parameter for temperature corrected estimation	fioat	Ah/K	PYC60000	PYC61000	4907
PBTTLC00	0x43500400	5	Charge Loss	Parameter for temperature corrected estimation	fioat	–	PYC60000	PYC61000	4908
PBTCCF00	0x43500400	6	Charge Change Factor	Parameter for temperature corrected estimation	fioat	–	PYC60000	PYC61000	4909
PBTTLD00	0x43500400	7	Discharge Loss	Parameter for temperature corrected estimation	fioat	–	PYC60000	PYC61000	4910
PBTDCF00	0x43500400	8	Discharge Change Factor	Parameter for temperature corrected estimation	fioat	–	PYC60000	PYC61000	4911
PBTMCP00	0x43500400	9	Maximum Capacity	The maximum capacity of one string	fioat	Ah	PYC60000	PYC61000	4912
PBTBCC00	0x43500400	10	BCC Current	Current fiowing through the balancers	fioat	A	PYC60000	PYC61000	4913
PBTCTH01	0x43500500	0	Calibration Current	The current below which a calibration is performed	fioat	A	PYC60000	PYC61000	4914
PBTVTH01	0x43500500	1	Calibration Voltage	The voltage above which a calibration is performed	fioat	V	PYC60000	PYC61000	4915
PBTSTH01	0x43500500	2	Calibration State of Charge		fioat	–	PYC60000	PYC61000	4916

(continued)

Table 5.10 (continued)

Name in MIB	Memory Id	Address	Name	Description	Type	Unit	Set TC ID	Get TC ID	TM SPID
				The state of charge above which a calibration is performed					
PBTTLI01	0x43500500	3	Temperature Threshold	The temperature above which the capacity is calculated with temperature effects	fioat	°C	PYC60000	PYC61000	4917
PBTTCL01	0x43500500	4	Thermal capacity loss	Parameter for temperature corrected estimation	fioat	Ah/K	PYC60000	PYC61000	4918
PBTTLC01	0x43500500	5	Charge Loss	Parameter for temperature corrected estimation	fioat	–	PYC60000	PYC61000	4919
PBTCCF01	0x43500500	6	Charge Change Factor	Parameter for temperature corrected estimation	fioat	–	PYC60000	PYC61000	4920
PBTTLD01	0x43500500	7	Discharge Loss	Parameter for temperature corrected estimation	fioat	–	PYC60000	PYC61000	4921
PBTDCF01	0x43500500	8	Discharge Change Factor	Parameter for temperature corrected estimation	fioat	–	PYC60000	PYC61000	4922
PBTMCP01	0x43500500	9	Maximum Capacity	The maximum capacity of one string	fioat	Ah	PYC60000	PYC61000	4923
PBTBCC01	0x43500500	10	BCC Current	Current fiowing through the balancers	fioat	A	PYC60000	PYC61000	4924

(continued)

Table 5.10 (continued)

Name in MIB	Memory Id	Address	Name	Description	Type	Unit	Set TC ID	Get TC ID	TM SPID
PBTCTH02	0x43500600	0	Calibration Current	The current below which a calibration is performed	fioat	A	PYC60000	PYC61000	4925
PBTVTH02	0x43500600	1	Calibration Voltage	The voltage above which a calibration is performed	fioat	V	PYC60000	PYC61000	4926
PBTSTH02	0x43500600	2	Calibration State of Charge	The state of charge above which a calibration is performed	fioat	–	PYC60000	PYC61000	4927
PBTTLI02	0x43500600	3	Temperature Threshold	The temperature above which the capacity is calculated with temperature effects	fioat	°C	PYC60000	PYC61000	4928
PBTTCL02	0x43500600	4	Thermal capacity loss	Parameter for temperature corrected estimation	fioat	Ah/K	PYC60000	PYC61000	4929
PBTTLC02	0x43500600	5	Charge Loss	Parameter for temperature corrected estimation	fioat	–	PYC60000	PYC61000	4930
PBTCCF02	0x43500600	6	Charge Change Factor	Parameter for temperature corrected estimation	fioat	–	PYC60000	PYC61000	4931
PBTTLD02	0x43500600	7	Discharge Loss	Parameter for temperature corrected estimation	fioat	–	PYC60000	PYC61000	4932
PBTDCF02	0x43500600	8	Discharge Change Factor		fioat	–	PYC60000	PYC61000	4933

(continued)

Table 5.10 (continued)

Name in MIB	Memory Id	Address	Name	Description	Type	Unit	Set TC ID	Get TC ID	TM SPID
				Parameter for temperature corrected estimation					
PBTMCP02	0x43500600	9	Maximum Capacity	The maximum capacity of one string	fioat	Ah	PYC60000	PYC61000	4934
PBTBCC02	0x43500600	10	BCC Current	Current fiowing through the balancers	fioat	A	PYC60000	PYC61000	4935

Table 5.11 Power subsystem functions (including parameters) commandable from ground

Name in MIB	Command Name	Verification	Object ID HEX	Object ID DEC	Function ID	P1 Name	P1 Type	P1 Values	P2 Name	P2 Type	P2 Values	P3 Name	P3 Type	P3 Values
PYC00100	SET_MAXIMUM_CAPACITY_OF_STRINGS	ASC	4.4E+07	1.129E+09	1	capacity_string_0	fioat	0-12 [Ah]	capacity_string_1	fioat	0–12 [Ah]	capacity_string_2	fioat	0–12 [Ah]
PYC0020 [0-2]	SET_SOC_ESTIMATION_PARAMETERS	ASC	4.4E+07	1.129E+09	2	stringId	uint8	0–2	[TBD]					
PYC00300	UPDATE_SOC_ESTIMATION_PARAMETERS_IN_PCDU	ASPC	4.4E+07	1.129E+09	3	stringId	uint8	0–2						

Table 5.12 Power subsystem monitors

Owner Name	Object Id	Parameter Name	Parameter #	Description	Limit ID	Type	Event below low	Event above high	Set TC
PSS_CONTROLLER	43500000	PBTSCC03	40003300	PSS State of Charge Warning	1	fioat	SOC_WARNING	SOC_HIGH	PYC12000
PSS_CONTROLLER	43500000	PBTSCC03	40003300	PSS State of Charge Critical	2	fioat	SOC_CRITICAL	SOC_HIGH	PYC12000
PSS_CONTROLLER	43500000	PBTSCC03	40003300	PSS State of Charge Shutdown	3	fioat	SOC_SHUTDOWN	SOC_HIGH	PYC12000
BATTERY_SENSOR_0	43500401	PBTSTC00	40000200	String current check	1	fioat	BATTERY_CURRENT_LOW	BATTERY_CURRENT_HIGH	PYC12000
BATTERY_SENSOR_0	43500401	PBTSTV00	40000300	String voltage check	1	fioat	BATTERY_VOLTAGE_LOW	BATTERY_VOLTAGE_HIGH	PYC12000
BATTERY_SENSOR_1	43500501	PBTSTC01	40000201	String current check	1	fioat	BATTERY_CURRENT_LOW	BATTERY_CURRENT_HIGH	PYC12000
BATTERY_SENSOR_1	43500501	PBTSTV01	40000301	String voltage check	1	fioat	BATTERY_VOLTAGE_LOW	BATTERY_VOLTAGE_HIGH	PYC12000
BATTERY_SENSOR_2	43500601	PBTSTC02	40000202	String current check	1	fioat	BATTERY_CURRENT_LOW	BATTERY_CURRENT_HIGH	PYC12000
BATTERY_SENSOR_2	43500601	PBTSTV02	40000302	String voltage check	1	fioat	BATTERY_VOLTAGE_LOW	BATTERY_VOLTAGE_HIGH	PYC12000
FUSE_00	43500100	PPTFSC00	41110400	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_01	43500101	PPTFSC01	41110401	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000

(continued)

Table 5.12 (continued)

Owner Name	Object Id	Parameter Name	Parameter #	Description	Limit ID	Type	Event below low	Event above high	Set TC
FUSE_02	43500102	PPTFSC02	41110402	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_03	43500103	PPTFSC03	41110403	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_04	43500104	PPTFSC04	41110404	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_05	43500105	PPTFSC05	41110405	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_06	43500106	PPTFSC06	41110406	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_07	43500107	PPTFSC07	41110407	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_08	43500108	PPTFSC08	41110408	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_09	43500109	PPTFSC09	41110409	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_10	4350010A	PPTFSC10	4111040A	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_11	4350010B	PPTFSC11	4111040B	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_12	4350010C	PPTFSC12	4111040C	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_13	4350010D	PPTFSC13	4111040D	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_14	4350010E	PPTFSC14	4111040E	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_15	4350010F	PPTFSC15	4111040F	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_16	43500110	PPTFSC16	41110410	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000

(continued)

Table 5.12 (continued)

Owner Name	Object Id	Parameter Name	Parameter #	Description	Limit ID	Type	Event below low	Event above high	Set TC
FUSE_17	43500111	PPTFSC17	41110411	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_18	43500112	PPTFSC18	41110412	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_19	43500113	PPTFSC19	41110413	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_20	43500114	PPTFSC20	41110414	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_21	43500115	PPTFSC21	41110415	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_22	43500116	PPTFSC22	41110416	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_23	43500117	PPTFSC23	41110417	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_24	43500118	PPTFSC24	41110418	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_25	43500119	PPTFSC25	41110419	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
FUSE_26	4350011A	PPTFSC26	4111041A	Fuse current check	1	fioat	FUSE_CURRENT_LOW	FUSE_CURRENT_HIGH	PYC12000
BUS_POWER_SENSOR	43500300	PPTBUC00	41112200	Bus current check	1	fioat	BUS_CURRENT_LOW	BUS_CURRENT_HIGH	PYC12000
BUS_POWER_SENSOR	43500300	PPTBUV00	41111100	Bus voltage check	1	fioat	BUS_VOLTAGE_LOW	BUS_VOLTAGE_HIGH	PYC12000
PANEL_RIGHT	43500200	PSTCUR00	48000200	Panel current check	1	fioat	PANEL_CURRENT_LOW	PANEL_CURRENT_HIGH	PYC12000
PANEL_RIGHT	43500200	PSTVLT00	48000300	Panel voltage check	1	fioat	PANEL_VOLTAGE_LOW	PANEL_VOLTAGE_HIGH	PYC12000
PANEL_LEFT	43500201	PSTCUR01	48000201	Panel current check	1	fioat	PANEL_CURRENT_LOW	PANEL_CURRENT_HIGH	PYC12000

(continued)

Table 5.12 (continued)

Owner Name	Object Id	Parameter Name	Parameter #	Description	Limit ID	Type	Event below low	Event above high	Set TC
PANEL_LEFT	43500201	PSTVLT01	48000301	Panel voltage check	1	fioat	PANEL_VOLTAGE_LOW	PANEL_VOLTAGE_HIGH	PYC12000
PANEL_CENTER	43500202	PSTCUR02	48000202	Panel current check	1	fioat	PANEL_CURRENT_LOW	PANEL_CURRENT_HIGH	PYC12000
PANEL_CENTER	43500202	PSTVLT02	48000302	Panel voltage check	1	fioat	PANEL_VOLTAGE_LOW	PANEL_VOLTAGE_HIGH	PYC12000
PANEL_TEST	43500203	PSTCUR03	48000203	Panel current check	1	fioat	PANEL_CURRENT_LOW	PANEL_CURRENT_HIGH	PYC12000
PANEL_TEST	43500203	PSTVLT03	48000303	Panel voltage check	1	fioat	PANEL_VOLTAGE_LOW	PANEL_VOLTAGE_HIGH	PYC12000

Table 5.13 Power subsystem parameters managed by the *PSS_Controller*

MIB Name	PID	Name	Unit	Type	Description
PPTBUP00	0x41111300	BUS_POWER	W	fioat	The bus power calculated with current and voltage
PBTSCC03	0x40003300	BATT_SOC_PSS	–	fioat	Combined state of charge calculated by PSS Controller
PBTCPC03	0x40003700	BATT_CAP_PSS	Ah	fioat	Combined capacity calculated by PSS Controller
PBTSCC00	0x40003000	BATT_SOC_PSS_S0	–	fioat	State of Charge of one string
PBTSCC01	0x40003100	BATT_SOC_PSS_S1	–	fioat	State of Charge of one string
PBTSCC02	0x40003200	BATT_SOC_PSS_S2	–	fioat	State of Charge of one string
PBTCPC00	0x40003400	BATT_CAP_PSS_S0	Ah	fioat	Capacity of one string
PBTCPC01	0x40003500	BATT_CAP_PSS_S1	Ah	fioat	Capacity of one string
PBTCPC02	0x40003600	BATT_CAP_PSS_S2	Ah	fioat	Capacity of one string
PBTPWR00	0x40001000	BAT_STR_POWER_0	W	fioat	Power in/out of one string
PBTPWR01	0x40001100	BAT_STR_POWER_1	W	fioat	Power in/out of one string
PBTPWR02	0x40001200	BAT_STR_POWER_2	W	fioat	Power in/out of one string
PSTPWR00	0x48000400	PANEL_POWER_RIGH	W	fioat	Solar panel power
PSTPWR01	0x48000500	PANEL_POWER_LEFT	W	fioat	Solar panel power
PSTPWR02	0x48000600	PANEL_POWER_CENT	W	fioat	Solar panel power
PSTPWR03	0x48000700	PANEL_POWER_TEST	W	fioat	Solar panel power
PPTPWR00	0x41120000	FUSE_POWER_0	W	fioat	Fuse power
PPTPWR01	0x41120100	FUSE_POWER_1	W	fioat	Fuse power
PPTPWR02	0x41120200	FUSE_POWER_2	W	fioat	Fuse power
PPTPWR03	0x41120300	FUSE_POWER_3	W	fioat	Fuse power
PPTPWR04	0x41120400	FUSE_POWER_4	W	fioat	Fuse power
PPTPWR05	0x41120500	FUSE_POWER_5	W	fioat	Fuse power
PPTPWR06	0x41120600	FUSE_POWER_6	W	fioat	Fuse power
PPTPWR07	0x41120700	FUSE_POWER_7	W	fioat	Fuse power
PPTPWR08	0x41120800	FUSE_POWER_8	W	fioat	Fuse power
PPTPWR09	0x41120900	FUSE_POWER_9	W	fioat	Fuse power
PPTPWR10	0x41120A00	FUSE_POWER_10	W	fioat	Fuse power
PPTPWR11	0x41120B00	FUSE_POWER_11	W	fioat	Fuse power

(continued)

Table 5.13 (continued)

MIB Name	PID	Name	Unit	Type	Description
PPTPWR12	0x41120C00	FUSE_POWER_12	W	fioat	Fuse power
PPTPWR13	0x41120D00	FUSE_POWER_13	W	fioat	Fuse power
PPTPWR14	0x41120E00	FUSE_POWER_14	W	fioat	Fuse power
PPTPWR15	0x41120F00	FUSE_POWER_15	W	fioat	Fuse power
PPTPWR16	0x41121000	FUSE_POWER_16	W	fioat	Fuse power
PPTPWR17	0x41121100	FUSE_POWER_17	W	fioat	Fuse power
PPTPWR18	0x41121200	FUSE_POWER_18	W	fioat	Fuse power
PPTPWR19	0x41121300	FUSE_POWER_19	W	fioat	Fuse power
PPTPWR20	0x41121400	FUSE_POWER_20	W	fioat	Fuse power
PPTPWR21	0x41121500	FUSE_POWER_21	W	fioat	Fuse power
PPTPWR22	0x41121600	FUSE_POWER_22	W	fioat	Fuse power
PPTPWR23	0x41121700	FUSE_POWER_23	W	fioat	Fuse power
PPTPWR24	0x41121800	FUSE_POWER_24	W	fioat	Fuse power
PPTPWR25	0x41121900	FUSE_POWER_25	W	fioat	Fuse power
PPTPWR26	0x41121A00	FUSE_POWER_26	W	fioat	Fuse power
PPTPWR99	0x4112FF00	FUSE_POWER_ALL	W	fioat	Combined Fuse power

The power subsystem relevant TC packet definitions are included in the spacecraft TC table in annex Sect. 17.3.1.

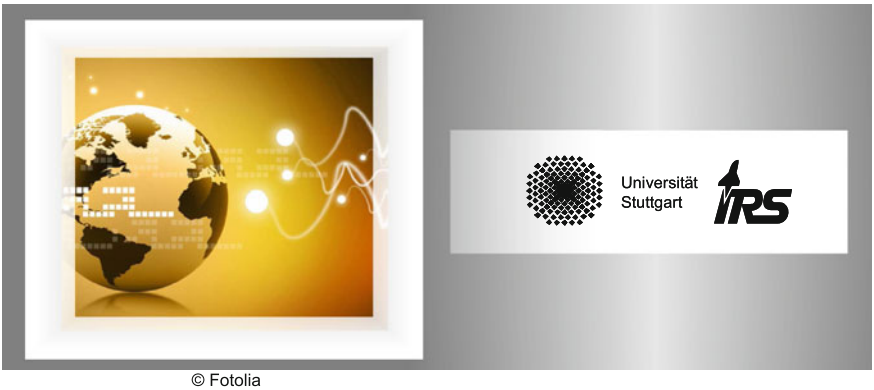
The power subsystem relevant TM packet definitions are included in the spacecraft TM table in annex Sect. 17.3.2.

The power subsystem relevant Event TM packets are included in the spacecraft Event TM table in annex Sect. 17.3.3.

Details on the TM parameter positions in the diverse packets have to be taken from the MIB.

Chapter 6
Platform Communication Subsystem

Jens Eickhoff and Kai-Sören Klemich



Abstract The chapter covers the Telemetry, Tracking and Command (TTC) subsystem of the FLP platform. It provides the S-Band link for the platform control. It provides a brief overview on the receivers, transmitters and on the CCSDS decoder/encoder boards of the onboard computer which logically belong to the TTC subsystem. Technical parameters on the equipment and an example link budget are included in the book's annexes. After explanation of the signal acquisition procedure the last part of the chapter comprises the onboard software elements and the on-board functionality for control of the platform communication subsystem.

J. Eickhoff (✉)
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

K.-S. Klemich
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: klemich@irs.uni-stuttgart.de

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_6

Keywords TTC subsystem overview · Receivers · Transmitters · CCSDS-Boards · Antennas · Downlink/uplink parameters · Signal acquisition procedure · TTC subsystem control

6.1 TTC Subsystem Overview

For commanding the satellite from ground and for downlink of platform housekeeping and Event telemetry a "Telemetry, Tracking and Command" subsystem—or TTC subsystem—is implemented on board the platform. It consists of two S-band receivers and two S-band transmitters (see Fig. 6.1). Each transmitter/receiver pair has an individual diplexer for parallel signal up- and downlink and both diplexers are coupled to the antennas via a 3 dB hybrid coupler.

The TC receivers (RX) are operated in hot redundancy throughout all spacecraft modes. Therefore they also are named with RX-0 and RX-1 in Fig. 1.6. The transmitters (TX) are only operated in ground station vicinity and only one transmitter is operated at a time because otherwise the signals from both transmitters would interfere. Therefore their naming convention follows the one for cold

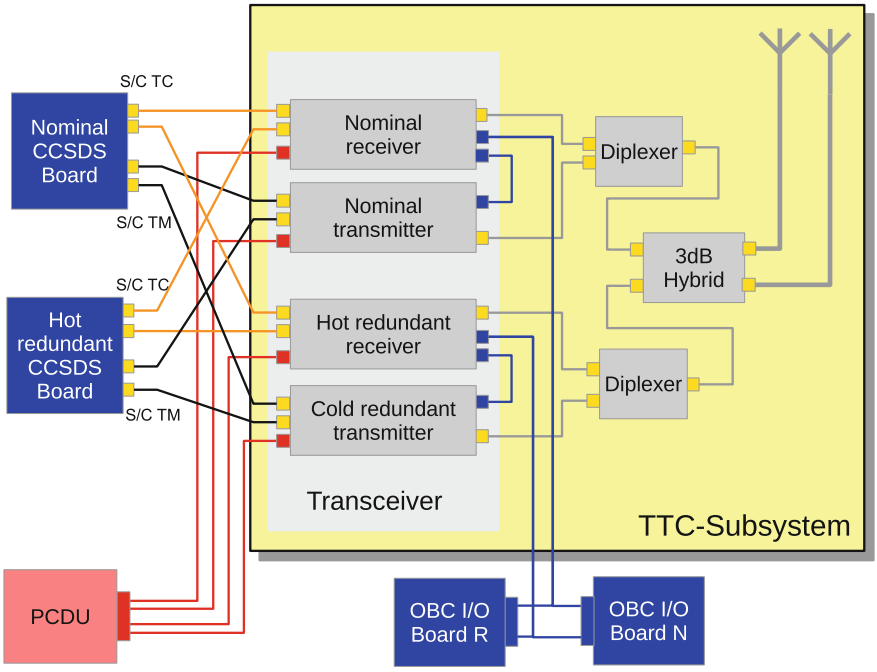


Fig. 6.1 TTC subsystem block diagram. © IRS, University of Stuttgart

redundant units—TX-N and TX-R. If telemetry downlink from the satellite seems to fail when trying to establish contact (e.g. after a ping command), switchover to the redundant transmitter will be commanded by ground (see Sect. 10.5).

The antennas are S-band turnstile antennas, which are mounted on opposite sides of the satellite (see Fig. 1.5). As they provide an approximately hemispherical radiation pattern [72] it is possible to send signals to and receive signals from the satellite at all times independent of its attitude (see Fig. 6.2).

Due to the two TTC antennas on opposite sides of the satellite in the case of the "Flying Laptop" the signals transmitted from both TTC antennas interfere in and around a plane perpendicular to the antennas' boresight. If the satellite's attitude is such that the ground station is within or at roughly ±25° of this plane during a pass, the signal from the satellite can be lost, which for example can occur in Idle Mode or Safe Mode (see Sect. 6.3.5). To minimize this problem the platform is equipped with the Coarse Nadir Pointing Mode (Sect. 2.1.7) to be able to perform certain critical operations from ground without an intermediate contact loss.

In the case of the "Flying Laptop", the payload Data Downlink System (DDS) also uses an S-band frequency (with a dedicated horn antenna visible in Fig. 1.5). In order to be able to separate the TTC signals from the payload data downlink system, both TTC antennas use right-hand circular polarization (RHCP) and for the DDS left-hand circular polarization (LHCP) is used.

Since the receivers are running in hot redundant they both deliver spacecraft TC data to the CCSDS-Board side (see Fig. 6.1). The CCSDS-Boards are designed such that they look for the first receiver providing carrier lock and bit lock signal. The separate lines for these signals are not depicted in Fig. 6.1 to avoid over-complexifying it. As only one transmitter is operated at a time, the CCSDS-Boards can also be coupled from their output to both transmitters. Thus the TM-submitting CCSDS-Board (selected by the OBSW) electrically distributes its spacecraft TM signal to both transmitters and does not have to care about which one being active. So the cross-coupling comprises:

- Both receivers are coupled to both CCSDS-Boards.
- Both transmitters are coupled to both CCSDS-Boards.
- The CCSDS-Boards output sides towards the Processor-Boards are also cross-coupled (please refer to Fig. 1.6), and
- the CLCW signal also is cross-exchanged between the CCSDS-Boards for correct down reporting depending on selected active TX and selected uplink Virtual Channel (VC).

The TTC subsystem operates innon-coherent mode. The detailed parameters for downlink and uplink are provided in Tables 6.1 and 6.2.

The subsystem performance data were evaluated in the RF compatibility test together with the 15 m antenna of the DLR antenna station in Weilheim, Germany. The details can be found in [79]. Only the operationally relevant data will be listed here:

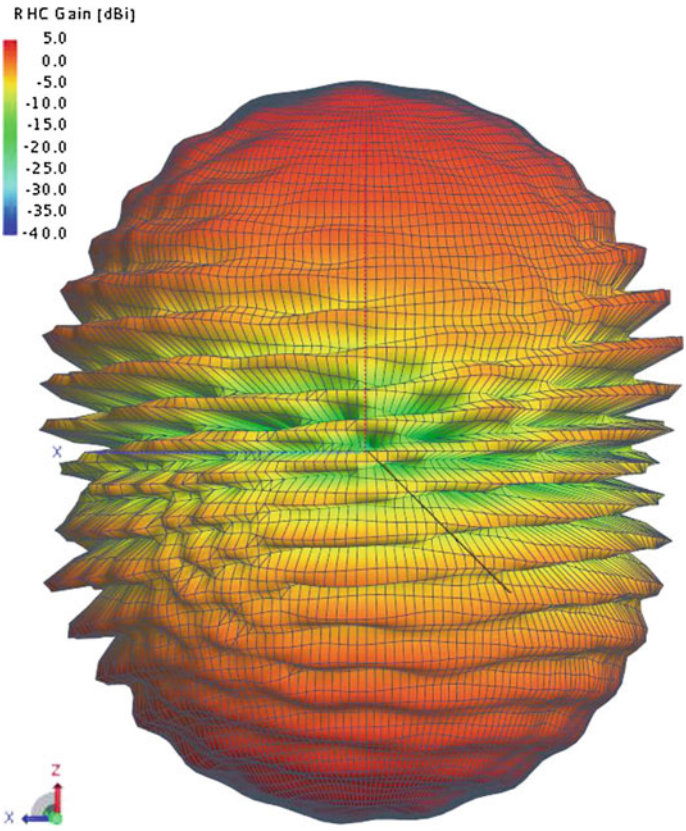


Fig. 6.2 S-band antenna far field simulation. © IRS, University of Stuttgart

Table 6.1 TTC subsystem downlink parameters

Downlink frequency:	TX1 = 2.263500 GHz
Polarization:	RHCP
Modulation:	BPSK
Symbol rate:	128.205 kbps NRZ-L Viterbi, Randomized, Reed Solomon
Sync word:	0x 1ACFFC1D
TLM frame length:	1115 byte (Frame) + 4 byte (ASM) + 160 byte (Reed Solomon Block) = 1279 byte
Forward Error Correction Encoding:	NRZ-L with • Reed-Solomon Coding: E = 16 (255,223), Interleave Depth = 5 • Convolutional Coding K = 7 Rate ½, G1/G2 (not punctuated)
Frame Error Control Field (FECW)	Not used
Transfer Frame Secondary Header	Not used

Table 6.2 TTC subsystem uplink parameters

Uplink frequency:	RX1 = 2.083500 GHz
Polarization:	RHCP
Modulation:	PCM/BPSK/PM
Subcarrier:	16 kHz sine wave
Bitrate:	4.000 kbps NRZ-L
Modulation index:	1.0 rad peak ±5 %
Encoding:	BCH Code: (63,56)
Sync. Word:	EB90
Carrier Sweep:	Settings for optimum lock: ±100 kHz around the center Frequency sweep rate 30 kHz/s up to 50 kHz/s with zig-zag method

- The test results proved that a special TC uplink acquisition procedure—under consideration of Doppler frequency offset—is not necessary (max Doppler offset: FDL ±60 kHz).
- The bit error rates of the transmitters have been measured in combination with the DLR antenna and the according diagrams are included in the annex Sect. 17.5.1 for reference to compare against in cases of potential degradations in flight.

6.2 Signal Acquisition Procedure

The acquisition procedure to be used with the receivers is fully in accordance with ECSS-E-50-05A. While following the specified procedure, acquisition is certain for both receivers after one triangular acquisition sweep.

HK downlink is not compulsory and blind acquisition is possible. At first ground contact of the satellite when the LEOP-Completion Flag is still Zero, the satellite will automatically activate not only the TTC receivers, but also the nominal TTC transmitter (see Sect. 11.6.2). Later during the mission, when the LEOP Completion Flag was set to 1, the transmitter activation is done by the *TTC-Controller*.

Activation then is initiated either as result of time tagged transmitter activation commands in the timeline or as result to receiver lock detection. It is foreseen to preprogram transmitter activation commands into the mission timelines so that the satellite activates the transmitter some minutes before ground station passover.

Note: During acquisition, no data or subcarrier modulation shall be present on the RF carrier transmitted by the ground station.

Link establishment procedure:

1. Switch on carrier frequency and set to nominal center
2. Initiate one triangular sweep around the frequency set in (1) with an amplitude of ±100 kHz at 50 kHz/s. Analog sweep would be preferred. If frequency generation is digital, step size/discontinuity shall not exceed 100 Hz.
3. Switch on modulation

6.3 TTC Subsystem Control

While in Fig. 6.1 the TTC subsystem is shown to comprise the transceiver, diplexers and coupler, from point of view of the OBSW the TTC subsystem already starts at the according Processor-Board SpaceWire interfaces, as these are used to read uplinked TCs and to submit TM for downlink. Thus the CCSDS-Boards are part of the TTC subsystem for the OBSW .

The TTC subsystem control is managed by the SW objects as depicted in Fig. 6.3. Main subsystem object is the *TTC_Subsystem* ("TTC" in the figure) which manages the *TTC_Controller* and the assembly objects for the CCSDS-Boards, the receivers and the transmitters. The assembly objects then manage

- the individual CCSDS-Board handlers CCSDS_Handler 0 and 1 for the hot redundant boards,
- the TTC_RX_Handlers 0 and 1 for the hot redundant receivers and the
- TTC_TX_Handlers N and R for the cold redundant transmitters.

The individual device handlers manage the health monitoring of the devices. The TTC subsystem cycle rate is provided in Sect. 3.4.2.

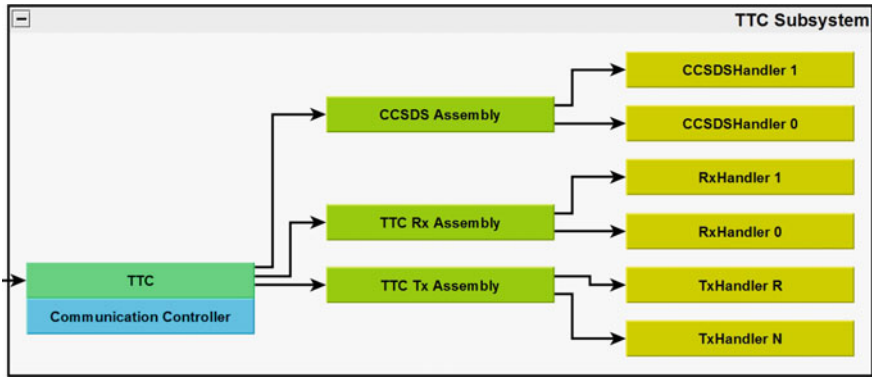


Fig. 6.3 TTC subsystem control elements in OBSW. © IRS, University of Stuttgart

6.3.1 TTC Subsystem Device Handlers

6.3.1.1 Receiver and Transmitter Handlers

The TTC_RX_Handlers also acquire the transceiver HK telemetry which is supplied for both TX and RX via one interface to the OBC/OBSW. The TTC subsystem HK telemetry is treated further in Sect. 6.3.5. The device handler modes are listed in Table 6.3.

Table 6.3 TTC TX and RX device handler modes

Handler	Mode	M#	Submode	SM#	Description
TTC Tx N/R	Off	0	Default	0	Off
	On	1	Default	0	Transmitter On
TTC Rx 0–1	Off	0	Default	0	Off. DO NOT USE!
	On	1	Default	0	Receiver On
	Normal	2	Default	0	Receiver On and TTC-HK telemetry being requested periodically
	Raw	3	Default	0	Default raw mode. Not useful

6.3.1.2 CCSDS-Board Handler

Apart from purely managing the CCSDS-Board hardware, the *CCSDS_Handler* class is responsible for the reception of telecommands (TC) from and sending of telemetry (TM) to ground. This is performed in conjunction with the CCSDS-Board hardware, which provides the CCSDS channel decoding/coding layer for TC and TM as well as the data link layer for TM. It is a hardware component of the OBC, but belongs logically to the TTC subsystem.

The CCSDS-Handler's main tasks are:

- Managing the CCSDS-Board hardware, which includes initialization, mode handling and configuration.
- For TC, the class manages packet extraction and the execution of all data link layer functionality, including COP-1, as defined in the CCSDS data link layer.
- Checking the "bit lock" and "RF available" signals and throwing events in case of changes.

The CCSDS-Handlers cyclically read out incoming TC frames from a buffer in the boards. The handlers internally forward the data to the software-based Data Link Layer, which checks the frames and extracts the TC packets. The packets are forwarded to a dedicated TC distribution (object/component) which inspects and forwards them to their according destination PUS service. Packets not yet processed further by the PUS service are buffered in the service's command reception queue.

Table 6.4 Events triggerd by the CCSDS-Board handlers

Event Name	EventId dec	Severity	Sub-service	Info
RF_AVAILABLE	7900	INFO	1	A RF signal was detected
RF_LOST	7901	INFO	1	The RF signal has faded
BIT_LOCK	7902	INFO	1	The receiver locks to a signal
BIT_LOCK_LOST	7903	INFO	1	The receiver lost the signal lock
FRAME_PROCESSING_FAILED	7904	LOW	2	Frame processing detected errors in the frame
DEVICE_SENDING_COMMAND_FAILED	2801	LOW	2	The board tried to send a command via RMAP, which failed
DEVICE_REQUESTING_REPLY_FAILED	2802	LOW	2	The board tried to fetch a reply via RMAP, which failed

If the queue is full, new incoming TC packets are discarded and a TC_Verification_Acceptance_Failed (1, 2) Report is generated.

All telemetry packets generated by PUS services are routed via the CCSDS-Board Assembly to the currently active CCSDS-Handler, which is responsible for forwarding the packets to the CCSDS-Board. This is necessary to avoid loosing or duplicating packets in the case where the active CCSDS-Board is reconfigured or telemetry forwarding is disabled entirely. The CCSDS-Board autonomously assembles transfer frames from the incoming packets. There are four Virtual Channels to forward TM packets. Each PUS service is statically configured to utilize one of these channels (e.g. live HK TM on VC0, Event TM on VC1, playback TM on VC2).

The CCSDS-Board handlers provide the following modes (see also Table 6.6):

- MODE_OFF: The power switches are opened, the board is turned off. Submode must be 0.

- MODE_ON: The board is turned on and configured for frame reception (SUBMODE_PASSIVE, 0). If the submode is SUBMODE_ACTIVE (1), the board will start forwarding TM frames.

The handlers also look for the "bit lock" and "RF available" signal lines attached to the boards. In case of a change on the line, the handler will trigger an Event, indicating the change. In addition the handlers throw Events in case of errors, e.g. in case of errors occurring during board initialization or during board communication. The defined Events are listed in Table 6.4.

In addition, both CCSDS-Board handlers can trigger the typical mode events (i.e. CHANGING_MODE, MODE_INFO, MODE_TRANSITION_FAILED).

All settings in the CCSDS-Boards are configured by default to match the configuration required for FLP operation. However, the handler provides memory access functionality to change the default configuration. There are two options:

- Directly changing settings in the CCSDS-Board registers:
  This is done by sending a memory load or memory dump command (Service 6) with the physical register address (see [52]) to one of the CCSDS-Board handlers. The address will be checked for validity. These changes are non-permanent, i.e. they will be overwritten when powercycling the CCSDS-Board.
- Changing the settings in the CCSDS-Board handler:
  This is possible for two registers which configure the TM bit rate and the encoding. The possible values are listed in Table 6.5. Details are described in [52]. For a change in one of those parameters to take effect, the CCSDS-Board needs to be power-cycled.

Table 6.5 Memory addresses in the CCSDS-Board handler

Object-ID	Memory address	Size	Info
0x44470000 0x44470100	0	4	Physical Layer Register: Handles TM bit rate
0x44470000 0x44470100	1	4	Coding sublayer register: Handles variants of TM channel coding

Table 6.6 CCSDS-Board device handler modes

Handler	Mode	M#	Submode	SM#	Description
CCSDS 0–1	On	1	Passive	0	No TM and idle frame forwarding
			Active	1	TM forwarding enabled and idle frame generation

6.3.2 TTC Subsystem Assemblies

6.3.2.1 TTC Receiver and Transmitter Assemblies

The logic for equipment switching according to health status or ground command is implemented in the *TTC_RX_Assembly* and the *TTC_TX_Assembly* respectively. The individual statuses into which the handlers and assemblies can be switched—including the according effect on the TC/TM transmission—can be found in the Tables 6.3 and 6.6. The *TTC_RX_Assembly* will always keep at least one receiver on. This means, that if e.g. *Rx-0* is marked as defective and ground commands *RX-1* off, the *TTC_RX_Assembly* will ignore the command (see also Mode 0, Submode 0 in Table 6.3).

Table 6.7 TTC TX and RX assembly modes

Assembly	Mode	M#	Submode	SM#	Description
TTC Tx	Off	0	Default	0	Both Transmitters off. Does not force them off when under external control
	On	1	Default	0	Always exactly one Transmitter is On
TTC Rx	On	1	Default	0	Same as children's modes. All healthy Receivers are switched On. Will overwrite health if not at least one child is ON
	Normal	2	Default	0	Same as children's modes. All healthy Receivers are switched to Normal. Will overwrite health if not at least one child is in Normal mode

6.3.2.2 CCSDS-Board Assembly

To manage the CCSDS-Board redundancy, the CCSDS-Board assembly object (*CCSDS_Assembly*) is introduced, which encapsulates the dual hot-redundancy from the rest of the system.

The assembly itself has one mode with two submodes, (ON, PASSIVE) and (ON, ACTIVE)—see also Table 6.8. Thus, it always tries to keep one board at least in a receiving state (if in passive mode), or exactly one sending and receiving if in submode "Active". The main operations logic is depicted in Fig. 6.4.

Table 6.8 CCSDS-Board assembly modes

Assembly	Mode	M#	Submode	SM#	Description
CCSDS	On	1	Passive	0	Both children in passive mode. Will overwrite health if not at least one is passive
			Active	1	One child in passive, one in active mode. Will overwrite health if not at least one is active

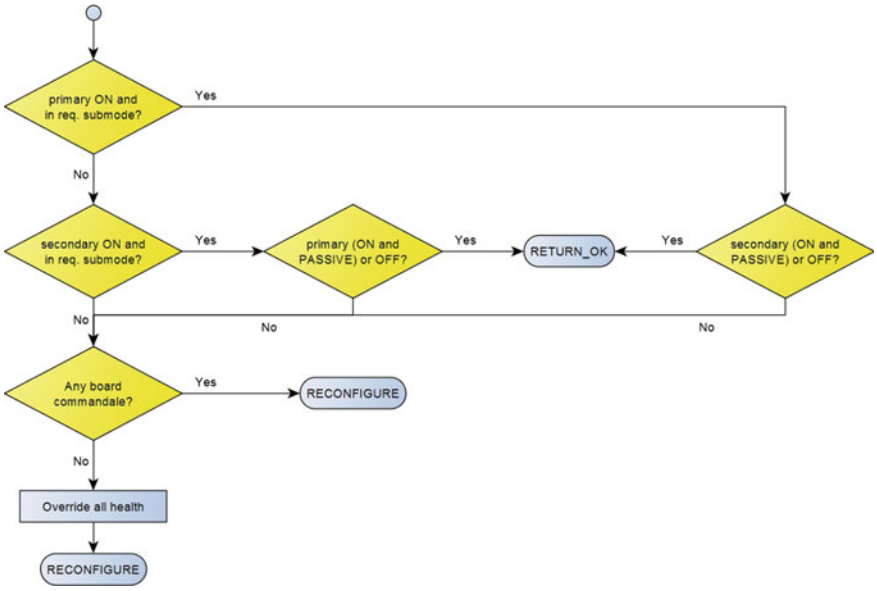


Fig. 6.4 Availability check in *CCSDS_Assembly*. © IRS, University of Stuttgart

The assembly is able to overwrite the boards' health information if there's no other way to reach the required state.

The CCSDS-Board assembly is also responsible for forwarding TM to the currently active board and discarding TM in case both boards are passive.

6.3.3 TTC Subsystem Object-IDs

For commanding the TTC subsystem objects from ground via PUS services 20×, the following Object_IDs are to be used (see Table 6.9):

Table 6.9 TTC subsystem Object_IDs

Device/Assembly/	Object_ID
CCSDS_HANDLER_0	0x44470000
CCSDS_0_TM_HEALTHDEVICE	0x44470001
CSDS_HANDLER_1	0x44470100
CCSDS_1_TM_HEALTHDEVICE	0x44470101
TTC_RX_HANDLER_N	0x44000600
TTC_RX_HANDLER_R	0x44000700
TTC_TX_HANDLER_N	0x44000800
TTC_TX_HANDLER_R	0x44000900
TTC_RX_ASSEMBLY	0x41000600
TTC_TX_ASSEMBLY	0x41000700
CCSDS_ASSEMBLY	0x41100000

6.3.4 TTC Controller

As the main duty in controlling the TTC lies in changing the mode of the entire subsystem, the controller functionality of the TTC subsystem was merged into the subsystem object itself, which is extended for this case (see Fig. 6.3). This is a design exception versus the other subsystems. Therefore, an external controller object is not required.

The following control functionality is required:

- Enable a transmitter in case a receiver lock is detected.
- Disable a transmitter a certain time after a lost signal.
- Turn on a transmitter if in LEOP phase.
- Provide a method to enable a transmitter after a certain period of time.

The control functionality directly changes the TTC subsystem modes as shown in Tables 6.3, 6.6, 6.7 and 6.8 to control the subsystem. It watches for events emitted by the CCSDS-Boards to check the current link state and maintains some internal timers.

The control algorithm state machine is shown in Fig. 6.5.

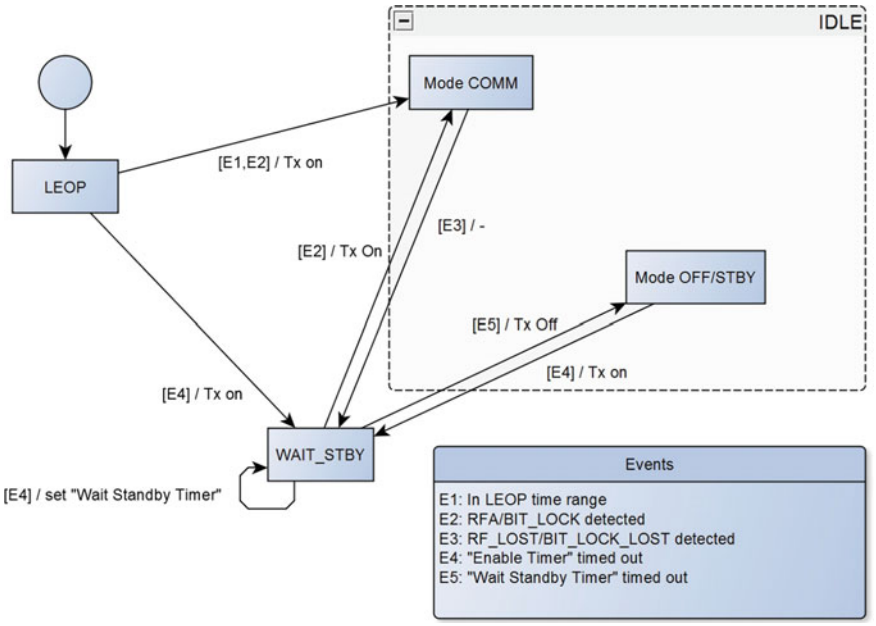


Fig. 6.5 TTC state machine. © IRS, University of Stuttgart

- Normally, the controller will look for RF_AVAILABLE and RF_LOST events to decide for activation of the transmitter and for deactivation after end of ground station contact.
  Note:
  In case the transmitter is turned off too often erroneously, it is configurable to use the less sensitive BIT_LOCK/BIT_LOCK_LOST events instead.
- To avoid a blind acquisition during LEOP phase, the controller has a hard-coded range of on-board time (see Sect. 2.6), within which the controller will turn on the transmitter directly. After turning on the transmitter once, it operates nominally and will be deactivated by ground after first ground contact via time-tagged command. This procedure is repeated after every reboot during the configured LEOP period. The default period is from 300 s after first boot up to 1 week after launch.
- In addition, the controller utilizes a function to turn on the transmitters independent of the scheduling service. It is based on a timer and therefore is not sensitive to changes and time jumps, as the scheduling service is. The transmitter is turned on for a configurable duration.

Table 6.10 TTC subsystem submodes

Object	Submode	SM#	Description
TTC Subsystem 0x01000400	CTRL	0	Ctrler changes TTC subsystem modes
	NO_CTRL	1	Off, no automatic mode changes

Table 6.11 TTC_Controller variables

Memory ID	Name	Addr.	Size	Type	Description
0x01000400	COMM_TIMEOUT	0	4	uint32_t	After RF_LOST Event, the Tx is turned off after this interval [ms]
0x01000400	ENABLE_FOR	1	4	uint32_t	When using the timer, the TX stays on for this time [ms]
0x01000400	USE_RFA	2	1	uint8_t	1: Ctrl. uses RFA/RFA_LOST 0: Ctrl. uses BIT_LOCK/_LOST

Enabling and disabling the controller functionality within the TTC subsystem is done by means of the submode, as shown in Table 6.10. SUBMODE_CTRL (0) as the default value enables the controller functionality, SUBMODE_NO_CTRL (1) disables all controller functionalities (Table 6.11). In this case, no autonomous mode changes are performed.

6.3.5 TTC Subsystem Mode Transitions and Telemetry

The TTC subsystem mode commanding—according to the TTC modes defined in Tables 6.6 and 6.8—is performed via PUS service 200 and the following sequences —see Table 6.12.

Apart from mode-switching via Srv. 200 controlled command sequences and Srv. 20× based object commanding of handlers and assemblies. The TTC subsystem does not have any dedicated TCs.

The transceivers provide monitor signals—through the HK-TM-line of its receivers to the OBC I/O-Boards. This HK-TM is managed by the Rx handlers. Only the transceiver internal temperatures are monitored on board. The other parameters concern the RF electronics and allow to draw conclusions about the health/degradation of the unit during lifetime.

For this purpose, drifts of the monitor signals and the like are much more interesting than their absolute values. They are not intended to be absolute and highly accurate measuring results. The parameters are collected cyclically and are transmitted to ground as HK-TM for evaluation. On ground the main monitored parameters are the Temp Monitor values which report the internal unit temperatures. The transceiver HK-TM parameter list is depicted in Table 6.13.

For the Parameter Type Codes and Format Codes please refer to Table 16.2.

Some further explanatory information on the data is given in Table 6.14 below. These parameters mainly serve for monitoring in regular intervals (e.g. once per month) the health, respectively potential degradation effects of the TTC subsystem.

Please also refer to the annex Sect. 17.5.2 for further reference curves to identify potential degradations w.r.t. the measured BoL statuses and for using these parameters for failure identification.

The TM parameters of the CCSDS-Boards are provided in Table 6.15. Besides statuses for RF-lock and Bit-lock and the TC availability and transmitter readiness TM set comprises a number of parameters evaluated only in case of communication problems. The parameters serve for health management by the *TTC_Controller* and for additional evaluation on ground (Table 6.16).

Table 6.12 TTC subsystem modes and transition sequences

	TTC Modes	**0x01000400**				
Commanding to Default Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Off	0x69000000	1			
	Name	**Table ID**	**WaitTime**	**Check?**		
	Off_Target	0x69000000	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Communic. Controller					
	Rx Assembly					
	Tx Assembly					
	CCSDS-Board Ass.					
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Off_Trans	0x69000100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Communic. Controller	0x43520000	Off	0	Off	0
	Rx Assembly	0x41000600	On	1	Default	0
	Tx Assembly	0x41000700	Off	0	Off	0
	CCSDS-Board Ass.	0x41100000	On	1	Passive	0
Commanding to Comm Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Comm	0x69010000	1	0x69020000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	Comm_Target	0x69010000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**

(continued)

Table 6.12 (continued)

	Communic. Controller	0x43520000	Normal	2	Comm	2
	Rx Assembly	0x41000600	Normal	2	Default	0
	Tx Assembly	0x41000700	On	1	Default	0
	CCSDS-Board Ass.	0x41100000	On	1	Active	1
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Comm_Trans_1	0x69010100	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Communic. Controller	0x43520000				
	Rx Assembly	0x41000600	Normal	2	Default	0
	Tx Assembly	0x41000700	On	1	Default	0
	CCSDS-Board Ass.	0x41100000	On	1	Active	1
	Name	**Table ID**	**WaitTime**	**Check?**		
	Comm_Trans_2	0x69010200	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Communic. Controller	0x43520000	Normal	2	Comm	2
	Rx Assembly	0x41000600				
	Tx Assembly	0x41000700				
	CCSDS-Board Ass.	0x41100000				
Commanding to Standby Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Standby	0x69020000	1	0x69000000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	Standby_Target	0x69020000	0	1		

(continued)

Table 6.12 (continued)

	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Communic. Controller	0x43520000				
	Rx Assembly	0x41000600	Normal	2	Default	0
	Tx Assembly	0x41000700				
	CCSDS-Board Ass.	0x41100000				
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Standby_Trans_1	0x69020100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Communic. Controller	0x43520000				
	Rx Assembly	0x41000600	Normal	2	Default	0
	Tx Assembly	0x41000700	Off	0	Off	0
	CCSDS-Board Ass.	0x41100000	On	1	Passive	0
	Name	**Table ID**	**WaitTime**	**Check?**		
	Standby_Trans_2	0x69020200	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	Communic. Controller	0x43520000	Normal	2	Standby	1
	Rx Assembly	0x41000600				
	Tx Assembly	0x41000700				
	CCSDS-Board Ass.	0x41100000				

Table 6.13 Transceiver HK-TM parameters

Name	Description	PTC	PFC	Unit	PID (hex)	Calibration
XNTCLF00	TTC N CTL Lock Flag	3	4		60000000	DSX00012
XNTCLL00	TTC N Loop Lock Flag	3	4		60000100	DSX00012
XNTCRL00	TTC N clock Record Lock Flag	3	4		60000200	DSX00012
XNTCLS00	TTC N carrier Loop Stress Monitor	4	12		60001000	
XNTAGC00	TTC N Signal AGC Monitor	3	12		60001100	
XNTRVC00	TTC N RX VCO monitor	4	12		60001200	
XNTPAC00	TTC N PA current monitor	3	12		60001300	
XNTTLS00	TTC N Tx Loop Stress Monitor	4	12		60001400	
XNTTTM00	TTC N Tx Temp Monitor	5	1	V	60001500	V
XNTRTM00	TTC N Rx Temp Monitor	5	1	V	60001600	V
XNTSCL00	TTC N Subcarrier Loop Stress Monitor	4	12		60001700	
XNTCAG00	TTC N Coherent AGC Monitor	3	12		60001800	
XNTLFR00	TTC N Raw Lock Flag Byte	3	4		60001A00	
XNTAIN00	TTC N Clock Recovery State	3	4		60001B00	
XNTSNR00	TTC N SNR Estimator	4	12		60001C00	
XRTCLF00	TTC R CTL Lock Flag	3	4		62000000	DSX00012
XRTCLL00	TTC R Loop Lock Flag	3	4		62000100	DSX00012
XRTCRL00	TTC R clock Record Lock Flag	3	4		62000200	DSX00012

Table 6.13 (continued)

Name	Description	PTC	PFC	Unit	PID (hex)	Calibration
XRTCLS00	TTC R carrier Loop Stress Monitor	4	12		62001000	
XRTAGC00	TTC R Signal AGC Monitor	3	12		62001100	
XRTRVC00	TTC R RX VCO monitor	4	12		62001200	
XRTPAC00	TTC R PA current monitor	3	12		62001300	
XRTTLS00	TTC R Tx Loop Stress Monitor	4	12		62001400	
XRTTTM00	TTC R Tx Temp Monitor	5	1	V	62001500	V
XRTRTM00	TTC R Rx Temp Monitor	5	1	V	62001600	V
XRTSCL00	TTC R Subcarrier Loop Stress Monitor	4	12		62001700	
XRTCAG00	TTC R Coherent AGC monitor	3	12		62001800	
XRTLFR00	TTC R Raw Lock Flag Byte	3	4		62001A00	
XRTAIN00	TTC R Clock Recovery State	3	4		62001B00	
XRTSNR00	TTC R SNR Estimator	4	12		62001C00	

Table 6.14 Explanatory information on transceiver HK telemetry

Description	TTC TM stream		Explanation
TTC CTL Lock Flag	Byte 0	Bit 0	Carrier lock indication (active low)
TTC Loop Lock Flag		Bit 1	SubCarrier lock indication (active low) NOTE: only valid, when CL is low (active)
TTC Clock Recovery Lock Flag		Bit 2	Data Clock Recovery lock indication (active low)
TTC Carrier Loop Stress Monitor	Byte 1–2	Bit 0–11	Deviation from Carrier LRF (Loop-Rest-Frequency)
TTC Signal AGC Monitor	Byte 3–4	Bit 0–11	Gives good indication of the received signal strength at high (ca. -50 to -100 dBm) signal levels.
TTC RX VCO monitor	Byte 5–6	Bit 0–11	VCO control voltage
TTC TX PA current monitor	Byte 7–8	Bit 0–11	Current flow through the final PA stage
TTC TX Loop Stress Monitor	Byte 9–10	Bit 0–11	VCO control voltage
TTC TX Temp Monitor	Byte 11–12	Bit 0–11	Temperature in the PA compartment
TTC RX Temp Monitor	Byte 13–14	Bit 0–11	Temperature in the Receiver baseband compartment
TTC Subcarrier Loop Stress Monitor	Byte 15–16	Bit 0–11	Deviation from Subcarrier LRF (Loop-Rest-Frequency)
TTC SNR Estimator	Byte 17–18		Gives an indication of the SNR which is estimated on board
TTC Carrier AGC Monitor	Byte 19–20	Bit 0–11	AGC voltage of 2nd mixer stage. Gives good indication of the received signal strength at low (< -100 dBm) signal levels
TTC Clock Recovery State	Byte 20	Bit 4–7	Debug info for supplier
TTC Raw Lock Flag Byte	Byte 0	entire Byte	Debug info for supplier

Table 6.15 CCSDS-Board hardware TM parameters

Name	Description	PTC	PFC	Unit	PID (hex)	Calibration
DCTAC000	CCSDS Board 0 Count of accepted Codeblocks (TC)	3	4		20000000	
DCTSC000	CCSDS Board 0 Count of single error corrections (TC)	3	4		20000100	
DCTCI000	CCSDS Board 0 Selected Channel Input (TC)	3	4		20000200	
DCTRF000	CCSDS Board 0 Physical RFA (TC)	3	4		20000300	
DCTBL000	CCSDS Board 0 Physical BLO (TC)	3	4		20000400	
DCTBF000	CCSDS Board 0 Rx Buffer Full (TC)	3	4		20000500	DSX00012
DCTFF000	CCSDS Board 0 Rx FIFO Full (TC)	3	4		20000600	DSX00012
DCTOV000	CCSDS Board 0 Overrun (TC)	3	4		20000700	DSX00012
DCTCR000	CCSDS Board 0 CLTU Ready (TC)	3	4		20000800	DSX00012
DCTTR000	CCSDS Board 0 Transmitter Ready (TM)	3	4		20000900	DSX00012
DCTAC100	CCSDS Board 1 Count of accepted Codeblocks (TC)	3	4		20010000	
DCTSC100	CCSDS Board 1 Count of single error corrections (TC)	3	4		20010100	
DCTCI100	CCSDS Board 1 Selected Channel Input (TC)	3	4		20010200	
DCTRF100	CCSDS Board 1 Physical RFA (TC)	3	4		20010300	
DCTBL100	CCSDS Board 1 Physical BLO (TC)	3	4		20010400	
DCTBF100	CCSDS Board 1 Rx Buffer Full (TC)	3	4		20010500	DSX00012
DCTFF100	CCSDS Board 1 Rx FIFO Full (TC)	3	4		20010600	DSX00012
DCTOV100	CCSDS Board 1 Overrun (TC)	3	4		20010700	DSX00012

(continued)

Table 6.15 (continued)

Name	Description	PTC	PFC	Unit	PID (hex)	Calibration
DCTCR100	CCSDS Board 1 CLTU Ready (TC)	3	4		20010800	DSX00012
DCTTR100	CCSDS Board 1 Transmitter Ready (TM)	3	4		20010900	DSX00012

For the Parameter Type Codes and Format Codes please refer to Table 16.2.

Table 6.16 Transceiver TM parameter decalibration

Name	Description	Value from	Value to	Calibrated Text
DSX00012	True/False Calibration	0	0	False
DSX00012	True/False Calibration	1	1	True

The TTC subsystem relevant TC packet definitions are included in the spacecraft TC table in annex Sect. 17.3.1.

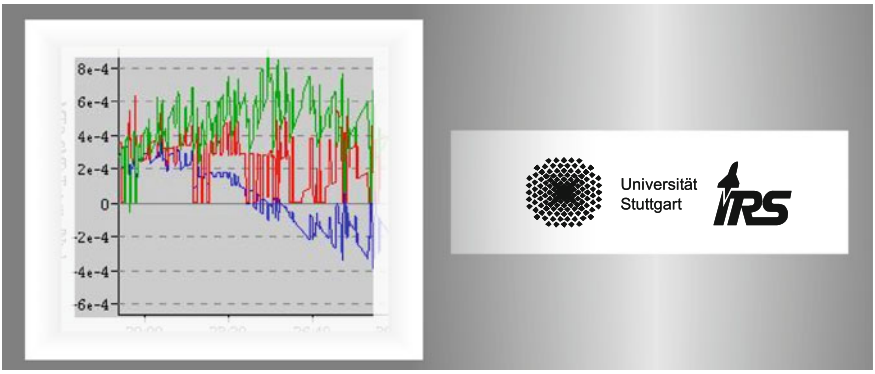
The TTC subsystem relevant TM packet definitions are included in the spacecraft TM table in annex Sect. 17.3.2.

The TTC subsystem relevant Event TM packets are included in the spacecraft Event TM table in annex Sect. 17.3.3.

Details on the TM parameter positions in the diverse packets have to be taken from the MIB.

Chapter 7
Attitude Control Subsystem

Oliver Zeile, Ulrich Mohr, Bastian Bätz and Nico Bucher



Abstract The chapter provides an overview on the Attitude Control Subsystem (ACS). Then it treats all the ACS equipment as far as relevant for spacecraft operations, also providing the mounting vectors/matrices. The option of extension with a propulsion subsystem is sketched out at hand of the Airbus Astrobus-XS (Myriade) propulsion equipment. The last part comprises the onboard software elements and functions for control of the subsystem, treating all the device handlers, assemblies and the ACS controller itself. This includes all the ACS modes and mode transitions. An example orbit analysis is included in the book's annexes.

O. Zeile (✉) · U. Mohr · B. Bätz · N. Bucher
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: zeile@irs.uni-stuttgart.de

U. Mohr
e-mail: mohr@irs.uni-stuttgart.de

B. Bätz
e-mail: baetz@irs.uni-stuttgart.de

N. Bucher
e-mail: bucher@irs.uni-stuttgart.de

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_7

Keywords ACS subsystem modes · ACS subsystem equipment and alignment · Propulsion subsystem as optional extension · ACS subsystem control

7.1 Subsystem Overview

The FLP attitude control subsystem (ACS) is a three axis stabilized control system. The structure and design of the attitude control system is prescribed by the performance requirements imposed by the mission specific payload experiments or observations and by the general constraints for work during different mission phases, such as from launch to nominal operations. The requirements supported by the FLP ACS through equipment and operational subsystem Modes are as follows:

1. Rate damping after the launcher separation or during operation in case of accidental increase of body rates.
2. To provide a Safe Mode to ensure the survival of satellite by using only very reliable components during the contingency situation.
3. To provide a house keeping mode to orient the solar panels towards the sun during the stand-by periods.
4. To provide a simplified nadir pointing mode w/o use of the RWs for safe platform commissioning of the ACS in orbit.
5. Pointing the satellite towards nadir, a target on the surface of the Earth or towards some inertial target to carry out scientific experiments.
6. Pointing knowledge better than 7 arc seconds (2σ) and absolute pointing accuracy better than 150 arc seconds (2σ) for the target pointing mode.
7. Proper sun avoidance strategy to prevent blinding of Earth oriented payload instruments.
8. To avoid blinding of star trackers by Sun, Earth or Moon.
9. Usability of the FLP platform for a range of orbits.
10. Fault detection isolation and recovery (FDIR) for the ACS components and subsystem.

The requirements on ACS clearly demand the three axis stabilized system and high accuracy requirements compel towards the use of fine pointing devices like star trackers. Five different kinds of sensors are used to observe the motion of the satellite and two different kinds of actuators are used to control its motion. The sensor and actuator types of the FLP ACS have already been mentioned in Chap. 1. The detailed list is given here below:

- 4 fiberoptic gyroscopes (FOG) in tetrahedron setup with a common FOG assembly electronic unit
- 2 star trackers heads with a common Data Processing Unit (DPU)
- 2 magnetometers (MGM)—each measuring all 3 axes—with a common MGM electronics unit

- One internally dual redundant GPS receiver (i.e. 3 receivers)
- 8 single redundant analog Sun sensors (16 sun sensor solar cells)
- 3 magnetotorquers (MGT) with coaxial coils for redundancy and a common MGT electronic unit
- 4 reaction wheels (RW) in tetrahedron setup with a common RW assembly electronic unit

When the satellite is in a pointing mode, the MGT electronics is powered but the coils themselves are deactivated most of the time. Low torques for precise fine pointing are applied by means of using the Reaction Wheels (RW). If the reaction wheels run into saturation, the MGTs are used to desaturate the RWs.

The data interfaces of the actuators and sensors (except for Sun sensors) are connected to the I/O-Board of the OBC. The power supply interfaces (including Sun sensor measurement interfaces) to the PCDU.

7.2 Mission Objectives and ACS Subsystem Modes

The satellite system operational modes have already been cited in Sect. 1.10.2 and the ACS control modes correspond to the spacecraft modes—see Fig. 2.2.

The nominal spacecraft system modes like Target Pointing Mode or Nadir Pointing mode can be split into submodes according to the different operational tasks like ground contact, EO area observation with imager (Nadir Pointing) or EO target observation with camera.

For this reason also the descriptions in Sect. 2.1.12 and Table 17.11 Power consumption versus spacecraft modes in annex Sect. 17.7.2 show specific submodes of the first FLP mission—"Flying Laptop"—as illustration examples.

7.3 Magnetometers

FLP based satellites are equipped with two three axis anisotropic-magnetoresistive (AMR) magnetometers manufactured by ZARM-Technik AG, Bremen, Germany [67]. The measured vector of Earth magnetic field is used as input information for magnetic torquer control by the ACS for Detumble Mode and Safe Mode.

The magnetometers as shown in Fig. 7.1 are micro-controller based magnetometers with a digital, serial RS-422 output. They measure the external magnetic field with the sensitivity of 8.5 nT, the measured power consumption of each module is 0.2 W and its mass including the 2–3 mm thick aluminum housing is 50 g. Figure 7.2 illustrates the orientation of the two magnetometers on the FLP platform.

The transformation matrices to FLP body coordinates are given in Equations below (for details refer to [124]):

Fig. 7.1 Magnetometer

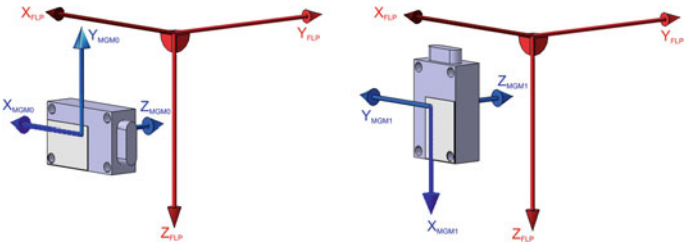


Fig. 7.2 Orientation of MGM.0 and MGM-1 in the platform. © IRS, University of Stuttgart

$$\vec{A}_{MGM0} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} \tag{7.1}$$

$$\vec{A}_{MGM1} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{7.2}$$

7.4 Sun Sensor Unit

The design of the Sun sensor unit is based on two GaAs solar cell on a small printed circuit board (PCB) to make it hot redundant by design. Eight such Sun sensor units are placed on the FLP body and are utilized to construct the overall 4π view.

Figure 7.3 shows the block diagram of the Sun sensor design and Fig. 7.4 illustrates an example for the arrangement of Sun sensors on an FLP satellite—depicting the first FLP mission "Flying Laptop".

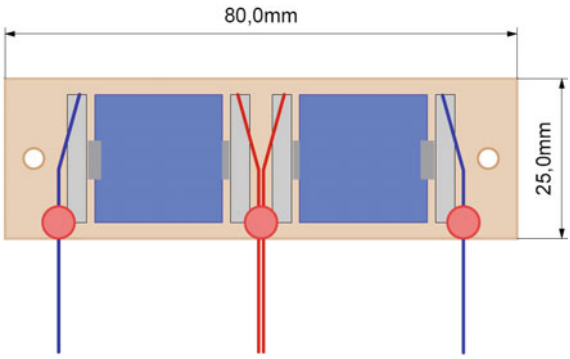


Fig. 7.3 Sun sensor unit diagram. © IRS, University of Stuttgart

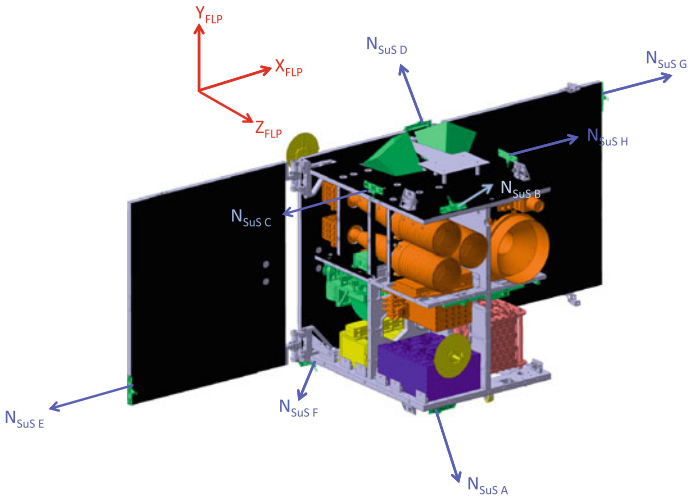


Fig. 7.4 Orientation of sun sensor units on an FLP spacecraft. © IRS, University of Stuttgart

An analog/digital converter (ADC) within the PCDU converts the analog signal of the cells into a digital signal that is provided within the PCDU HK data packet to the OBC/OBSW.

This results in the following Sun sensor orientation vectors (for details refer to [124]):

$$\vec{n}_{SuS-A} = \begin{bmatrix} 0 \\ \frac{-\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{bmatrix} \tag{7.3}$$

$$\vec{n}_{SuS-B} = \begin{bmatrix} 0 \\ \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{bmatrix} \tag{7.4}$$

$$\vec{n}_{SuS-C} = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \tag{7.5}$$

$$\vec{n}_{SuS-D} = \begin{bmatrix} 0 \\ \frac{\sqrt{2}}{2} \\ \frac{-\sqrt{2}}{2} \end{bmatrix} \tag{7.6}$$

$$\vec{n}_{SuS-E} = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \tag{7.7}$$

$$\vec{n}_{SuS-F} = \begin{bmatrix} 0 \\ \frac{-\sqrt{2}}{2} \\ \frac{-\sqrt{2}}{2} \end{bmatrix} \tag{7.8}$$

$$\vec{n}_{SuS-G} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{7.9}$$

$$\vec{n}_{SuS-H} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{7.10}$$

7.5 GPS Receiver System

The GPS receiver system is internally dual redundant and running in hot redundancy. It consists of 3 independent GPS receiver boards, each connected to a separate antenna via a separate low noise amplifier as depicted in Fig. 7.5.

The selected configuration is a requirement for an experiment which will be conducted on the first FLP mission in cooperation with the German Space Agency (DLR/GSOC) for GPS based attitude determination.

The DLR Phoenix receiver boards [63] are used as miniature low-power 12-channel GPS L1 receivers based on a SigTech commercial MG5001 OEM receiver board, with DLR/GSOC firmware which was developed for space and high dynamic applications. Single DLR Phoenix receivers already have flight heritage.

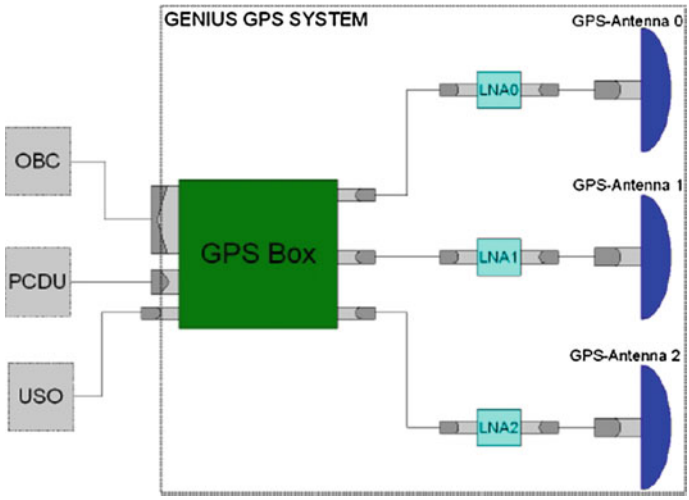


Fig. 7.5 GPS system. © IRS, University of Stuttgart

7.6 Fiberoptic Gyroscopes

Four fiberoptic gyroscopes (FOG) are used on board the FLP in a tetrahedron configuration for the measurement of the FLP body angular rates. The tetrahedron configuration results in a 3-out-of-4 redundancy by design.

The sensors of the first FLP spacecraft are of type LITEF C-FORS (Commercial Fiber Optic Rate Sensor)—see Fig. 7.6 and [64]. As specified by the manufacturer the C-FORS has a a maximum rate bias of 27°/h over −40 to +77 °C and a maximum angular random walk of $0.15°/\sqrt{h}$.

Fig. 7.6 FOG unit C-FORS

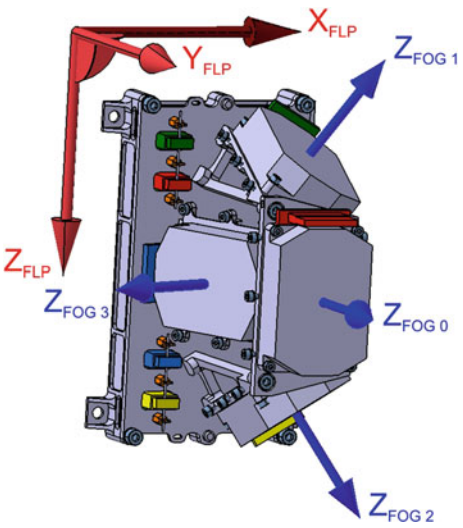


Fig. 7.7 Alignment of the FoGs

The alignment vectors to FLP boby coordinates are giver in equations below (for details refer to [124]) (Fig. 7.7):

$$\vec{z}_{FOG0} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{7.11}$$

$$\vec{z}_{FOG1} = \begin{bmatrix} 0.4741 \\ 0.3333 \\ -0.8165 \end{bmatrix} \tag{7.12}$$

$$\vec{z}_{FOG2} = \begin{bmatrix} 0.4741 \\ 0.3333 \\ 0.8165 \end{bmatrix} \tag{7.13}$$

$$\vec{z}_{FOG3} = \begin{bmatrix} -0.9428 \\ 0.3333 \\ 0 \end{bmatrix} \tag{7.14}$$

The alignment matrix results to:

$$\overline{A}_{FOG} = [\,\vec{z}_{FOG0} \quad \vec{z}_{FOG1} \quad \vec{z}_{FOG2} \quad \vec{z}_{FOG3}\,] \tag{7.15}$$

The C-FORS gyroscopes are the commercial variant and were selected for the first FLP based mission as a university satellite simply for cost reasons. They have limited aging stability (see also Sect. 10.6.5).

For commercial missions and for FLP Generation 2, Airbus DS foresees to use the LITEF μFORS-6U as also proposed in [144].

7.7 Star Tracker

The star tracker system as shown in Fig. 7.8 consists of a micro data processing unit and two camera head units (CHU) with a baffle. The selected model is the micro-Advances Stellar Compass (μASC) developed by the Technical University of Denmark (DTU). See also [61, 62].

The CHU uses a CCD-chip with a size of 7.95 × 6.45 mm^2 with 752 × 558 pixels to take the image of the stars. With the used optics the field of view (FOV) results to 13.4° × 18.4° and its diagonal covers 22.76° (Fig. 7.9).

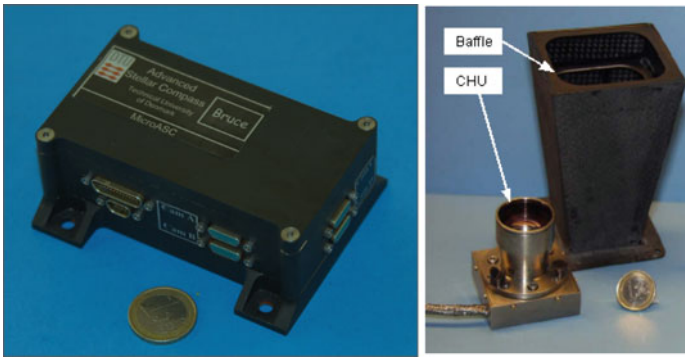


Fig. 7.8 Star tracker system

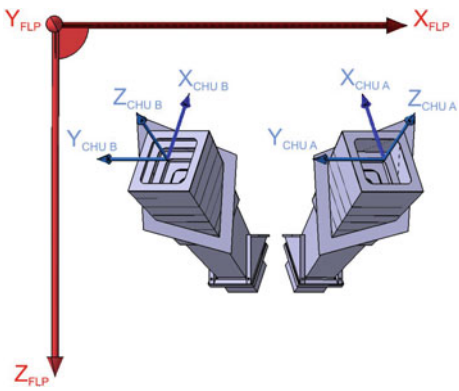


Fig. 7.9 Alignment of the STR camera head units viewed along y_{FLP} axis

The transformation matrices to FLP boby coordinates are given in the equation below (for details refer to [124]):

$$\bar{A}_{CHUA} = \begin{bmatrix} -0.2693 & -0.8829 & 0.3846 \\ -0.5064 & 0.4695 & 0.7233 \\ -0.8192 & 0 & -0.5736 \end{bmatrix} \tag{7.16}$$

$$\overline{\bar{A}}_{CHUB} = \begin{bmatrix} 0.2693 & -0.8829 & 0.3846 \\ -0.5064 & 0.4695 & 0.7233 \\ -0.8192 & 0 & -0.5736 \end{bmatrix} \tag{7.17}$$

7.8 Reaction Wheels

Four reactions wheels (RW) are integrated on board the FLP in a tetrahedron configuration [65]. The reaction wheel of the FLP Generation 1 is a Rockwell Collins type RSI 01-5/28 (see Fig. 7.10 and [68]). It has a mass of 0.7 kg and an angular momentum capacity of 0.12 Nms and a reaction torque of 5 mNm over the range of ±3000 rpm.

Each RW is connected via an FPGA-based RW electronics in a star like configuration via RS-422 bidirectional interface to allow the parallel executions of the ACS commands [65].

The alignment vectors of the RWs in the tetrahedron assembly (see Fig. 7.11) are by default the same as for the FOGs. This however my vary for each mission. For the mission discussed in [144] a more flat assembly (>60°) was selected as higher agility was required around one particular spacecraft axis.

For the FLP Generation 2, Airbus DS foresees an RW type with greater torque performance and allowing to eliminate the RW assembly electronics by direct RW connectivity to the OBC and simpler control by OBSW. The selected type is the RW-90 from Astro- und Feinwerktechnik GmbH, Berlin [69].

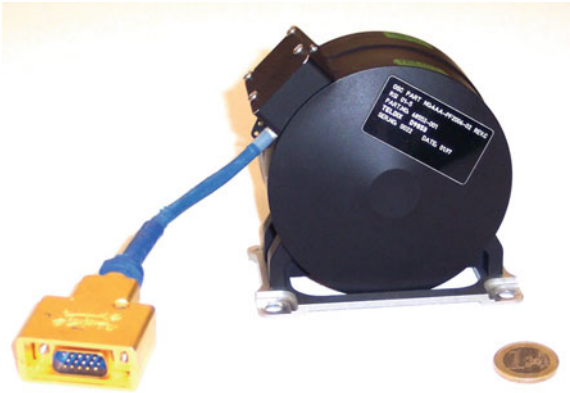


Fig. 7.10 Reaction wheel

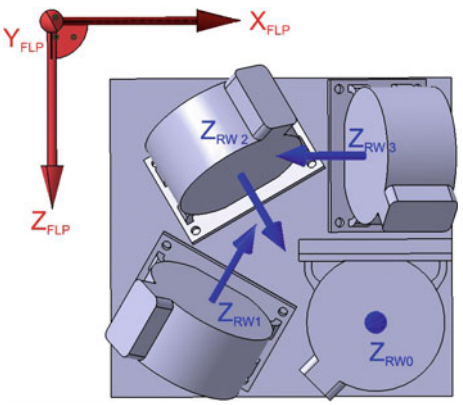


Fig. 7.11 RW tetrahedron assembly. © IRS, University of Stuttgart

7.9 Magnetotorquers

Three magnetic torquers with a linear dipole moment of 6 Am^2 are used as an actuator in Detumble and Safe Mode and during normal operation they dump the momentum accumulated by the reactions wheels. The magnetic torquers are also provided by the ZARM-Technik (see Fig. 7.12). Each torquer consists of two separate coils wound parallel on the top of each other. Both can be operated independent of each other providing redundancy.

The magnetotorquers are aligned along the major body axes of the FLP structure (Fig. 7.13).

To apply a torque onto the spacecraft the coils are activated by commanding the MGT electronics from the OBC. They only offer the states “active” and “not-active”, the current is not regulated. Control is achieved by Pulse Width Modulation (PWM). When the satellite is in a pointing mode, the MGT electronics is powered but the coils themselves are deactivated most of the time. Figure 7.14 shows the electrical interaction between OBC, PCDU and MGT electronics and coils. Low torques for precise fine pointing are applied by means of using the Reaction Wheels (RW).

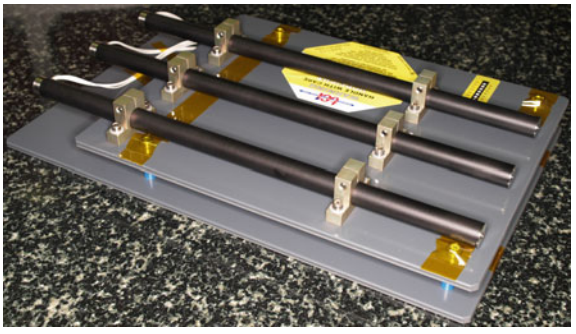


Fig. 7.12 Magnetotorquers

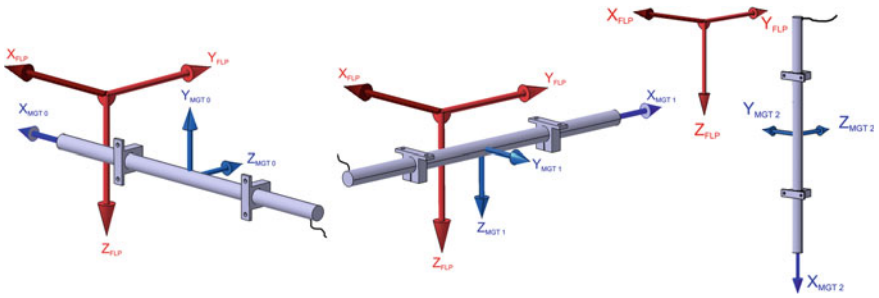


Fig. 7.13 Magnetotorquer alignment. © IRS, University of Stuttgart

$$\vec{x}_{MGT0} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{7.18}$$

$$\vec{x}_{MGT1} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{7.19}$$

$$\vec{x}_{MGT2} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{7.20}$$

The PCDU contains the switch to provide power to the MGT electronics for side 0 and redundant side 1. IF the ACS algorithm in the OBSW requires torque to be generated, the MGT electronics is commanded to activate the coils for the computed amount of time.

The MGT electronics 0 and 1, and the OBC are connected in the FLP Generation 1 platform design via two separate I^2C interfaces. An improved MGT Electronics with less critical interface is foreseen for FLP Generation 2.

7.10 Extensions for FLP Generation 2

Some enhancements of the ACS for the FLP Generation 2 such as better FOGs and more performant RWs already have been mentioned.

Furthermore the ACS can be enhanced by a digital Earth sensor for improved attitude precision in Safe Mode. This also allows simpler missions with reduced

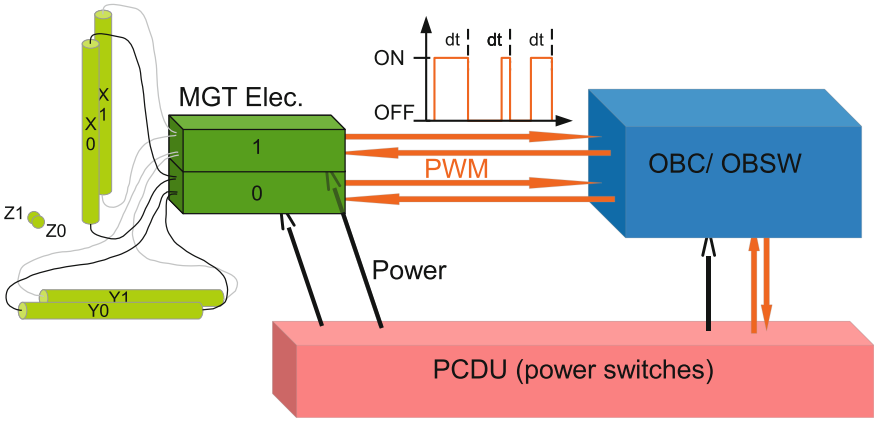


Fig. 7.14 MGT electronic architecture and control. © IRS, University of Stuttgart

pointing requirements to fly with Earth and Sun sensors only and without star trackers.

Another enhancement—already mentioned in Sect. 1.11—is a Reaction Control System (RCS) for orbit correction maneuvers as it is studied by Airbus DS in [144] (hydrazine monopropellant) or in [154] (Xenon resistojet). An example propulsion module which is compatible with the FLP Generation 2 OBC architecture ([146] and Sect. 9.3.2) is depicted in Fig. 7.15.

These however are not yet on board in the first FLP based mission and are not detailed yet in this Flight Operations Manual.

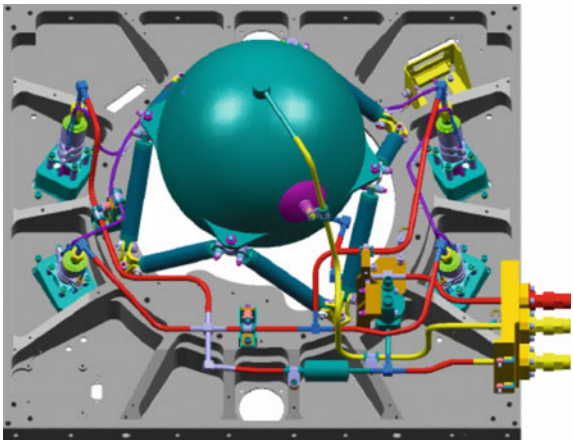


Fig. 7.15 Hydrazine monopropellant propulsion module—from [159]. © Airbus DS

7.11 ACS Subsystem Control

The ACS subsystem is represented in the OBSW as a collection of device handler and assembly objects, as well as one controller object. The subsystem itself is represented by the *ACS_Subsystem* object.

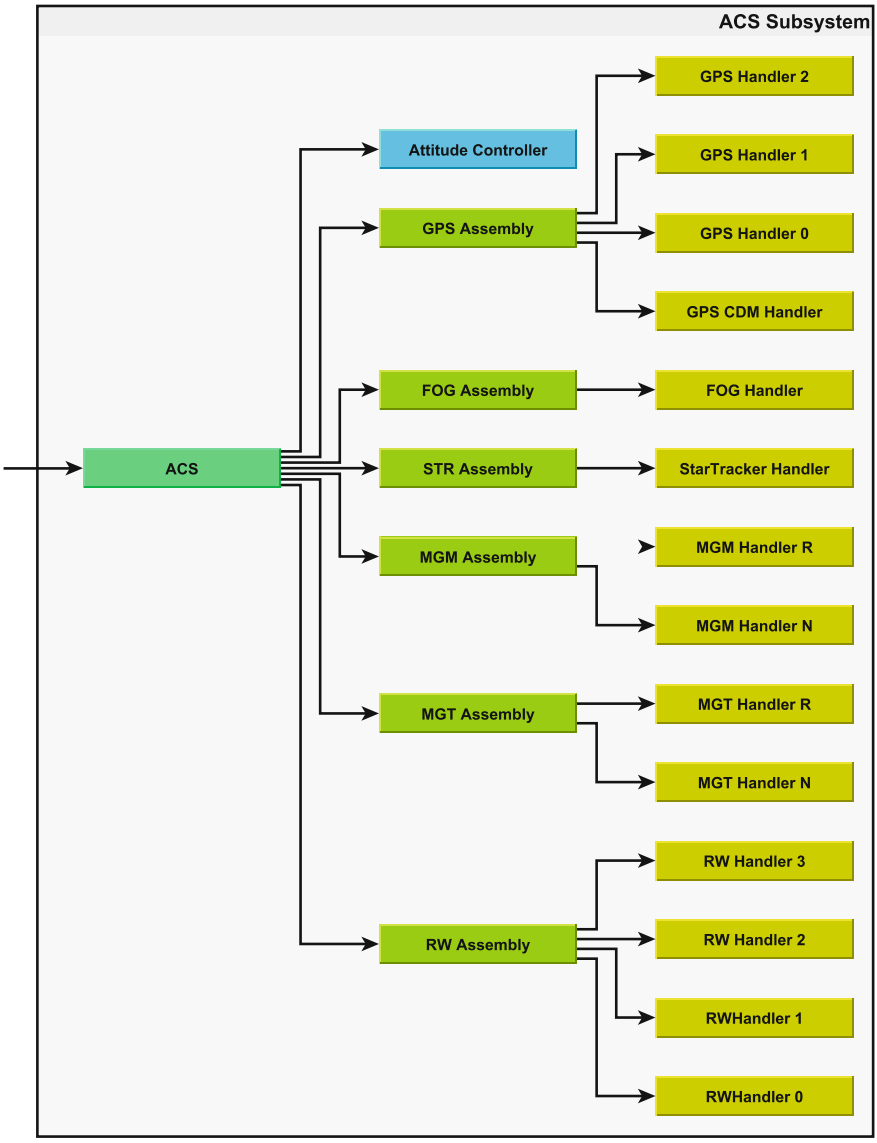


Fig. 7.16 Attitude control subsystem overview. © IRS, University of Stuttgart

As shown in Fig. 7.16, the ACS subsystem objects can be layered hierarchically. It should be noted that the layering is only with respect to the mode commanding of the subsystem—not w.r.t. object-oriented inheritance. The objects are all system objects on their own and can also be commanded individually by commanding mode changes or by sending direct commands for the device handlers. The layers are there to allow commanding of a subset or the complete subsystem with only one command, which will be hierarchically forwarded to the lower layers. It also allows reacting to FDIR events, as lower layers can handle redundant devices without need to notify the higher layers.

The lowest layer is formed by the device handlers, which are described in Sect. 7.11.2. The device handlers are grouped by assembly objects, described in Sect. 7.11.3. On the same layer as the assemblies is the *ACS_Controller* which encapsulates the control algorithms. The *ACS_Controller* is described in Sect. 7.11.4.

7.11.1 ACS Subsystem Modes

The *ACS_Subsystem* modes including all subobjects and their modes and submodes are included in Table 7.1.

7.11.2 ACS Device Handlers

In this section, those device handlers which are part of the ACS are described in detail. The device handlers adhere to the concepts outlined in Sect. 3.2.2.

7.11.2.1 FOG Device Handler

There is only one device handler handling all four FOG units. This results from the FOGs being connected to a common bus and only having one interface on the I/O-Board. Details on the FOG units can be found in [64]. The FOG is a simple instrument, only returning rotational rate measurement values upon request. There are no different modes or additional telemetry and commands available.

Modes and Submodes
When commanded to On from Off, the FOG units are activated. No further commands are sent. On mode is entered immediately after all switches are on. The transition from Off to On takes 4 s at maximum.

There are no submodes defined. Instead, the submode is used as a deactivation mask. The bits 0–4 are used to deactivate FOG unit 0–4. If the corresponding bit is 0 the FOG is used, if it is 1, the FOG is switched to Off or not turned on.

Table 7.1 ACS subsystem modes

Subsystem Mode	**Off**	**0x19000000**		
	Mode		**Submode**	
ACS Controller	Off	0	Off	0
MGT Ass.	Off	0	Off	0
MGM Ass.	Off	0	Off	0
RW Ass.	Off	0	Off	0
STR Ass.	Off	0	Off	0
FOG Ass.	Off	0	Off	0
GPS Ass.	Off	0	Off	0
Subsystem Mode	**Safe**	**0x19020000**		
	Mode		**Submode**	
ACS Controller	Normal	2	Safe	2
MGT Ass.	Normal	2	Multi	1
MGM Ass.	Normal	2	Fast	2
RW Ass.	Off	0	Off	0
STR Ass.	Off	0	Off	0
FOG Ass.	Off	0	Off	0
GPS Ass.	Off	0	Off	0
Subsystem Mode	**CoarseNadir**	**0x19030000**		
	Mode		**Submode**	
ACS Controller	Normal	2	CoarseNadir	3
MGT Ass.	Normal	2	Single	0
MGM Ass.	Normal	2	Fast	2
RW Ass.	Off	0	Off	0
STR Ass.	Normal	2	Default	7
FOG Ass.	Off	0	Off	0
GPS Ass.	On	1	Single	0
Subsystem Mode	**Idle**	**0x19040000**		
	Mode		**Submode**	
ACS Controller	Normal	2	Idle	4
MGT Ass.	Normal	2	Single	0
MGM Ass.	Normal	2	Fast	2
RW Ass.	Normal	2	Torque	1
STR Ass.	Normal	2	Default	7
FOG Ass.	Normal	2	Default	0
GPS Ass.	On	1	Single	0
Subsystem Mode	**NadirPt**	**0x19050000**		
	Mode		**Submode**	
ACS Controller	Normal	2	NadirPt	5
MGT Ass.	Normal	2	Single	0
MGM Ass.	Normal	2	Fast	2

(continued)

Table 7.1 (continued)

Subsystem Mode	**Off**	**0x19000000**		
RW Ass.	Normal	2	Torque	1
STR Ass.	Normal	2	Default	7
FOG Ass.	Normal	2	Default	0
GPS Ass.	On	1	Single	0
Subsystem Mode	**TargetPt**	**0x19060000**		
	Mode		**Submode**	
ACS Controller	Normal	2	TargetPt	6
MGT Ass.	Normal	2	Single	0
MGM Ass.	Normal	2	Fast	2
RW Ass.	Normal	2	Torque	1
STR Ass.	Normal	2	Default	7
FOG Ass.	Normal	2	Default	0
GPS Ass.	On	1	Single	0
Subsystem Mode	**InertialPt**	**0x19080000**		
	Mode		**Submode**	
ACS Controller	Normal	2	InertialPt	7
MGT Ass.	Normal	2	Single	0
MGM Ass.	Normal	2	Fast	2
RW Ass.	Normal	2	Torque	1
STR Ass.	Normal	2	NEO	7
FOG Ass.	Normal	2	Default	0
GPS Ass.	On	1	Single	0

Table 7.2 FOG device handler modes

Handler	**Mode**	**M#**	**Submode**	**SM#**	**Description**
FOG	Off	0	Default	0	Off
	On	1	*Bit Mask*	0–15	Bit mask in bits 0–3 (0 on, 1 off), where FOG 0 is the least significant bit
	Normal	2	*Bit Mask*	0–15	Bit mask in bits 0–3 (0 on, 1 off), where FOG 0 is the least significant bit
	Raw	3	Default	0	Raw Mode unchanged from before

For example, if the bit pattern 00001101 is used, FOG 1 is on and the other FOGs are Off (compare to Table 7.2). This mask can be used in On, Raw and Normal mode. The Off mode only has the default submode 0x00.

When On, Raw and Normal mode are commanded with a submode unequal to zero, the deactivated FOG units are switched off before entering the commanded mode.

TM/TC Interface
A single TM parameter containing the FOG rate measurements is used. Also, a single Service (2,130) direct TC to request the measurement data exists. The command affects all four FOG units.

7.11.2.2 STR Device Handler

The Star Tracker is a sophisticated device, as it is commanded and responds in a subset of PUS services. These are:

- Service 1: Telecommand Verification Service
- Service 3: Housekeeping Service ((3,25) only)
- Service 5: Event Reporting Service
- Service 6: Memory Management Service
- Service 8: Function Management Service
- Service 9: Time Management Service

Parts of the commands and responses are created and interpreted on-board.

Unknown or unexpected responses are always forwarded to ground unchanged (with the STR's APID). Details on commanding the STR can be found in [62].

As described in Sect. 7.7, the STR unit consists of micro data processing unit (DPU) and two camera head units (CHU). The DPU is a single unit but provides internal redundancy. Each CHU delivers an independent attitude solution to the OBC. In addition, one CHU is equipped with a "micro inertial reference unit" (MIRU) as an experiment. This unit, together with additional software features in the DPU provide an additional attitude solution, which is augmented with data from an internal reference unit.

Modes and Submodes
Despite this complex device protocol, the STR device handler is a default device handler object as described in Sect. 3.2.2. Particularly, it adheres to the default device mode concept with Off, On and Normal modes. The STR device handler modes are listed in Table 7.3:

The STR can start up in three different modes:

- Safe-Boot Up
- Standby Mode
- Attitude Mode

To have a clear starting point, the handler always boots up the device in Safe-Boot-Up Mode. The procedure is triggered when going from mode Off to mode On, but also when going form mode NORMAL back to On (because a reboot is the only way to come back to Safe-Boot Up Mode). During the transition to On, certain parameters are set. These are

Table 7.3 STR device handler modes

Handler	Mode	M#	Submode	SM#	Description
STR	Off	0	Default	0	Off
	On	1	Default	0	Safe boot-up mode
	Normal	2	*Bit Mask*	0–127	Complicated bit mask with many settings. See Table 7.4 for details. The idea is that the submode 0 can be used as default mode
	Raw	3	Default	0	No changes from previous state

Table 7.4 STR Submode bit mask

Bit	7	6	5	4	3	2	1	0
Usage	–	Auto Imaging on	DPU Mode		MIRU on	Time update off	CHU-B off	CHU-A off
Values	Must be 0	1: on 0: off	00 OR 10: Attitude 01: Standby mode 11: Simulation mode		1: on 0: off	1: off 0: on	1: off 0: on	1: off 0: on

- The HK interval
- The image integration time
- Power settings for both CHU's. These are turned off.

There are a number of Normal modes. There is a bit mask for submodes which is shown in Table 7.4. Almost all of these submodes are valid and may be useful. These submodes can only be used in mode Normal. All other modes only have the default submode 0x00.

If the time update bit (bit 2) of the submode is set to On, the time of the STR PPS pulse is recorded and sent to the star tracker every few seconds. The interval is settable as STR device handler configuration parameter.

If an Off command is issued, the device is simply turned off.

Being a PUS-capable device, the STR varies the general device concept for Raw mode. When switching to Raw mode, the Handler starts reading a dedicated message queue that receives commands routed directly to its APID. Any received telemetry is immediately inserted in the normal downlink stream (with STR APID —see Table 2.6). However, a length check is performed to ensure packets are well-formed. If a corrupted packet is found, it is downlinked by making use of the default (2,129) service. The data content does not have any structure then. The STR EGSE can be used to display raw STR packets on ground.

The handler **does not** check any uplinked commands. Therefore, any command (including ill-formed) can be submitted to the star tracker.

To switch on the STR, there are two redundant switches in the PCDU, each of which can be used to power the STR. There is a private pool variable defining

which switch is used, which can be set by a memory load command. It is however only possible to set this variable when the STR handler is in Off mode to avoid problems during the handler transition to Off mode.

TM/TC Interface
Direct commands are sent to the STR by the standard Service (2,130). The handler forwards any direct command to the device, with the exception of device mode commands, as they would jeopardize the mode synchronization between handler and device.

In the attitude information and the housekeeping packets, the STR delivers the following data:

- Attitude information (one from each CHU)
- Augmented attitude information (from the MIRU experiment)
- DPU and CHU temperatures
- CHU voltages

The attitude information is used in the ACS subsystem. Temperature data are processed in the TCS system.

The attitude information has the form of quaternions, but also additional information like the number of detected stars, the average time the image was taken and additional information flags. The handler extracts this information and stores them in the *datapool*.

STR telemetry, which is unknown to the device handler (such as image data), will be put to the telemetry stream unchanged—i.e. with STR APID and packet header structure.

Image Forwarding
For the FLP platform, the I/O-Board is not capable of handling a constant stream of data from the STR device. Such a stream is created if large amounts of data (such as a full raw image) are requested from the device.

In such cases, the OBC will not receive any data from the STR. Therefore, such conditions must be avoided, i.e. operators should never request a full image as a whole block, only in segments.

7.11.2.3 MGM Device Handler

There are two MGM device handlers, one for the nominal and one for the redundant MGM. For details on the MGM commanding, see [67].

When commanded to On, the MGM is put into standby mode. The transition from Off to On takes 10.6 s at maximum.

When commanded to Raw mode, no commands are sent, i.e. the MGM mode stays the same. Thus, if the MGM was in standby mode at the time of the mode change, a power up TC needs to be sent first to get the MGM into Normal mode. If the handler is set to normal mode, the MGM is switched into its Normal mode and TM is requested regularly.

Table 7.5 MGM device handler modes

Handler	Mode	M#	Submode	SM#	Description
MGM N/R	Off	0	Default	0	Off
	On	1	Default	0	MGM in standby mode
	Normal	2	Slow Sampling	0	MGM Slow Sampling
			Medium Sampling	1	MGM Medium Sampling
			Fast Sampling	2	MGM Fast Sampling
	Raw	3	Default	0	Raw Mode unchanged from before

There are three submodes for Normal mode. Slow (0) Medium (1) and Fast (2), corresponding to slow, medium and fast sampling (see [67]). All other modes only offer the default submode 0x00.

Modes and Submodes

The MGM device handler modes are listed in Table 7.5. When commanded to On, the MGM is put into standby mode. The transition from Off to On takes 10.6 s at maximum.

When commanded to Raw mode, no commands are sent, i.e. the MGM mode stays the same. Thus, if the MGM was in standby mode at the time of the mode change, a power up TC needs to be sent first to get the MGM into Normal mode. If the handler is set to Normal mode, the MGM is switched into its Normal mode and TM is requested regularly.

There are three submodes for Normal mode. Slow (0) Medium (1) and Fast (2), corresponding to slow, medium and fast sampling (see [67]). All other modes only offer the default submode 0x00.

TM/TC Interface

For the MGM, there are TM parameters containing the measured magnetic field by the two MGMs and their measured temperature. Service (2,130) direct commands include requests for measurement data in the three sampling rates and power up and down commands. It needs to be noted that due to a bug in the MGM firmware, the MGM power up TC acts as a toggle command, i.e. if the MGM is in Normal mode and the power up command is received, it is set into standby mode. Using the power down TC will always put the MGM into standby mode.

7.11.2.4 GPS Device Handler

There are three GPS device handlers, one for each GPS unit. Details on the GPS receiver operations can be found in [63]. The GPS receivers are complex instruments offering different modes and allowing a range of settings to be configured by the user. However, most of the time, the standard settings can be left unchanged.

The GPS receivers return navigation solution TM (position, velocity, time) in the so-called F40 message and further information on how many GPS satellites are visible and which channels are used in the F62 message. These two are the main TM packets which are regularly polled from the receivers by the OBSW.

The GPS receivers offer a filtered navigation solution through a functionality named Extended Navigation Solution (XNS), whose quality may be better than that of the simple solution. The receivers must be commanded to a certain mode in order to use this functionality. If switched on, the XNS is returned in the F80 message.

There is a Pulse Per Second line from each GPS receivers to the OBC Processor-Board. By this line, the GPS signals the exact GPS clock seconds. This line can be used to synchronize the on-board time to GPS with a very high accuracy.

Modes and Submodes

The GPS device handler modes are listed in Table 7.6. The GPS Handler does not offer a Normal mode. Instead, in On mode messages from the GPS receiver are received and parsed by the OBSW. This is in violation of the general device handling concept, which defines the ON mode as a passive mode without any communications. However, as the GPS provides data whenever it is powered, distinguishing between On and Normal modes would not increase operability.

The GPS receivers can be operated in two modes, Aided and Unaided. For Aided mode, they must be supplied with orbit information. This orbit information must either be uplinked from ground before starting the GPS receivers in Aided mode or be already available on-board, e.g. from previous GPS data, when the GPS system was switched off. Aided and Unaided mode are implemented as submodes to the On mode. Whenever an Aided submode is commanded, the handler sets the orbit information and activates the aided mode. Whenever Unaided submode is commanded, the handler disables the aiding mode of the GPS.

The GPS can also be started Aided or Unaided. This is selected by either commanding mode on, submode Aided or mode on, submode Unaided when the GPS handler is in mode off.

Table 7.6 GPS device handler modes

Handler	Mode	M#	Submode	SM#	Description
GPS 0–2	Off	0	Default	0	Off
	On	1	Unaided	0	Unaided Boot
			Aided	1	Aided Boot
			Unaided XNS	2	Unaided Boot with XNS
			Aided XNS	3	Aided Boot with XNS
	Raw	3	Default	0	Raw Mode unchanged from before
GPS CDM	On	1	Internal	0	Internal Clock used in every GPS
			CDM	1	Common Clock used for all GPS

When starting Unaided, the GPS receiver is switched on and the Handler waits until the GPS receiver has accomplished a fix. This will be the standard situation in Launch and Early Orbit Phase (LEOP)—see Sect. 11.6.

When starting Aided, the Handler sets the time of the GPS receiver, transmits almanac and orbit data, sets the tracking mode and doppler window and enables Aided operation.

The GPS Receiver offers an extended Kalman filter which can be activated to compute a filtered navigation solution with increased precision. The filter, which is called Extended Navigation Solution (XNS), can be enabled by commanding the Device Handler to two additional modes, one with aiding disabled, one with aiding enabled.

Almanac data will be updated by the receiver as soon as it achieves a fix. These updates only affect the almanac data stored by the receiver itself. The almanac data that is sent to the receiver by the device handler during aided start is read from a memory within the device handler. This memory must be updated (via memory management service 6) regularly to contain an up to date almanac, an update interval of 1 month is recommended.

When commanded to On from Off, the GPS receiver is switched on. Then, depending on the requested submode, the handler either waits for a fix (unaided) or sets time, almanac, orbit, elevation mask and doppler window values, activates aided mode and waits for a fix (aided). As soon as the receiver is configured, the On mode is reached. Reaching the On mode does not imply a fix.

When commanded to On, Aided mode is activated or deactivated depending on the submode, and, if commanded from Unaided to Aided, the Orbit information is transmitted. Also, the XNS filter is activated or deactivated depending on the commanded submode.

When XNS is enabled, the message containing the filtered navigation data (F80) is enabled, and the interval of the message containing the normal navigation data (F40) is decreased.

Raw mode is entered immediately without sending any commands.

When commanded to mode Off, the handler does not perform any actions and directly shuts down the GPS.

There are four submodes for On mode: Unaided (0), Aided (1), Unaided with XNS (2) and Aided with XNS (3). The other modes only have the default submode 0.

TM/TC Interface

TM parameters generated by the GPS are the basic and filtered navigation data (in the F40 and F80 messages respectively), including position, velocity, range, range rate, time, etc. as well as additional data (in the F62 message) like the number of tracked satellites, position dilution of precision (PDOP) values and so on.

The GPS handler offers access to some configuration variables through its *private pool*. The variables include an almanac, the intervals with which the telemetry data is generated and the Doppler window used during aided startup.

The variables of the F40 message (navigation solution) are only written to the OBSW *datapool* if the navigation solution is valid. Otherwise, these variables are

set to invalid. The number of tracked satellites, the navigation status and the PDOP are updated regardless of this rule.

The interval in which the various periodic GPS messages are reported can be changed via direct command. For this it is important to note that some state transitions within the handler depend on the reception of a F40 message before continuing. This means that mode transitions can take longer or even fail if the interval for the F40 message is increased too much. A value of 10 s is a safe upper limit with larger values possible but not guaranteed to work.

Direct Service (2,130) commands include commands to set the mode of the GPS receivers (Aiding Mode, Tracking Mode, XNS), set the data generation rates, Doppler window and TLE data as well as set and request XNS configuration variables.

GPS CDM Handler

The three GPS receivers have two clock sources, one internal ("Single Clock") and one external ("Common Clock")—the latter being shared by all three receivers. The concept of the common clock is similar to the Clock Distribution Module (CDM), which can still be found in some documentation and is the namesake of the handler described here. As the clock management is shared by all receivers when the Common Clock is used, an additional device handler is available which configures the clock sources of all receivers.

The GPS CDM device handler modes are also listed in Table 7.6. Unlike other device handlers, the GPS CDM handler does not have a normal, Off or Raw mode. It can be commanded to one of two On modes to set the clock sources.

When commanded to On, the GPS CDM Handler sets the clock sources for all receivers according to the commanded submode and reads back the clock source TM from all receivers. The two submodes for the on mode are internal (0) and CDM (1).

There is one TM parameter containing the type of the clock source. There are no direct commands to the GPS CDM handler.

7.11.2.5 Sun Sensor Handler

The Sun sensors (SuS) are simple solar cells with their generated voltages and currents being measured by the PCDU. Therefore, there is no dedicated device handler in the OBSW and the SuS do not have modes. Instead, the PCDU Handler (see Sect. 5.6) requests the SuS variables from the PCDU on a regular basis and writes them to the *datapool* so that they can be used by the *ACS_Controller*.

Each of the eight physical SuS units consists of two redundant sensors (solar cells), both of which have an associated TM parameter containing the measured current. The temperature of each physical SuS unit is measured by one thermistor. These temperatures are also represented as TM parameters. There is no dedicated TC for the SuS.

7.11.2.6 MGT Device Handler

There are two MGT handlers, one for the nominal coils x/y/z and one for the redundant coils x/y/z. Details on the MGT commanding can be found in [66].

The MGT is a very simple device with coils being commanded in plus or minus direction or off, without the possibility to adjust the strength of the resulting magnetic field. The only accepted command is to set the coils to one of the three states and read back the value.

Modes and Submodes

The MGT device handler modes are listed in Table 7.7. When commanded from Off to On, the MGT is switched on and On mode is entered. The transition from Off to On takes 2.6 s at maximum. Whenever On mode (including from startup) or Raw mode is entered, the coils are deactivated. In Normal mode, the coil state is set as indicated in the *datapool*.

There are no submodes defined, instead the individual coils can be deactivated in the Normal mode by setting bit 0–3 in the submode, corresponding to coil x–z. For example, submode 0x02 (binary 00000010) means that only the x and z coils are set, the y coil will be off. In Submode 0x01 (binary 00000001) the x-coil is deactivated, in Submode 0x04 (binary 00000100) the z-coil is deactivated. This can be used if both nominal and redundant MGTs have lost a coil. Both can be commanded to nominal mode with mutually exclusive deactivated coils.

All other modes only have the default submode 0x00.

TM/TC Interface

The used TM parameters include only the commanded state and the dual-coil flag. There is no parameter indicating the actual state of the coils as indicated by the instrument. This results from the fact that in normal operations the coils will be switched on and off at a high frequency and the positive/off/negative value at a given time does not carry any essential information. There is only one Service (2,130) direct TC to set the coil state as defined in the *datapool*.

Table 7.7 MGT device handler modes

Handler	Mode	M#	Submode	SM#	Description
MGT N/R	Off	0	Default	0	Off
	On	1	Default	0	Coils are deactivated
	Normal	2	Nominal, *Bit Mask*	0–7	Coil state set as indicated in the *datapool*. Coils in x-, y- and z-direction can be deactivated by the bits 0–2 (0 on, 1 off), where the least significant bit is the x direction
			Redundant	32	Coil state set as indicated in the *datapool* if dual coil flag is set to true. Otherwise, all coils are switched off
	Raw	3	Default	0	Coils are deactivated

7.11.2.7 RW Device Handler

There is one RW handler for each of the four reaction wheels. Details on the RW commanding can be found in [65].

The reaction wheels are actuators that produce torque by spinning a metal disc. They can either be commanded in speed mode or in torque mode. In the first, the rotational speed of the disc is commanded directly while in the latter, the torque that should be produced is commanded.

Modes and Submodes

The RW device handler modes are listed in Table 7.8. When commanded from Off to On, the RW is switched on and On mode is entered. The transition from Off to On takes 5.6 s at maximum.

Whenever On mode is entered (including from startup), the RW is commanded to Standby (command "Disable Wheel"). If the RW is commanded from Normal mode/submode Torque, to the Raw mode, then the RW is commanded to zero torque before entering the Raw mode.

Normal mode/submode Idle, is used during RW commissioning for first start of the RW assembly with the careful test of RW polarity in orbit. The RW speed will be set to 10 % of the maximum permissible value, i.e. at 300 rpm in this submode. For use of the mode also refer to Sect. 11.7.

The RW handler main modes are Normal mode/Torque submode (1) and Normal mode/Speed submode (2). In Torque submode, a torque read from the *datapool* is

Table 7.8 RW device handler modes

Handler	Mode	M#	Submode	SM#	Description
RW 0–3	Off	0	Default	0	Off
	On	1	Default	0	Wheel disabled
	Normal	2	Idle	0	RW 0 at 300 rpm, RW 1–3 at −300 rpm.
			Torque	1	Torque is set as indicated in the *datapool*
			Speed	2	Speed is set as indicated in the *datapool*
			Manual	3	TM is requested normally, but no speed or torque is set. Instead, direct commanding is possible for test purposes
			Spin Down	4	TM is requested normally, wheels are spinning down freely. No commands are sent to the wheels
	Raw	3	Default	0	Torque is set to 0 before entering raw mode when coming from torque mode, otherwise no changes from previous state

set. In Speed submode, a speed read from a *datapool* variable is set. In both Torque and Speed mode, a get RW TM command is sent additionally in every cycle.

In Normal mode/submode Manual, TM is requested normally, but no speed or torque is set. Instead, direct commanding is possible for test purposes.

Normal mode/submode SpinDown is to let the RWs run down without breaking actively.

All other modes only have the default submode 0x00.

TM/TC Interface

TM parameters for the RW include commanded and measured speeds and commanded torques of the four wheels, the measured motor current as well as the measured temperature.

Through Service (2,130) commands, the wheels can be reset, TM can be requested, the wheels can be dis- and enabled and a torque or a speed can be commanded. Note that in normal mode, a speed or torque commanded per direct TC will always be overwritten in the next cycle with the value from the *datapool*.

7.11.3 ACS Assemblies

The ACS assemblies are used to group redundant devices, hiding the redundancy. Assemblies are also an interface between the device handlers and the subsystem mode machine. The assemblies implement the concepts as described in Sect. 3.2.3, notably offering the same modes and submodes as the device handlers, if not noted otherwise. They keep their respective mode and submode as long as enough children are in the correct state and fall back to mode OFF otherwise.

7.11.3.1 FOG_Assembly

The *FOG_Assembly* operates in a 3 out of 4 hot redundant configuration. That means as long as at least three FOG handlers are in the currently active mode, it keeps its mode and falls back to Off otherwise. The modes of the *FOG_Assembly* are listed in Table 7.9.

Table 7.9 *FOG_Assembly* modes

Assembly	Mode	M#	Submode	SM#	Description
FOG	Off	0	Default	0	Off
	On	1	Default	0	Mode is kept as long as 3 FOGs are available
	Normal	2	Default	0	Mode is kept as long as 3 FOGs are available

7.11.3.2 STR Assembly

The STR_*Assembly* only has one child but is needed for multiple reasons. First and foremost, the STR has an internal redundancy which is handled by the assembly. Secondly, the STR Handler offers different submodes which can be used indifferently of the system mode. To abstract this from the mode machine, the assembly offers only the default submode whereas the STR handler will be commanded to a submode stored in the assembly and configurable through the memory service. The modes of the STR_*Assembly* are listed in Table 7.10.

Table 7.10 *STR_Assembly* modes

Assembly	Mode	M#	Submode	SM#	Description
STR	Off	0	Default	0	Off
	On	1	Default	0	Safe boot-up mode
	Normal	2	*Bit Mask*	0–127	Same as STR modes. See Table 7.3
	Normal	2	Hidden	128	Use submode configured in assembly. Default is the STR handler default mode, 0. Settable by memory load

7.11.3.3 MGM Assembly

The *MGM_Assembly* operates in a 1 out of 2 hot redundant configuration. That means, that as long as at least one MGM is the currently active mode, the assembly will keep its mode, falling back to Off otherwise. The modes of the *MGM_Assembly* are listed in Table 7.11:

Table 7.11 *MGM_Assembly* modes

Assembly	Mode	M#	Submode	SM#	Description
MGM	Off	0	Default	0	Off
	On	1	Default	0	Same as children's modes. Mode is kept as long as one MGM is healhty and in the correct mode
	Normal	2	Slow Sampling	0	Same as children's modes. Mode is kept as long as one MGM is healhty and in the correct mode
			Medium Sampling	1	Same as children's modes. Mode is kept as long as one MGM is healhty and in the correct mode
			Fast Sampling	2	Same as children's modes. Mode is kept as long as one MGM is healhty and in the correct mode

7.11.3.4 GPS Assembly

The *GPS_Assembly* offers submodes which are not related to the GPS handlers' submodes. The submode into which the handlers are commanded are stored in the assembly and can be configured through the memory service.

The submodes of the assembly are:

- Single: Exactly one GPS is on.
- Multiple: At least one GPS is on, if more are healthy, more are on.
- GENIUS: All three GPS are on.

If the condition for the current submode is not met, the assembly falls back to Off.

Additionally to the GPS handlers, the GPS assembly also manages the GPS CDM Handler. The GPS CDM handler is commanded to internal clock in single and multiple submode and to common clock in GENIUS submode. The modes of the *GPS_Assembly* are listed in Table 7.12:

Table 7.12 *GPS_Assembly* modes

Assembly	Mode	M#	Submode	SM#	Description
GPS	Off	0	Default	0	Off
	On	1	Single	0	Only one GPS on. GPS CDM Handler in internal submode
			Multi	1	At least one GPS on, the others are on when healthy and off when faulty. Mode is kept as long as at least one GPS is healthy. GPS CDM Handler in internal submode
			GENIUS	2	All three GPS are on, only possible if all three GPS are healthy. Mode is kept as long as all GPS are healthy. GPS CDM Handler in CDM Submode

7.11.3.5 MGT Assembly

Like the *GPS_Assembly*, the *MGT_Assembly* offers submodes unrelated to the submodes of the MGT handler, which offers only the default submode.

The submodes of the *MGT_Assembly* are listed in Table 7.13:

- Single: One MGT handler is in Normal mode, the other on On, or Off if unhealthy
- Multi: At least one MGT handler is in Normal mode, the second one is Normal if healthy, Off if unhealthy
- Dual: Both MGT handlers are in Normal mode.

Table 7.13 *MGT_Assembly* modes

Assembly	Mode	M#	Submode	SM#	Description
MGT	Off	0	Default	0	Off
	On	1	Default	0	Same as children's modes
	Normal	2	Single	0	MGT N Handler in Normal mode, MGT R in On mode. If MGT N is unhealthy, MGT R in Normal mode. Mode is kept as long as at least one coil in each direction is healhty
			Multi	1	Both Handlers in Normal Mode as long as all coils and both handlers are healthy. If not all coils healthy or not both handlers healthy same as single mode
			Dual	2	Both Handlers in Normal mode. Only possible if all coils and both handlers are healthy

If the condition for the current submode is not met, the assembly falls back to Off.

7.11.3.6 RW Assembly

The *RW_Assembly* operates in a 3-out-of-4 redundant configuration. That means as long as at least three RW handler are in the currently active mode, it keeps its mode and falls back to Off otherwise. The *RW_Assembly* modes are listed in Table 7.14:

Table 7.14 *RW_Assembly* modes

Assembly	Mode	M#	Submode	SM#	Description
RW 0–3	Off	0	Default	0	Off
	On	1	Default	0	Same as children's modes
	Normal	2	Idle	0	Same as children's modes. Mode is kept as long as three RW are healhty and in the correct mode
			Torque	1	Same as children's modes. Mode is kept as long as three RW are healhty and in the correct mode
			Speed	2	Same as children's modes. Mode is kept as long as three RW are healhty and in the correct mode
			Manual	3	Same as children's modes. Mode is kept as long as three RW are healhty and in the correct mode
			Spin Down	4	Same as children's modes. Mode is kept as long as three RW are healhty and in the correct mode

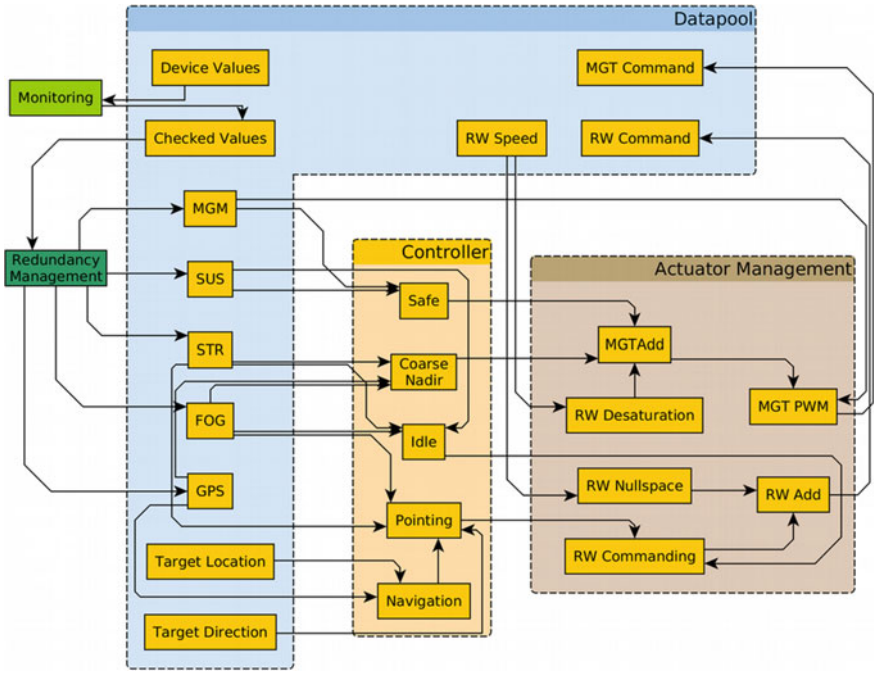


Fig. 7.17 *ACS_Controller* overview. © IRS, University of Stuttgart

7.11.4 ACS Controller

The *ACS_Controller* is divided into different modules (see Fig. 7.17), which are kept as separate as possible. The first modules are the sensor monitoring and the sensor redundancy management. The next module is the actual controller module which encapsulates the various algorithms. The last module is the actuator management which manages the commanding of the actuators.

The failure case detection sketched out in the following subsections is closely related to the ACS FDIR in Sect. 10.6. The mechanisms for failure detection and recovery are detailed there while in the following only a brief overview on the principle monitoring and control in the *ACS_Controller* shall be given.

The ACS control cycle rate is provided in Sect. 3.4.2.

7.11.4.1 Sensor Monitoring

The sensor monitoring module tries to detect device failures by a two step approach. First, each sensor value is checked against physical limits. Then each sensor is checked against its redundant sensors. Except for the GPS, after one device fails, the relative checks can not be done any more as the redundancy is lost.

For the GPS this applies after the second failure. As all sensors have only one redundancy (except for the GPS), in most cases a failure can only be detected but not located using the relative checks. This does not apply for the GPS, which has a triplex monitoring before the first failure, and for the FOGs which allow the attribution of some failures. The sensor monitoring is described in detail in [136] and Failure Detection, Isolation and Recovery in Sect. 10.6.

MGM
The physical limit for the MGMs is the maximum magnetic filed strength expected in the orbit of the satellite.

The relative check is performed by comparing the magnitude as well as the direction reported by the two MGMs. As only two devices are compared, a failure attribution is not possible (see Sect. 10.6.1).

GPS
GPS measurements are checked against the orbit of the satellite as a physical limit. If the radius or the absolute speed reported exceed the values of the orbit, a failure is reported.

Relative checks are performed by comparing the distance between the positions reported by the individual receivers (see Sect. 10.6.2).

STR
As the STR output is a quaternion which describes a "direction of view" relative to the spacecraft reference coordinate system, no physical limit check can be performed as all rotations described by a rotation quaternion are valid.

The relative check is performed by comparing the rotation reported by the two CHUs. With only two devices and no physical checks possible, a failure can only be detected, but not attributed (see Sect. 10.6.3).

SUS
The physical limits for the individual SUS cells are zero ampere and the maximal current that the cell can produce in direct sunlight.

Relative checks are done between the two redundant cells of each SUS sensor (see Sect. 10.6.4). Again, this does not allow an attribution of a failure.

FOG
The physical limit for the FOGs is their measurement range.

The relative checks are only done when all four FOGs are available. The relative check is performed by calculating a satellite rotation from three FOGs and comparing the result to the fourth (see Sect. 10.6.5). By permutating the single FOG, a faulty FOG can be detected in most cases. There are still cases in which a faulty FOG can not be detected by this method but will only show through a degraded performance of the *ACS_Controller*.

7.11.4.2 Sensor Redundancy Management

In this Module, the individual sensor values of redundant sensors are combined. The strategy depends on the individual sensor type. All have in common that no output value is calculated, if not enough sensor values are available. In this case, the corresponding *datapool* entry will be marked invalid.

MGM
If both MGMs are available, the direction of the magnetic field is calculated by combining both values using the pseudo inverse of the alignment matrix of both MGMs.

If only one MGM is available, the direction of the magnetic field is calculated by transforming the measurement into body coordinates.

GPS
GPS measurements are not combined, the first valid GPS information is used as an output.

STR
When both CHU are operated together, the orientation of the satellite is calculated by combining the two measurements, respecting the different accuracy in different axes.

If only one CHU is operated, the orientation is calculated by transforming the measured quaternion into the body coordinate system.

SUS
Depending on the deployment state of the solar panels, the correct set of SUS has to be selected. This selection is already done in the monitoring module.

Before further operations, the SUS values are filtered individually through a low pass filter.

The sun vector is then calculated by transforming the filtered SUS values using their alignment matrix.

Optionally an albedo reduction is performed, configurable through a memory load TC. The albedo reduction identifies sensors that would be in the shadow with the current sun vector and recalculates a new sun vector without these sensors.

FOG
If all four FOG units are available, the rotation of the satellite is calculated by combining the individual measurements using the pseudo inverse of the alignment matrix.

If only three FOG units are available, the rotation of the satellite is calculated using the inverse of the alignment matrix of the available FOG units.

STR and FOG combination
As the frequency of the STR updates is slower than the frequency of the control algorithm execution, the STR attitude measurements are interpolated using the FOG measurements.

As the STR and FOG measurements are related, one being the integral of the other, this function can also compensate a short outage of one of the two sensors, using only the other.

7.11.4.3 Control Logic

Within the controller module, there are individual submodules for the different control modes.

Safe Mode
The Safe Mode submodule implements the Detumble Mode as well as the Safe Mode algorithm. The transition between the two strategies is performed autonomously.

The Detumble Mode algorithm is a classical B-dot law by which the change of the magnetic field is measured and the MGTs are commanded to reduce this rotation. This is a very robust approach but does not lead to the satellite's inertial rotation completely being damped away but rather stabilizes the satellite with respect to the magnetic field.

The Safe Mode algorithm has a two phase approach. First, the satellite's solar panels are oriented towards the Sun, then, the satellite is spun up around the Sun vector to enter a spin stabilized state. This algorithm uses the SUS in addition to the MGMs and MGTs. As the algorithm depends on the Sun vector being available, it is passive during eclipse phase.

To use the additional information of the SUS to determine a more reliable estimation of the inertial rotation, the switching from Detumble to Safe Mode strategy and back is only performed during the daylight phase. The inertial rotation is estimated by combining the absolute value of the temporal derivative of the Sun vector (which is the absolute value of the rotation perpendicular to the Sun vector) and absolute value of the rotation around the Sun vector with respect to the magnetic field.

For the switching to the detumble mode, the two values are checked individually. The rotation around the Sun vector must not divert from the nominal rotation of +2°/s by more than 3°/s, while absolute value of the rotation perpendicular to the Sun vector must not exceed 3°/s (see also Fig. 2.3).

Coarse Nadir-Pointing Mode
The Coarse-Nadir-Pointing mode is used during the Commissioning Phase when the RWs are not yet available. It is able to keep the z-axis of the satellite approximately Earth pointing, using only the MGTs. It uses all ACS sensors to gain positional and rotational knowledge. To minimize blinding, the remaining degree of freedom is used to rotate the STR away from the Sun.

The much weaker actuators lead to a decreased pointing accuracy in contrast to the RW-based pointing submodule. But still the mode significantly increases the ground station visibility time compared to the Safe Mode with the rotating satellite and cyclical temporary mispointing of the S-band TTC antennas.

Idle Mode
The Idle Mode is similar to the Safe Mode in that it keeps the solar panels oriented to the Sun. However, it does this not using spin stabilization, but 3-axis stabilized, using all ACS sensors and actuators except for the GPS.

As pointing to the Sun offers one degree of freedom, namely the rotation around the Sun vector, the x-axis of the satellite is held parallel to the ecliptic normal vector. This also minimizes blinding of the STR (see also Fig. 2.4).

As long as a certain attitude information is not available, either due to eclipse or blinding of the STR, the motion in the corresponding direction is damped to keep the orientation of the satellite as constant as possible.

Pointing Mode
The pointing mode algorithm is designed to put into and keep the attitude of the satellite in a reference attitude. This algorithm is the most complex of the attitude control algorithms, employing variable gains and a slew rate constrains.

It can be used with a static and a variable attitude reference. With a static reference vector, this algorithm implements the inertial-pointing mode. For other pointing modes, a variable reference attitude as calculated by the navigation submodule is used.

Navigation
The navigation submodule is used to calculate a reference attitude which is then used by the pointing submodule to rotate the satellite.

The navigation submodule is supplied with a target point in the earth reference frame, usually a point on the surface or the center of the earth, forming the target and the nadir pointing mode. It calculates the attitude needed so that the instruments are pointing to the target point and solves the remaining degree of freedom such that the STR blinding is minimized.

To be able to calculate the relative direction to the target point, the navigation submodule employs the position as reported by the GPS.

7.11.4.4 Actuator Management

The actuator management calculates the individual actuator commands from the torque commands returned by the control algorithms. The torque commands are in the body coordinate system and do not consider actuator redundancies.

The MGT Add submodule (see Fig. 7.17), adds the torque commanded by the control algorithm to the torque commanded by the RW Desaturation submodule (see further below).

As the torque of the MGT depends on the external magnetic field, the commanded torque of the MGT Add submodule is combined with the measured magnetic field to get the power command for the individual coils. The power command is transformed into a pulse width modulated signal as the coils can only be commanded on or off. There is a dual coil flag by witch the control algorithm can signal to the actuator management that it should use both MGTs if available.

One MGM/MGT cycle is 5 s. Of this timeslot the MGM performs measurements for 0.5 s. Hence, the MGT can be activated for max. 4.5 s in every cycle, depending on the request from ACS control. The MGT electronics unit detects and provides information to the OBC/OBSW at 2 Hz about each coil's status and whether the coils are supplied with electric current (1 = current in positive direction, 0 = no current, −1 negative direction).

The RW management consists of three elements. The commanding submodule calculates the torque command for the individual wheels from the overall requested torque. The nullspace submodule is only active when all four RWs are available. It controlls the nullspace which is the degree of freedom which exists when four RWs are used to control the rotation around three axes. The aim of the nullspace module is to avoid unnecessary high speeds of a single RW. The desaturation submodule reduces the inertia stored in the RW assembly to reduce the speed of all RWs. The RW Add submodule sums up the commands from the commanding and the nullspace submodule.

7.11.4.5 ACS Controller Modes

The controller uses the general system-wide mode concept (i.e. it can be commanded with default mode commands). For its modes see also Table 7.15.

Table 7.15 *ACS_Controller* mode table

Object-ID	Mode	M#	Submode	SM#	Description
0x43410000	Off	0	Default	0	The controller is turned off entirely, output is set to invalid
	On	1	Default	0	The controller performs all monitoring activities
	Normal	2	Safe	2	The controller performs detumble and safe mode algorithms
			Coarse Nadir	3	The controller performs coarse nadir pointing algorithm
			Idle	4	The controller performs idle mode algorithms
			Nadir-Pointing	5	The controller performs nadir pointing algorithm
			Target-Pointing	6	The controller performs target pointing algorithm
			Inertial-Pointing	7	The controller performs inertial pointing algorithm

In Off mode, no activity is performed by the controller. In ON mode, the controller executes the sensor management modules but does not execute the controller module and the actuator management module and keeps the MGT and RW command output values invalid.

In Normal mode, the controller executes the sensor management modules, the controller module and the actuator management module. Which submodules are active is commanded by the submode:

In Safe submode, the Safe submodule of the controller module and the MGT related submodules of the actuator management are executed.

In the Idle submode, the Idle submodule of the controller module and the complete actuator management module is executed.

In Coarse-Nadir-Pointing submode, the Coarse Nadir submodule of the controller module and the MGT related submodules of the actuator management are executed.

In the Nadir- and Target-Pointing submode, the navigation and pointing submodules of the controller module and the complete actuator management module are executed.

In the Inertial-Pointing submode, the pointing submodule of the controller module and the complete actuator management module are executed.

Table 7.15 lists all available modes. When not in Off mode, the sensor monitoring and redundancy management are performed independently of the current submode, i.e. controller mode, as long as the input values are available. Yet, that additional information is not used by the control algorithms. If for example the FOG assembly is commanded to Normal mode during Safe mode, the rotation information about the satellite deduced by the FOGs is calculated and is available through the *datapool*, but the rotation is not used by the Safe submode algorithm.

7.11.5 ACS Subsystem Mode Transitions and Telemetry

The ACS subsystem mode commanding—according to the *ACS_Controller* modes defined in Table 7.15—is performed via PUS service 200 and the following sequences—see Table 7.16:

The ACS subsystem relevant TC packet definitions are included in the spacecraft TC table in annex Sect. 17.3.1.

The ACS subsystem relevant TM packet definitions are included in the spacecraft TM table in annex Sect. 17.3.2.

The ACS subsystem relevant Event TM packets are included in the spacecraft Event TM table in annex Sect. 17.3.3.

Details on the TM parameter positions in the diverse packets have to be taken from the MIB.

Table 7.16 ACS subsystem modes and transition sequences

#	**ACS Modes**	**0x01000200**				
Commanding to Off Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Off	0x19000000	1			
	Name	**Table ID**	**WaitTime**	**Check?**		
	Off_Target	0x19000000	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000				
	MGT Ass.	0x41000300				
	MGM Ass.	0x41000200				
	RW Ass.	0x41000100				
	STR Ass.	0x41000800				
	FOG Ass.	0x41000400				
	GPS Ass.	0x41000500				
#						
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Off_Trans	0x19000100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Off	0	Off	0
	MGT Ass.	0x41000300	Off	0	Off	0
	MGM Ass.	0x41000200	Off	0	Off	0
	RW Ass.	0x41000100	Off	0	Off	0
	STR Ass.	0x41000800	Off	0	Off	0
	FOG Ass.	0x41000400	Off	0	Off	0

(continued)

Table 7.16 (continued)

	GPS Ass.	0x41000500	Off	0	Off	0
Commanding to Detumble Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Detumble	0x19010000	1	0x19000000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	Detumble_Target	0x19010000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	Detumble	1
	MGT Ass.	0x41000300	Normal	2	Dual	1
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100				
	STR Ass.	0x41000800				
	FOG Ass.	0x41000400				
	GPS Ass.	0x41000500				
#						
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Detumble_Trans_1	0x19010100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000				
	MGT Ass.	0x41000300	Normal	2	Dual	1
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100	Off	0	Off	0
	STR Ass.	0x41000800	Off	0	Off	0

(continued)

Table 7.16 (continued)

	FOG Ass.	0x41000400	Off	0	Off	0
	GPS Ass.	0x41000500	Off	0	Off	0
	Name	**Table ID**	**WaitTime**	**Check?**		
	Detumble_Trans_2	0x19010200	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	Detumble	1
	MGT Ass.	0x41000300				
	MGM Ass.	0x41000200				
	RW Ass.	0x41000100				
	STR Ass.	0x41000800				
	FOG Ass.	0x41000400				
	GPS Ass.	0x41000500				
Commanding to Safe Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Safe	0x19020000	1	0x19000000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	Safe_Target	0x19020000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	Safe	2
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100				
	STR Ass.	0x41000800				
	FOG Ass.	0x41000400				

(continued)

Table 7.16 (continued)

	GPS Ass.	0x41000500				
#						
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Safe_Trans_1	0x19020100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000				
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100	Off	0	Off	0
	STR Ass.	0x41000800	Off	0	Off	0
	FOG Ass.	0x41000400	Off	0	Off	0
	GPS Ass.	0x41000500	Off	0	Off	0
	Name	**Table ID**	**WaitTime**	**Check?**		
	Safe_Trans_2	0x19020200	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	Safe	2
	MGT Ass.	0x41000300				
	MGM Ass.	0x41000200				
	RW Ass.	0x41000100				
	STR Ass.	0x41000800				
	FOG Ass.	0x41000400				
	GPS Ass.	0x41000500				

(continued)

Table 7.16 (continued)

Commanding to CoarseNadir Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	CoarseNadir	0x19030000	1	0x19020000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	CoarseNadir_Target	0x19030000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	CoarseNadir	3
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100				
	STR Ass.	0x41000800	Normal	2	Default	7
	FOG Ass.	0x41000400				
	GPS Ass.	0x41000500	On	1	Single	0
#						
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	CoarseNadir_Trans_1	0x19030100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000				
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100	Off	0	Off	0
	STR Ass.	0x41000800	Normal	2	Default	7
	FOG Ass.	0x41000400	Off	0	Off	0

(continued)

Table 7.16 (continued)

	GPS Ass.	0x41000500	On	1	Single	0
	Name	**Table ID**	**WaitTime**	**Check?**		
	CoarseNadir_Trans_2	0x19030200	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	CoarseNadir	3
	MGT Ass.	0x41000300				
	MGM Ass.	0x41000200				
	RW Ass.	0x41000100				
	STR Ass.	0x41000800				
	FOG Ass.	0x41000400				
	GPS Ass.	0x41000500				
Commanding to Idle Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Idle	0x19040000	1	0x19020000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	Idle_Target	0x19040000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	Idle	4
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100	Normal	2	Torque	1
	STR Ass.	0x41000800	Normal	2	Default	7
	FOG Ass.	0x41000400	Normal	2	Default	0
	GPS Ass.	0x41000500	On	1	Single	0

(continued)

Table 7.16 (continued)

#						
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Idle_Trans_1	0x19040100	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000				
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100	Normal	2	Torque	1
	STR Ass.	0x41000800	Normal	2	Default	7
	FOG Ass.	0x41000400	Normal	2	Default	0
	GPS Ass.	0x41000500	On	1	Single	0
	Name	**Table ID**	**WaitTime**	**Check?**		
	Idle_Trans_2	0x19040200	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	Idle	4
	MGT Ass.	0x41000300				
	MGM Ass.	0x41000200				
	RW Ass.	0x41000100				
	STR Ass.	0x41000800				
	FOG Ass.	0x41000400				
	GPS Ass.	0x41000500				
Commanding to NadirPt Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	NadirPt	0x19050000	1	0x19040000		

(continued)

Table 7.16 (continued)

	Name	**Table ID**	**WaitTime**	**Check?**		
	NadirPt_Target	0x19050000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	NadirPt	5
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100	Normal	2	Torque	1
	STR Ass.	0x41000800	Normal	2	Default	7
	FOG Ass.	0x41000400	Normal	2	Default	0
	GPS Ass.	0x41000500	On	1	Single	0
#						
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	NadirPt_Trans_1	0x19050100	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000				
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100	Normal	2	Torque	1
	STR Ass.	0x41000800	Normal	2	Default	7
	FOG Ass.	0x41000400	Normal	2	Default	0
	GPS Ass.	0x41000500	On	1	Single	0
	Name	**Table ID**	**WaitTime**	**Check?**		
	NadirPt_Trans_2	0x19050200	0	0		

(continued)

Table 7.16 (continued)

	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	NadirPt	5
	MGT Ass.	0x41000300				
	MGM Ass.	0x41000200				
	RW Ass.	0x41000100				
	STR Ass.	0x41000800				
	FOG Ass.	0x41000400				
	GPS Ass.	0x41000500				
Commanding to TargetPt Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	TargetPt	0x19060000	1	0x19040000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	TargetPt_Target	0x19060000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	TargetPt	6
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100	Normal	2	Torque	1
	STR Ass.	0x41000800	Normal	2	Default	7
	FOG Ass.	0x41000400	Normal	2	Default	0
	GPS Ass.	0x41000500	On	1	Single	0
#						
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	TargetPt_Trans_1	0x19060100	0	1		

(continued)

Table 7.16 (continued)

	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000				
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100	Normal	2	Torque	1
	STR Ass.	0x41000800	Normal	2	Default	7
	FOG Ass.	0x41000400	Normal	2	Default	0
	GPS Ass.	0x41000500	On	1	Single	0
	Name	**Table ID**	**WaitTime**	**Check?**		
	TargetPt_Trans_2	0x19060200	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000	Normal	2	TargetPt	6
	MGT Ass.	0x41000300				
	MGM Ass.	0x41000200				
	RW Ass.	0x41000100				
	STR Ass.	0x41000800				
	FOG Ass.	0x41000400				
	GPS Ass.	0x41000500				
Commanding to InertialPt Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	InertialPt	0x19070000	1	0x19040000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	InertialPt_Target	0x19070000	0	1		

(continued)

Table 7.16 (continued)

	System	Obj ID	Mode	Mode ID	Submode	Submode ID
	ACS Controller	0x43410000	Normal	2	InertialPt	7
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100	Normal	2	Torque	1
	STR Ass.	0x41000800	Normal	2	Default	7
	FOG Ass.	0x41000400	Normal	2	Default	0
	GPS Ass.	0x41000500	On	1	Single	0
#						
Achieved by transitions						
	Name	**Table ID**	**WaitTime**	**Check?**		
	InertialPt_Trans_1	0x19070100	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	ACS Controller	0x43410000				
	MGT Ass.	0x41000300	Normal	2	Single	0
	MGM Ass.	0x41000200	Normal	2	Fast	2
	RW Ass.	0x41000100	Normal	2	Torque	1
	STR Ass.	0x41000800	Normal	2	Default	7
	FOG Ass.	0x41000400	Normal	2	Default	0
	GPS Ass.	0x41000500	On	1	Single	0
	Name	**Table ID**	**WaitTime**	**Check?**		
	InertialPt_Trans_2	0x19070200	0	0		

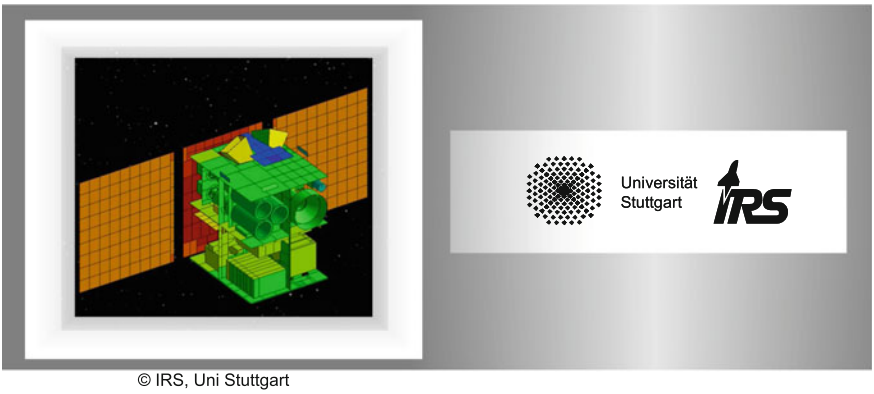
(continued)

Table 7.16 (continued)

	System	Obj ID	Mode	Mode ID	Submode	Submode ID
	ACS Controller	0x43410000	Normal	2	InertialPt	7
	MGT Ass.	0x41000300				
	MGM Ass.	0x41000200				
	RW Ass.	0x41000100				
	STR Ass.	0x41000800				
	FOG Ass.	0x41000400				
	GPS Ass.	0x41000500				

Chapter 8
Thermal Control Subsystem

Fabian Steinmetz and Sebastian Keil



Abstract The chapter provides an overview on the platform thermal control subsystem. It explains the placement of thermal equipment over the diverse satellite compartments at hand of an example FLP-based spacecraft. It explains the logic of component operational and non operational limits, the allocated sensors, sensor redundancies, allocated compartment heaters and heater redundancies. The last part comprises the onboard software elements and functions for control of the thermal subsystem.

Keywords Thermal subsystem overview · Satellite compartments (Modules) and temperature control · Sensors · Heaters and redundancies · Thermal subsystem control

F. Steinmetz (✉) · S. Keil
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: steinmetz@irs.uni-stuttgart.de

S. Keil
e-mail: keil@irs.uni-stuttgart.de

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_8

8.1 Thermal Subsystem Overview

The thermal control subsystem (TCS) has to control the thermal balance in order to keep all components of the satellite within the permissible temperature limits. The thermal balance is analyzed in all operating modes and in all of the thermal environments it may be exposed to.

The overall thermal control of a spacecraft in orbit is usually achieved by balancing the heat emitted by the spacecraft as IR radiation with radiators, against the heat dissipated by its internal components plus the heat absorbed from the environment, such as direct sunlight, reflected sunlight from Earth (albedo) and Earth infrared. The thermal balance of a satellite is illustrated in Fig. 8.1.

Thermal control techniques can be divided into two main categories. Passive thermal control which uses materials, coatings (black paint for internal heat exchange of the components), insulation or surface finishes (such as blankets or second surface mirrors) to maintain temperature limits. Active thermal control—which is generally more complex and expensive—maintains the temperature by active means, such as heaters or thermo-electric coolers. This requires extra electrical power. In addition, a control algorithm is necessary to control the heater and/or the coolers. In both categories an iterative design process has to be performed, where all environmental effects, requirements of the components, the mission and cost is considered [141].

In general, thermal control subsystems (TCS) are designed to keep the spacecraft at the cool end of permissible temperature ranges. Cooler components generally have a longer lifetime—especially for electronic equipment and batteries—and can heat up during operation without the need for active cooling. Furthermore, the dissipated heat can be used to heat other components, which saves heating power and energy [141]. To prevent components from dropping below the permissible temperature limits in non-operational modes, the satellite can be equipped with electric heaters [142].

For thermal control there are usually two temperature limits defined for each component:

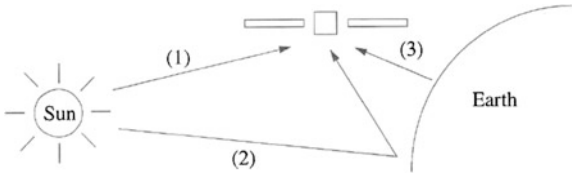


Fig. 8.1 Thermal balance of a satellite—sunlight (1), albedo (2), Earth infrared (3)

- **Operational limits** These are the boundaries within which the component must remain while being in operation. The component will switch off, in case the temperature exceeds the permissible operational limits.
- **Non-operational limits** These are the boundaries within which the component must remain permanently, even when not being powered/operated.

Exceeding the permissible temperature limits can result in permanent equipment damage. Every component is qualified for those two limits. When a component operates close or exactly at a temperature limit, operation is not yet affected.

Thermal control is also used to ensure that temperature gradient requirements are within range. An example of a gradient requirement is to ensure that one side of a structure does not get warmer or colder than the opposite side by more than 30 °C. A larger gradient could cause structural deformation such that pointing is adversely impacted, possibly permanently [141].

The thermal control subsystem is defined as generic term and includes all hardware and software components as well as techniques of a satellite to maintain a certain temperature range. The *TCS_Controller* is part of the Onboard Software (OBSW) running on the Onboard Computer (OBC) and provides the interfaces for the TCS algorithm to receive the hardware and software data, such as heater status and temperature sensor values.

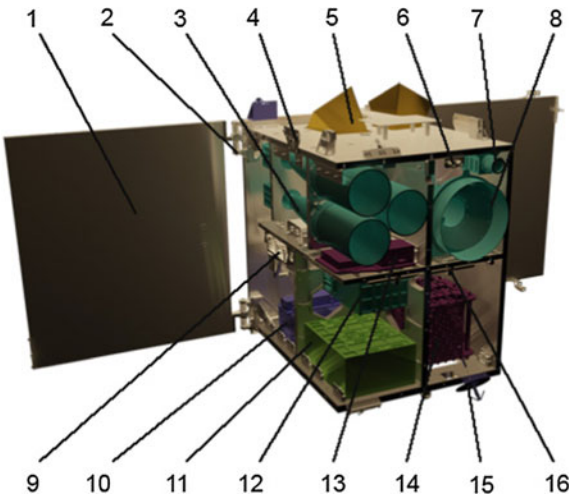


Fig. 8.2 Placement of thermally active equipment—first FLP-based satellite "Flying Laptop". © IRS, University of Stuttgart. *1* Solar panel, *2* Deployment mechanism of solar panels, *3* Payload 2 (MICS Camera), *4* Sun sensor, *5* Star tracker, *6* Payload 4 (OSIRIS Laser Terminal), *7* Payload 1 (PAMCAM Camera), *8* Antenna of the data downlink system, *9* Reaction wheels, *10* Communication system, *11* On-board computer, *12* Payload computer PLOC, *13* Payload 3 (AIS Receiver), *14* Battery, *15* Antenna of communication system, *16* Magnetotorquer

The TCS algorithm monitors the temperature and the thermal status of the individual satellite components, generates the heater commands to achieve the permissible temperature limits and returns fault signals, if sensors are invalid or required values do not match or cannot be achieved. Figure 8.2 depicts thermally relevant elements of the satellite system—here with the example payloads of the first FLP mission "Flying Laptop".

Some components are thermally decoupled from the rest of the satellite platform, such as the battery to obtain a certain temperature level or the star tracker in order to protect them from the heat dissipated by the surrounding components. In addition, vital components as the battery, OBC, PCDU and TT&C are equipped with thermostats. These thermostats switch on additional heaters, in case of a system failure and a temperature drops below the allowed non-operational limit. These internal unit heaters are not controlled by the TCS algorithm.

All data used by the TCS algorithm is provided through the *TCS_Controller* from the OBSW *datapool*. The thermal equipment detected to be defective by the *TCS_Controller* has to be marked in separate arrays at the *datapool* and will be handled by the Failure Detection, Isolation and Recovery (FDIR). The TCS algorithm only identifies failures. It will not perform any isolation nor recovery actions. For details on thermal FDIR please refer to Chap. 10.

The generated heater commands from the TCS algorithm are stored in the OBSW *datapool* as on/off statuses and are handled by the OBSW through the corresponding device handler. This architecture allows to develop the control algorithm without caring about the source of the data and the target of the result data. The entire requirements for the TCS algorithm are described in Sect. 8.2 and have been implemented within the spacecraft OBSW.

Figure 8.3 depicts the thermal overview of the satellite platform—here detailed using the example of the FLP Generation 1 spacecraft "Flying Laptop" with its 4 payloads. Thermally there exist elements which are fully exposed to space and not included into the "box" of the satellite. The overview comprises the following modules:

- The first module is called "Solar Array", as it contains the solar panels as well as several other equipments that are exposed to the space environment.
- The next Module is the "Payload Module" which contains the payloads and some avionics units like magnetometers.
- Then follows the "Avionics Module"—also referred to as "Core Module"—which contains most of the ACS sensors and actuators.
- And on the bottom coupled to the S/C radiator is the "Service Module" with OBC, PCDU, Battery and S-band transceiver.

For temperature measurements Pt-1000 thermistors and equipment internal sensors of the different components are used by the TCS. The thermistors are placed near those S/C components which are not equipped with internal electronic

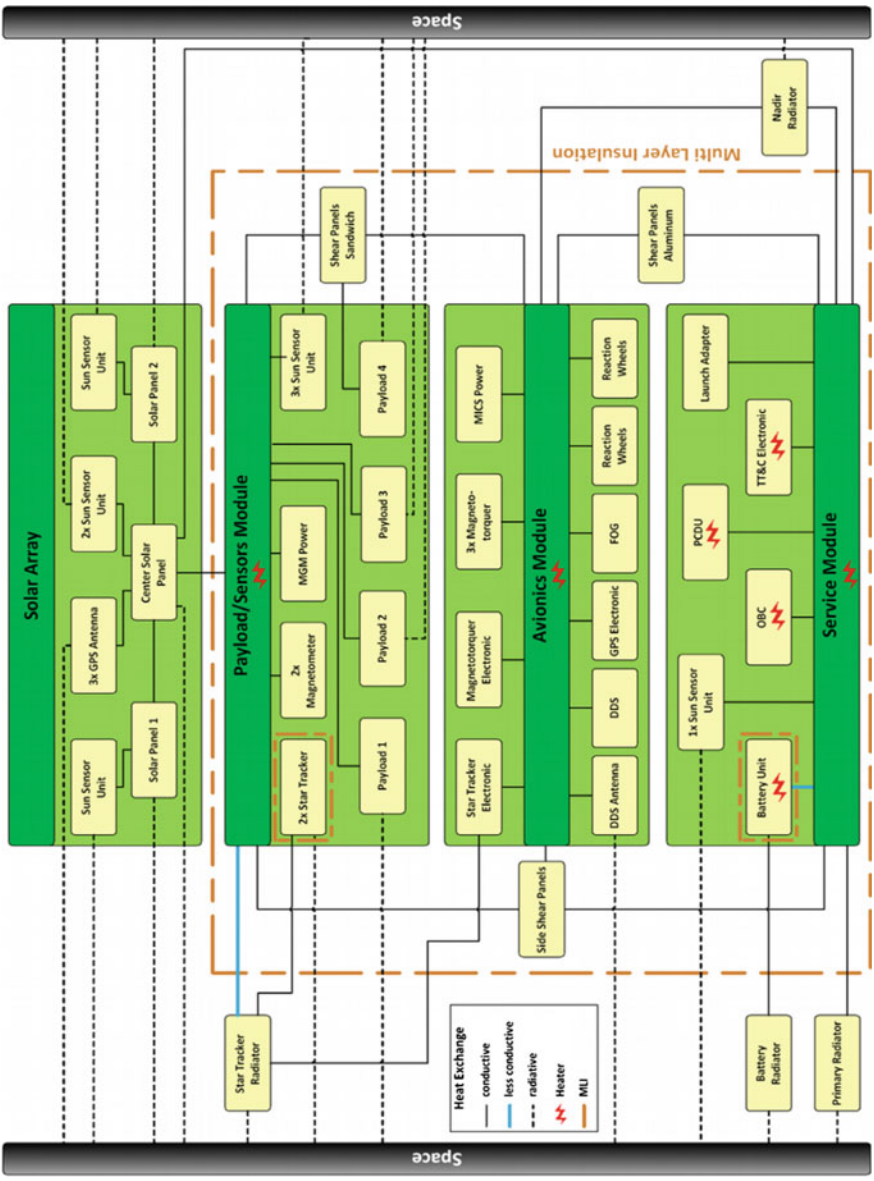


Fig. 8.3 Thermal subsystem overview—4 payloads. © IRS, University of Stuttgart

Table 8.1 Component temperatures, limits and sensors. © IRS, University of Stuttgart

Component	Component Temperature Parameter Name	Component Temperature Parameter Unit	Component Temperature Parameter ID [hex]	NOP-lower	OP-lower	OP-upper 1st Sensor	OP-upper 2nd Sensor	OP-upper Fallback	NOP-upper	1st Sensor	2nd Sensor	3rd Sensor	4th Sensor	5th Sensor	Module
BATTERY_S0	TYTCTP00	°C	58000000	0	0	40			40	S0 Top	S0 Bottom				Battery
BATTERY_S1	TYTCTP01	°C	58000100	0	0	40			40	S1 Top	S1 Bottom				Battery
BATTERY_S2	TYTCTP02	°C	58000200	0	0	40			40	S2 Top	S2 Bottom				Battery
DDS_TRANSMITTER_N	TYTCTP03	°C	58000400	−20	0	40			70	DDS TX N	DDS TX R				Core
DDS_TRANSMITTER_R	TYTCTP04	°C	58000500	−20	0	40			70	DDS TX R	DDS TX N				Core
FOG	TYTCTP05	°C	58000600	−54	−40	77			85	FOG 0	FOG 1				Core
GPS_0	TYTCTP06	°C	58000700	−50	−20	70			80	GPS	MGT Electronic				Core
GPS_1	TYTCTP07	°C	58000800	−50	−20	70			80	GPS	MGT Electronic				Core
GPS_2	TYTCTP08	°C	58000900	−50	−20	70			80	GPS	MGT Electronic				Core
MAGNETOMETER_0	TYTCTP09	°C	58000A00	−40	−30	85			100	MGM 0					Payload
MAGNETOMETER_1	TYTCTP10	°C	58000B00	−40	−30	85			100	MGM 1					Payload
MGT_N	TYTCTP11	°C	58000C00	−40	−30	85			105	MGT Electronic	GPS				Core
MGT_R	TYTCTP12	°C	58000D00	−40	−30	85			105	MGT Electronic	GPS				Core
PL2(MICS)	TYTCTP13	°C	58000E00	−10	0	40			50	PL2 MICSG CCD	PL2 MICSR CCD	PL2 MICSN CCD			Payload
OBC_0	TYTCTP14	°C	58000F00	−50	−40	80			125	OBC 0	OBC 1				Service
OBC_1	TYTCTP15	C	58001000	−50	−40	80			125	OBC 1	OBC 0				Service
PL4(OSIRIS)	TYTCTP16	°C	58001100	−40	−20	60			80						Payload

(continued)

Table 8.1 (continued)

Component	Component Temperature Parameter Name	Component Temperature Parameter Unit	Component Temperature Parameter ID [hex]	NOP-lower	OP-lower	OP-upper 1st Sensor	OP-upper 2nd Sensor	OP-upper Fallback	NOP-upper	1st Sensor	2nd Sensor	3rd Sensor	4th Sensor	5th Sensor	Module
										PL4 OSIRIS Laser	PL4 OSIRIS Internal	PL4 OSIRIS			
PL1(PAMCAM)	TYTCTP17	°C	58001200	−30	−20	50			75	PL1 PAMCAM					Payload
PCDU	TYTCTP18	°C	58001400	−40	−40	70			80	PCDU Internal P1.0	PCDU Internal P1.1	PCDU Internal P2	PCDU Internal P3	PCDU Internal P5	Service
PLOC_N	TYTCTP19	°C	58001500	-30	0	70	60	40	80	PLOC N FPGA	PLOC N PCB	OBC1			Core
PLOC_R	TYTCTP20	°C	58001600	−30	0	70	60	40	80	PLOC R FPGA	PLOC R PCB	OBC1			Core
RW_0	TYTCTP21	°C	58001700	−35	−20	60			70	RW 0	RW 1	RW 2			Core
RW_1	TYTCTP22	°C	58001800	−35	−20	60			70	RW 1	RW 2	RW 3			Core
RW_2	TYTCTP23	°C	58001900	−35	−20	60			70	RW 2	RW 3	RW 0			Core
RW_3	TYTCTP24	°C	58001A00	−35	−20	60			70	RW 3	RW 0	RW 1			Core
STAR_TRACKER	TYTCTP25	°C	58001C00	−50	−30	80	73	60	100	STR CPU	STR SMPS				Core
TTCN_RX	TYTCTP26	°C	58001D00	−40	−25	55			80	TTCN Rx	TTCR Rx	TTCN Tx			Service
TTCN_TX	TYTCTP27	°C	58001E00	−40	−25	55			80	TTCR Rx	TTCN Rx	TTCR Tx			Service
TTCR_RX	TYTCTP28	°C	58001F00	−40	−25	55			80	TTCN Tx	TTCR Tx	TTCN Rx			Service
TTCR_TX	TYTCTP29	°C	58002000	−40	−25	55			80	TTCR Tx	TTCN Tx	TTCR Rx			Service
PL3(AIS)	TYTCTP30	°C	58002100	−30	−20	40			50						Payload
MGT ROD 0/2*	TYTCTP31	°C	58002200	−55	−50	70			80	MGT Rod 02	MGT Rod 1				Core
MGT ROD 1*	TYTCTP32	°C	580022300	−55	−50	70			80	MGT Rod 02	MGT Rod 1				Core

temperature sensors (for further details refer to Sect. 8.2). The temperature parameters, tolerable equipment temperature ranges and allocated sensors (including spacecraft module identification) are provided in a first overview in Table 8.1. Some of the sensors are for information purpose only.

To ensure a temperature monitoring for every component (equipment instance) of the satellite, almost each component temperature has two assigned redundant sensors. In case of an invalid sensor the value of the redundant sensor is used. The sensor redundancy assignment was performed after the implementation of the sensors into the satellite and is listed in Table 17.13. Please note that sensors being marked exclusively with "−1" in this table are those which are not used for control of TCS heaters and which therefore are not implemented redundant. They are only used by the *TCS_Controller* to measure certain equipment characteristics, such as for the Solar Panels etc.

To protect the components from dropping below the permitted temperature limits in unexpected situations, the FLP is equipped with nominal and redundant heater groups. For example the FLP spacecraft "Flying Laptop" has three nominal and three redundant heaters—for Service Module, Core Module and Payload Module.[1] Heaters are electrical resistors that release heat caused by an electrical current. If the temperature drops below the permitted limit, the TCS algorithm activates the assigned heater.

For heater control, it is necessary to know the temperature limits of all components. As mentioned earlier there are two types of limits to be defined—the operational and the non-operational limits. The current thermal status can be defined by means of the current temperature and the limits of each component. Figure 8.4 describes the thermal status definition of a component.

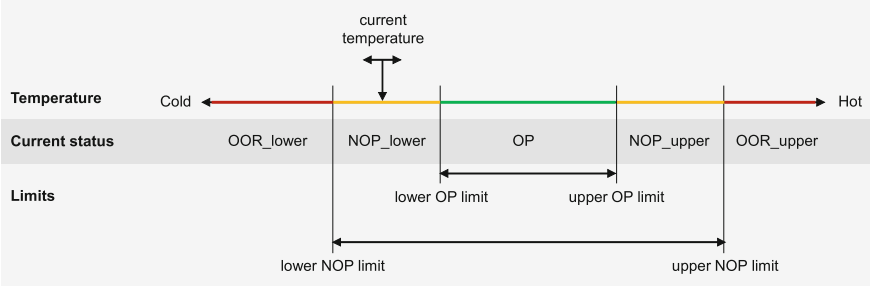


Fig. 8.4 Thermal status definition of a component. © IRS, University of Stuttgart

[1]The battery compartment is treated as a dedicated own "Module" by the thermal control algorithm, as the battery is thermally decoupled from the spacecraft. More details are provided in Sect. 8.3.2.3.

Table 8.2 Module temperatures and assigned sensors

Name	Component Temperature Parameter Name	Component Temperature Parameter Unit	Component Temperature Parameter ID [hex]	Assigned Sensors
CORE_MODULE	TYTMTP00	°C	58000300	Core Module
				FOG 0
				MGT Electronic
				MGT Rod 1
				GPS
				MGT Rod 0/2
				DDS Transmitter N
				DDS Transmitter R
				FOG 1
PAYLOAD_MODULE	TYTMTP01	°C	58001300	Payload Module
				Payload 1 (PANCAM)
				Payload 4 (OSIRIS)
SERVICE_MODULE	TYTMTP02	°C	58001B00	Service Module
				OBC 1
				OBC 0
BATTERY_MODULE				Battery S0 Bottom
				Battery S1 Bottom
				Battery S2 Top
				Battery S0 Top
				Battery S1 Top
				Battery S2 Bottom

Table 8.3 Heater assignment to the satellite modules

Name	Abbreviation	PCDU Fuse #	Heater Object ID [hex]	Fuse Current Parameter Name	Fuse Current Parameter Unit	Fuse Current Parameter ID [hex]
Payload Module Heater N	PM	22	43540075	PPTFSC22	A	41110416
Payload Module Heater R	PM	23	43540076	PPTFSC23	A	41110417
Core Module Heater N	CM	22	43540071	PPTFSC22	A	41110416
Core Module Heater R	CM	23	43540072	PPTFSC23	A	41110417
Service Module Heater N	SM	22	43540073	PPTFSC22	A	41110416
Service Module Heater R	SM	23	43540074	PPTFSC23	A	41110417

In addition to the individual equipment temperatures, a temperature value is aggregated for each satellite module. This is done numerically and the algorithm is explained further down in Sect. 8.3. Table 8.2 provides the list of temperature parameters and their aggregation component sources.

And finally Table 8.3 lists the allocation of the heaters to the corresponding satellite modules.

Figure 8.5 provides a graphical overview on the temperature limits of each "Flying Laptop" component as representative mission example (the detailed values are already included in Table 8.1). The red-marked area identifies the range beyond the measurement capability of the applied equipment (PCDU plus PT1000 thermistors).

The battery and the star tracker are marked in yellow as they are thermally decoupled from the other components of the satellite. The star trackers are facing towards deep space and are decoupled from the Payload Module inner side. The battery is geometrically included in the Service Module but is thermally decoupled from the structure.

The components marked in grey are just for temperature information, they will have no effect on heater control.

8.2 Sensors, Calibration, Limits

There exist 66 temperature sensors on board the first FLP spacecraft—the "Flying Laptop"—of which 32 are individually placed at important locations or on/near components in the structure. As an exception, there are two sensors for each battery string since this is a crucial component. This allows for the detection of local temperature gradients in the battery strings. The other 34 are internal temperature sensors which are included within equipment units, providing a temperature in their respective housekeeping telemetry data packet. With these 66 sensors it is possible to determine a temperature for all components. There are six heaters—two on each Structure Module—for nominal thermal control.

For those thermistors implemented in electronic units (such as PCDU) the calibration curves are provided by the unit manufacturer—respectively the units directly report their temperatures calibrated in °C.

For thermistors applied externally to S/C equipment units or for thermistors applied to the satellite structure per compartment the PT1000 thermistor raw values in Ω are measured by the PCDU and they are digitized by the PCDU's AD converters. The translation of the digital raw values to temperature values into °C is performed according to calibration functions which result from the characterization

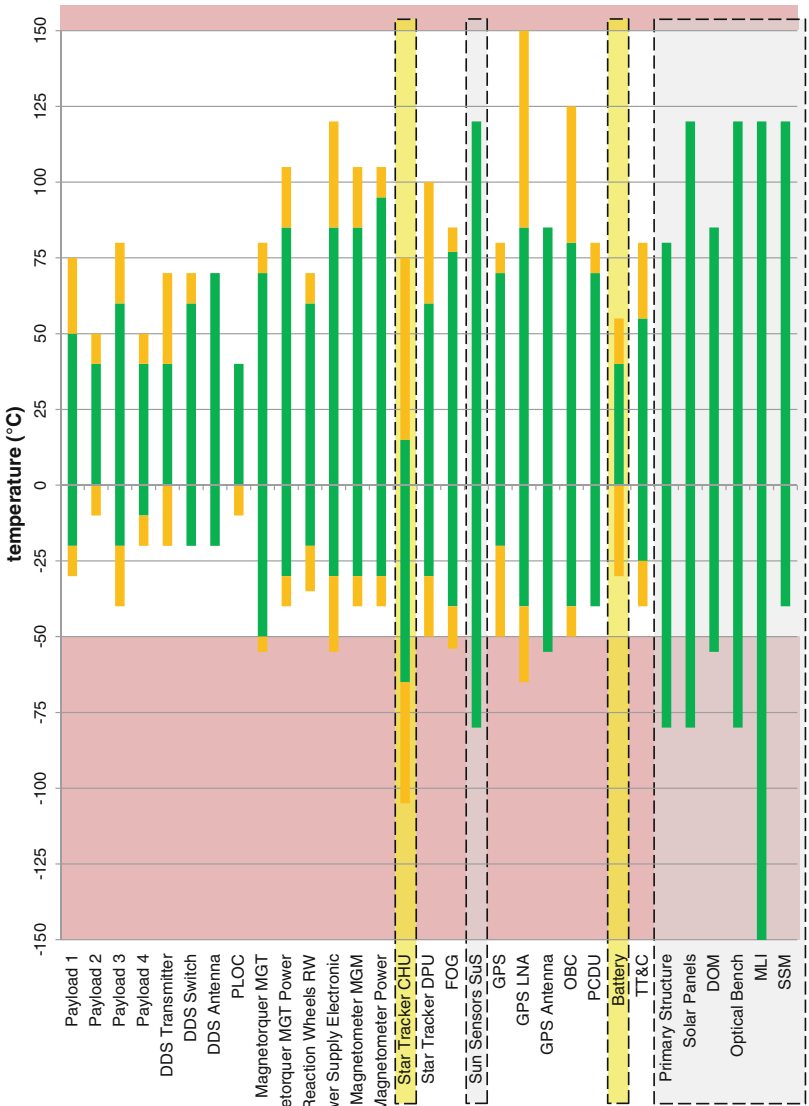


Fig. 8.5 Equipment operational and non-operational temperature limits. © IRS, University of Stuttgart

campaign performed in [130] where further details and tolerance factors can be taken from. The conversion function is a simple polynomial

$$f(R) = aR^3 + bR^2 + cR + d \tag{8.1}$$

with:

f	temperature value
R	sensor raw value
a, b, c, d	calibration function parameters.

The polynomial parameter values for all sensors are included in Table 17.13. The full calibration function parameter set of Eq. 8.1 for all such component sensors (internal and external) is included in Table 17.13.

8.3 TCS Subsystem Control

The *TCS_Controller* in the OBSW (see Fig. 8.6) is responsible for the monitoring of the temperature sensors and for commanding the heaters. The thermal control algorithm was developed in the frame of a thesis [130]. The algorithm was implemented and verified on the basis of MATLAB Simulink® from where then C-Code was generated and was adapted later to properly integrate into the C++ OBSW architecture.

The block-diagram of Fig. 8.7 depicts the top level design of the TCS algorithm. The necessary input is represented by the blue boxes on the left side. The purple box indicates the additional constants to provide settings, limits and function parameters. The generated output is in blue boxes on the right and the grey box on the right represents the failure information. The green box in the middle includes the individual functions of the TCS algorithm in the *TCS_Controller*. The block diagram shows that the TCS algorithm only works with OBSW *datapool* data while the *TCS_Controller* in light green provides the interface. This design of the TCS algorithm fulfills the requirements for integration into the object-oriented OBSW

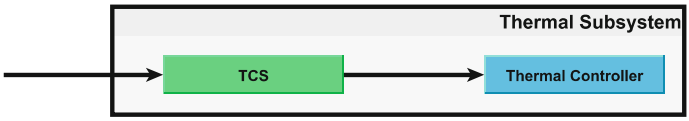


Fig. 8.6 TCS controller as part of thermal subsystem. © IRS, University of Stuttgart

architecture. A more detailed description on implemented TCS relevant OBSW objects their functions and allocated parameters is provided in Sect. 8.3.2.
The interaction of the different software elements are described hereafter:

- All data used by the TCS algorithm is provided to the *TCS_Controller* from the OBSW *datapool* for accessibility by other controllers and software components.
- All heater switch statuses generated by the TCS algorithm is provided to the OBSW *datapool*.
- All sensor or heater fault status values and subsequent marking of such thermal equipment as defective is provided to the OBSW *datapool*.
- All identified failures are stored in separate arrays in the *datapool* to allow the *FDIR* to react. More details on variable handling are provided in Sect. 8.3.2.

The developed algorithm considers the following elements:

Temperatures
The temperature of each component of the satellite has to be monitored. The temperature has to be measured with the given hardware. These values have to be transformed to unified temperature values and checked for physical reasonable, the measurement range and temperature gradient. In case of a failure a redundancy must be guaranteed.

Component statuses
The temperature limits of each component have to be checked and a temperature status has to be defined. It also has to be monitored, if the target status is achieved after a certain time.

Heater circuits
Each heater status and in particular the heater current has to be compared to the target status. To maintain or archive the target status, heater commands have to be provided by the TCS algorithm. As important components have priority, a priority order must be respected.

Sub-functions
Due to the main requirements it is necessary to store current values, such as the current temperature value for the temperature gradient check.

Battery system
The battery system of the FLP platform imposes further requirements to the algorithm. The current temperature of each battery temperature sensor has to be provided by the *TCS_Controller* in a separate array in the *datapool*. The TCS algorithm has to calculate the average temperature of each of the three battery strings and return these values in a separate array. The TCS algorithm has to return a warning if the temperature difference between the two temperature sensors of one battery string exceeds a certain threshold.

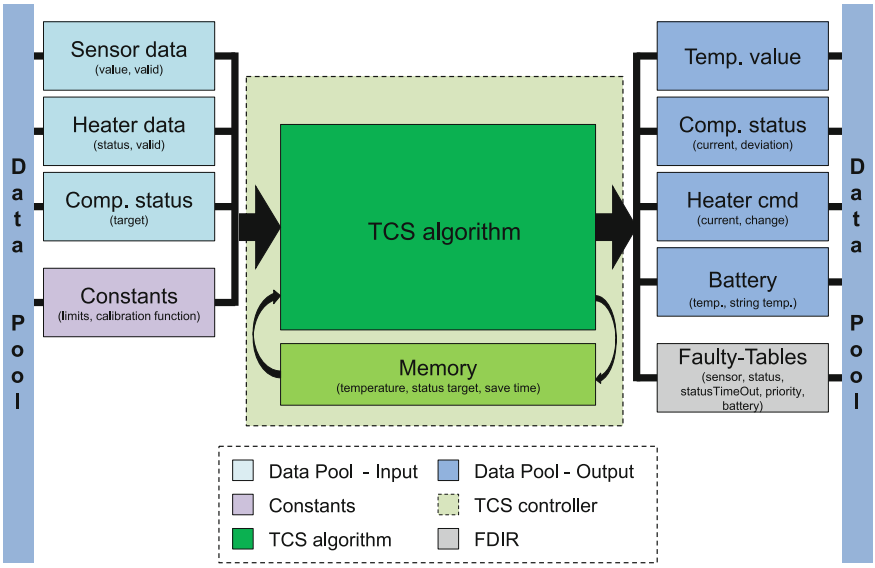


Fig. 8.7 Design of the TCS algorithm. © IRS, University of Stuttgart

8.3.1 Initial Control Concept

The initial control concept as developed in [130] worked as "loop" controlling the spacecraft heaters as response to thermal sensor values considering multiple sensors and their limit values for each spacecraft component to be thermally controlled. Figure 8.8 shows the complete MATLAB Simulink® model of the TCS algorithm.

Functional breakdown

The different functions of the initially designed TCS algorithm are listed in the following Table. 8.4. All functions of the TCS algorithm are listed as followed and described on the next pages.

Table 8.4 Initial TCS control subfunctions

1 converterFunction	6 statusCurrent vs. statusTarget
2 temperatureGradientCheck	7 heaterCmdFunction
3 redundancyFunction	8 batterySystem
4 statusCheck	9 memoryFunction
5 statusTime	

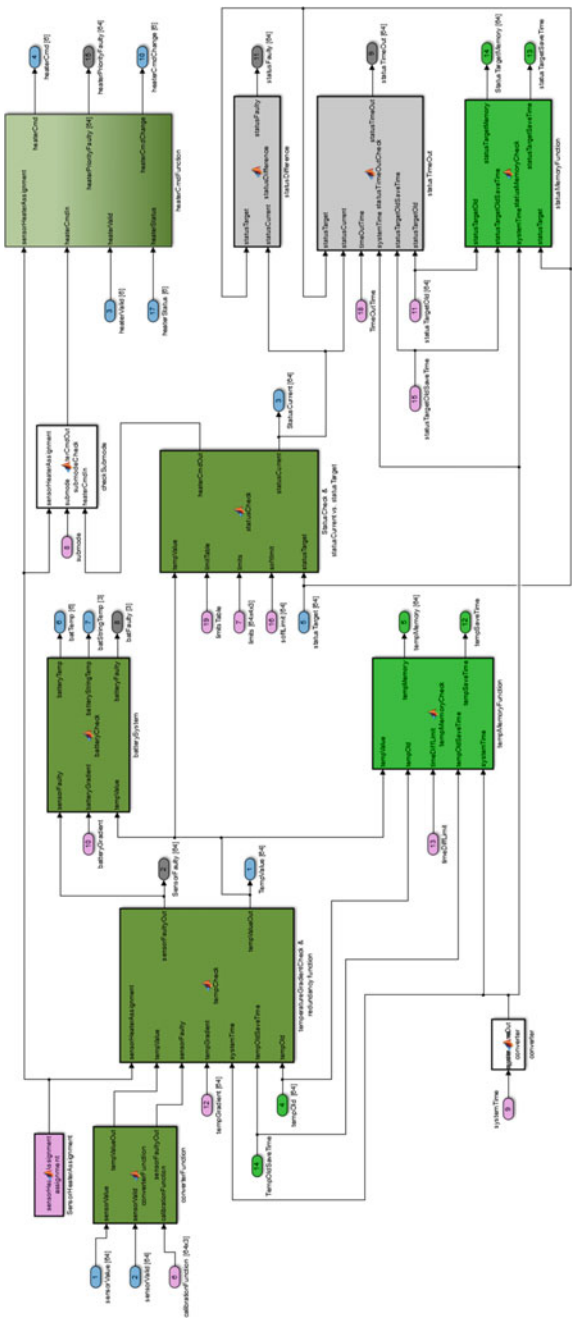


Fig. 8.8 Matlab Simulink model of the TCS algorithm. © IRS, University of Stuttgart

8.3.2 TCS Controller

The internal of the *TCS_Controller* as finally implemented is modeled according to the physical subsystem layout (see Fig. 8.9). In its object-oriented architecture it holds temperature sensor objects, satellite module objects, component objects and heater objects.

In the first approximation "components" are the same as the "devices" cited in the previous chapters. However while devices always are electronic or electric functional units (like a GPS receiver), a "component" also can be an artificial structure or an entirely passive element that needs thermal balancing—see later the battery string.

Components are allocated to modules. Each component has one or more temperature sensors assigned. Each module has two redundant heaters, which are used to heat all components of a module simultaneously.

Each temperature sensor object monitors one physical temperature sensor. A thermal component then uses multiple sensors to calculate a temperature value for one physical component. Depending on the state (on/off) of the component and its temperature, components request the activation or deactivation of the module heater from the module. The module calculates from all sensor the requests which heaters should be activated.

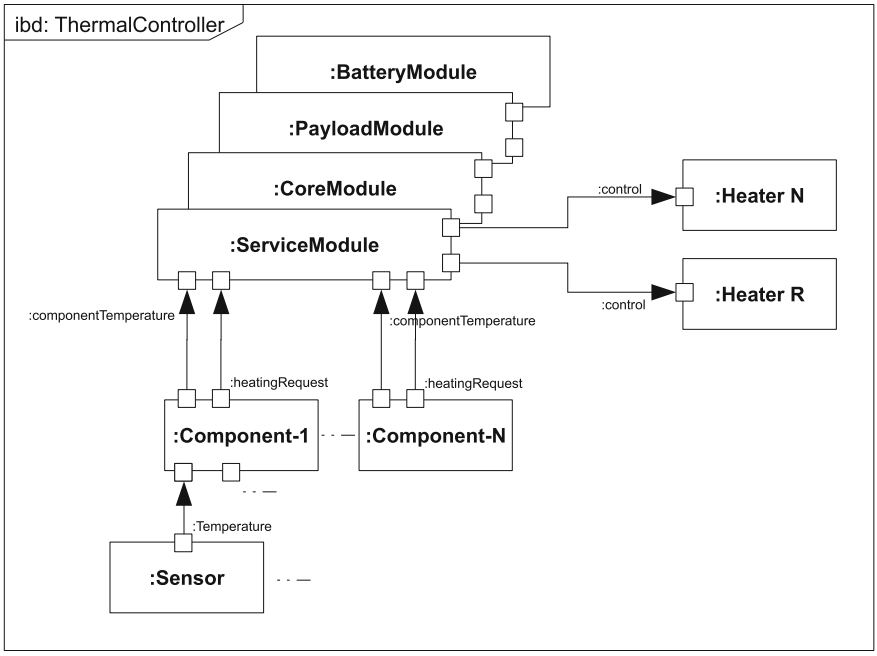


Fig. 8.9 *TCS_Controller* block diagram. © IRS, University of Stuttgart

Event reports are generated for all critical thermal Events. Monitoring reports are used to inform about non critical Events.

The TCS control cycle rate is provided in Sect. 3.4.2. The *TCS_Controller* also can be run with a selectable time offset relative to the other OBSW tasks.

8.3.2.1 Temperature Sensors

For each physical temperature sensor there is a temperature sensor object which performs two functions. First, it monitors the sensor output and flags erroneous behavior. Secondly, it converts the sensor value to °C.

Monitoring

Temperature sensors are monitored against violation of physical temperature limits and so are the gradients of each temperature. The limit monitoring is performed before the conversion of the sensor value to °C—i.e. on raw value level—whereas the gradient monitoring is performed after conversion.

In every monitoring step, the current unconverted value is checked against a maximum and a minimum value. The gradient is checked only if the previous execution step calculated a valid sensor value. If a sensor is unavailable—be it invalid or not updated—the gradient is not monitored. The gradient is only calculated and monitored again when the value is available for two consecutive execution steps.

Conversion

The values of most sensors are not directly available in °C but in different units. The conversion from the sensor units to °C is done using the already cited polynomial function. A polynomial of 2nd degree is applied for most sensors. In the case of the transceiver internal temperatures a polynomial function of 3rd degree is applied—see Eq. 8.1.

Failure handling

Each temperature sensor object has a health flag. If a temperature sensor is not healthy or the input is not within validity range, no monitoring is performed and the output is set to invalid. Setting the sensor output to invalid implies setting the value to an invalid temperature.

The FLP contains also some sensors only for information purposes. When these sensors become invalid, the temperature value is set to an artificial value of 999 °C.

If the sensor value is out of the monitoring limits, the output is set to invalid as well. If the gradient is too high, the output value will still be valid.

Reporting

When the sensor value exceeds the physical valid range or the gradient becomes too high, a failure Event is reported. The limit violation is only reported once—as soon as the value exceeds the limit. The gradient violation is reported every time a too high gradient is calculated, independent of the previous execution step.

All violations of the upper or lower limits are reported using Monitoring reports.

In case where the health state of a sensor is not healthy, no Events nor reports are generated.

8.3.2.2 Thermal Components

For every spacecraft equipment device (each STR, RW etc.), there exists a corresponding thermal component object.[2] This object tries to calculate a best estimate of the temperature of the component and determines the thermal state the of the component. If a component is too cold or too hot, the thermal component object requests the module heater to be switched on or off, respectively.

Temperature estimation—Sensor redundancy

Every component has a number of redundant temperature sensors assigned. Here lies the main difference of the implemented algorithm to the initial MATLAB Simulink® based design from [130]:

Depending on the type of the component here the following strategies are used to resolve the redundancy:

- The default case is that the sensors are ordered and the temperature value of the first valid sensor (see Table 8.1) is used as the estimated temperature. In case where the sensor is marked as defective the corresponding 2nd sensor from is used and so forth (see Table 8.1).
- For the PCDU and the Battery components, the mean value of all available sensors is used as the estimated temperature.
- For all components, if there is no valid sensor available, the module temperature as calculated by the associated module (see Table 8.1 and Sect. 8.3.2.3) object is used as estimated temperature.
  In case of the "Flying Laptop" this e.g. applies for the payload 3, the AIS receiver, which has neither an internal sensor nor an external due to its late accommodation into the mission as customer furnished item.

This implementation allows that in case where the first sensor is marked as healthy no further sensor values are to be checked and in addition contradicting in-limit/out-of-limit results from the multiple sensors of a component near boundary temperatures can be avoided.

Thermal state

Components have two temperature ranges, operational and non operational—see Figs. 8.4 and 8.5 and Table 8.1. Based on these ranges, the component object determines the current state of the component as either

[2]To be entirely precise: The actual rule is that for every device handler which controls a device that must be thermally monitored, there is a thermal component. The only exception is the OBC which only has one thermal component for each redundant OBC side although it has more equipment device handlers.

- out of limit low,
- non operational low,
- operational,
- non operational high,
- out of limit high.

If no temperature can be estimated, the current state is set to unknown. The above mentioned thermal state is used in addition to the temperature range to identify whether the temperature component is to be actively controlled. Marking a component NOT to be actively controlled is done by adding an *ignored* state for all states mentioned before. The cause for a component not to be controlled can be resulting from several factors:

- The target state of the component is ignored (see paragraphs below).
- The heater request is being overwritten in the request voting (see Sect. 8.3.2.3).
- The Module strategy (see Sect. 8.3.2.3) does not include the components.

Target state
Each component has a target state that can be

- *operational*,
- *non-operational*,
- *heating*,
- *ignored*.

The target state controls how the component object will request heater activation or deactivation from the module and also the monitoring of the thermal state.

- If the target state is *operational*, the component object will try to keep the component temperature in the operational temperature range and generate a failure message as soon as the current state is not operational.
- If the target state is *non-operational*, the component object will try to keep the component in the non-operational temperature range and generate a failure message as soon as the current state is out of limit.
- If the target state is *heating*, the component object will try to keep the component in the operational temperature range but unlike the operational target state, failure messages will only be generated if the component temperature exceeds the non-operational range.
- If the target state is *ignored*, the component object will not request heater activation or deactivation. This is also reflected in the current state which will be marked as ignored.

The *non-operational* target state is the default state for components that are switched off. The *operational* target state is set by the device handler when a component is switched on. This state cannot be commanded and the target state can

also not be changed by command if it is *operational*. This is done to ensure that no component is operated outside of its specified temperature range. In contingency cases the operational target state can be overwritten to *non-operational* but this should be done with great care as it poses the risk of damaging a component.

Note: The ignored target state should only be used in contingency cases.

Heater request

There are five heater requests that the component object can request from the module.

- *emergency_On*: This indicates that the component is close to its lower non-operational temperature limit and the heater should be activated.
- *on*: This indicates that the component is close to its lower operational temperature limit and the heater should be activated.
- *don't_care*: This indicates that the component is within its desired temperature range and no heating is required.
- *off*: This indicates that the component is close to its upper operational temperature limit and the corresponding heater should be switched off.
- *emergencyOff*: This indicates that the component is close to its upper non-operational temperature limit and the heater should be switched off.

The request behavior is controlled using three variables, *heaterOn*, *hysteresis and heaterOff*. *heaterOn* is the temperature difference between the lower limit and the temperature at which the heater will be requested to be activated. *hysteresis* is the temperature difference the component must be heated before the heater can be switched off. *heaterOff* is the temperature difference between the upper limit and the temperature at which the heater is requested to be deactivated.

If the target state is *non-operational*, only the non-operational limits of the component are considered. Figure 8.10 shows the resulting request graph.

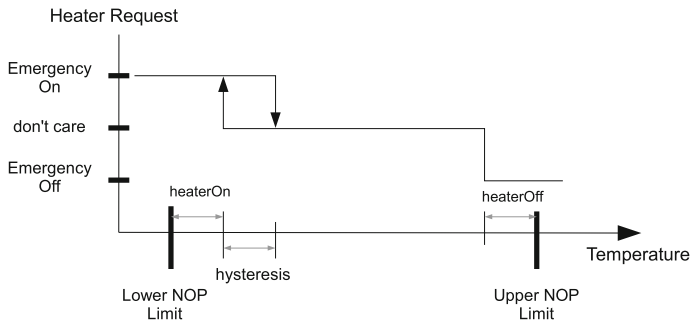


Fig. 8.10 Heater request in target state non-operational. © IRS, University of Stuttgart

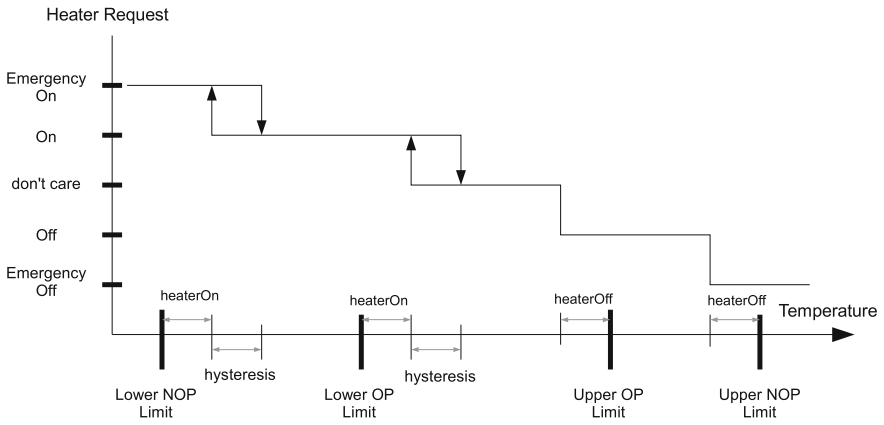


Fig. 8.11 Heater request in target state heating or operational. © IRS, University of Stuttgart

If the target state is *heating* or *operational*, both the non-operational and operational limits are considered, leading to the more complex request graph in Fig. 8.11.

Failure handling

In case of no valid sensors and no valid module temperature or if no module is assigned, the estimated temperature is marked as invalid and *don't_care* is returned as heater request.

Reporting

If a component exceeds its requested temperature range—to either the upper or lower side—a failure event is reported. If a component leaves or enters its requested range, a monitoring report is sent. The requested temperature range is the operational range for target state operational and the non-operational range for the other target states. If the target state is set to *ignored*, no Events or reports are generated.

Special cases

One special component is the battery string component. It has two sensors assigned, the top and the bottom sensor. The temperature is estimated as the mean value of all available sensors. In addition to the monitoring of the estimated temperature, the battery string component also monitors the spatial gradient in the battery string. If both sensors are available, the spatial gradient is calculated and is compared to a limit value. If the gradient is larger than the permitted limit, a failure Event is reported.

There are some other components which select the upper operational limit depending on the currently used sensor. This is done to compensate adverse placement of some sensors, for example sensors close to a strong heat source such as the CPU of a device.

8.3.2.3 Thermal Modules

For every Spacecraft Module, there is a thermal Module object. This object groups the component objects of the spacecraft equipment located in the Spacecraft Module. The Module object calculates a module temperature, receives the heater requests from its components, votes between these requests and controls the two module heaters.

Additionally, the Module management in the *TCS_Controller* can control the heater itself to heat up the entire Module to such a temperature that all components are finally within their operational temperature range.

Module temperature and limit temperature
The Module temperature is calculated as the average of all available temperature sensors assigned to this S/C Module. If no temperature sensors are available, the Module temperature will be set invalid and the Module state will be set to unknown.

The Module also calculates a limit temperature, which is the highest of the lower operational limits of all components assigned to the Module.

Current state
The current state of a thermal Module can be

- *operational*,
- *non-operational*,
- *timeout*
- or *unknown*.

If the Module temperature is above the lower limit temperature, the current state is *operational*. If the temperature is lower than the limit temperature, the current state is *non-operational*. If no temperature sensors are available, the current state is *unknown*. The *timeout* state is defined in [130] but is not yet used in the implemented algorithm. Heater timeout is handled by the TCS FDIR—see Sect. 10.7

Target state
Each Module has a target state assigned which controls the heating strategy of the Module itself. If the target state is set to *heating*, the Module will try to keep the Module temperature in such a range that the Module's current state is *operational*. If the target state is *passive*, the Module will not control the Module temperature.

The way the module requests heater activation is controlled by two variables, *heaterOn* and *hysteresis*. *heaterOn* is the temperature difference between the limit temperature and the temperature at which the heater is requested to be activated. The heater limits are defined with a small offset to the status limits, to ensure the temperature of a component does not drop below the status limit. This soft limit ensures that a heater turns off after a certain temperature difference to the heater

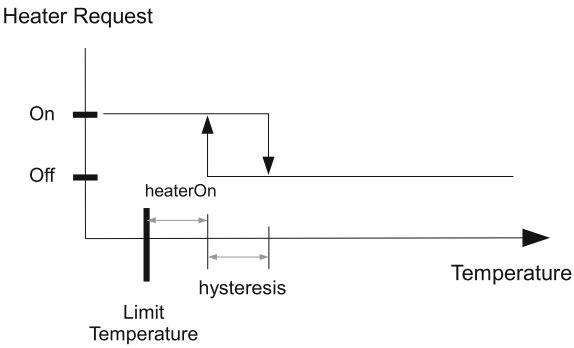


Fig. 8.12 Heater request in target state heating. © IRS, University of Stuttgart

limit. *hysteresis* is the temperature that the module must be heated before the heater is requested be deactivated. The resulting request map is shown in Fig. 8.12.

Request voting
Based on all the heater requests from all the components assigned to it, a module performs a two step voting. First, components are grouped by their priority (see Table 17.13 in the annex). Thereafter within the group the requests are voted with the following priorities—highest first:

- *emergency_Off*
- *emergency_On*
- *off*
- *on*
- *don't_care*

After that, the requests between the groups are voted with the following priorities—highest first:

- *safe*
- *idle*
- *payload*

The resulting request is the component request. This can be either *on* or *off*, where *dont'_care* is treated as *off*. The component request is logically OR-ed with the module's heater request to get the heater command.

Heater control
Each module has two heater objects assigned. The heater objects have health flags and thus can be marked as faulty. The module will only activate healthy heaters. Unhealthy heaters will be deactivated, while heaters marked as externally controlled will be ignored.

Component information
After the heater command has been calculated, each component is informed whether its request was fulfilled or not. Components of which the heating request is not being fulfilled are marked as ignored.

Strategy
A thermal S/C Module offers four control strategies:

- *passive*,
- *survival*,
- *single*
- and *dual*.

Using *passive* strategy, the heaters are not commanded, while all other functions are executed nominally. When entering *passive* strategy, the heaters are deactivated once and then are ignored.

The *single* strategy is the default strategy. It will perform all functions nominally and use the nominal heater to heat the components. If the nominal heater is faulty, the redundant heater will be used. If the redundant heater is faulty as well, no heater will be commanded and all components will be marked as ignored.

Using *survival* strategy (see Sect. 10.7.3. on TCS Survival Mode), only the components with a priority safe are considered in the request voting. All other components will be marked as ignored. Only one heater is used, as detailed for the Single strategy.

The *dual* strategy differs from single strategy only in using all available healthy heaters in parallel.

Special Cases
The battery Module differs from the other Modules in that it has no heater assigned. Thus it marks all it components as *ignored*—independent from the current strategy.

8.3.2.4 TCS Controller Modes

The *TCS_Controller* offers the modes/submodes as listed in Table 8.5.

Table 8.5 *TCS_Controller* modes

Ctrler	Mode	M#	Submode	SM#	Description
TCS	Off	0	Default	0	Off
	On	1	Default	0	Passive, i.e. only temperature sensors values are processed, no heaters are commanded
	Normal	2	Auto	0	Controller decides whether to use survival, single or dual control according to Battery SoC
			Survival	1	Only Safe Mode devices are heated if necessary until a timeout is reached
			Single	2	All components are heated if necessary using one heater circuit per module only
			Dual	3	All components are heated if necessary using both heater circuits in each module

8.3.3 TCS Subsystem Mode Transitions and Telemetry

The TCS-Subsystem mode commanding—according to the *TCS_Controller* modes defined in Table 8.5—is performed via PUS service 200 and the following sequences—see Table 8.6.

The TCS subsystem relevant TC packet definitions are included in the spacecraft TC table in annex Sect. 17.3.1.

Table 8.6 TCS subsystem modes and transition sequences

#	**TCS Modes**	**0x01000500**				
#						
#						
Commanding to Off						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Off	0x59000000	1			
	Name	**Table ID**	**WaitTime**	**Check?**		
	Off_Target	0x59000000	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	TCS Controller	0x43540000				
#						
Achieved by transition						
	Name	**Table ID**	**WaitTime**	**Check?**		
	Off_Trans	0x59000100	0	0		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	TCS Controller	0x43540000	Off	0	Off	0
#						
#						
Commanding to Default Mode						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Default	0x59010000	1	0x59000000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	Default_Target	0x59010000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	TCS Controller	0x43540000	Normal	2	Auto	0
#						
Achieved by transition						
	Name	**Table ID**	**WaitTime**	**Check?**		
	_Trans	0x59010100	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	TCS Controller	0x43540000	Normal	2	Auto	0

(continued)

Table 8.6 (continued)

#						
#						
Commanding to Survival						
Sequence	**Name**	**Seq ID**	**HardCoded?**	**Fallback**		
	Survival	0x59020000	1	0x59000000		
	Name	**Table ID**	**WaitTime**	**Check?**		
	Survival_Target	0x59020000	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	TCS Controller	0x43540000	Normal	2	Survival	1
#						
Achieved by transition						
	Name	**Table ID**	**WaitTime**	**Check?**		
	_Trans	0x59020100	0	1		
	System	**Obj ID**	**Mode**	**Mode ID**	**Submode**	**Submode ID**
	TCS Controller	0x43540000	Normal	2	Survival	1
#						
#						

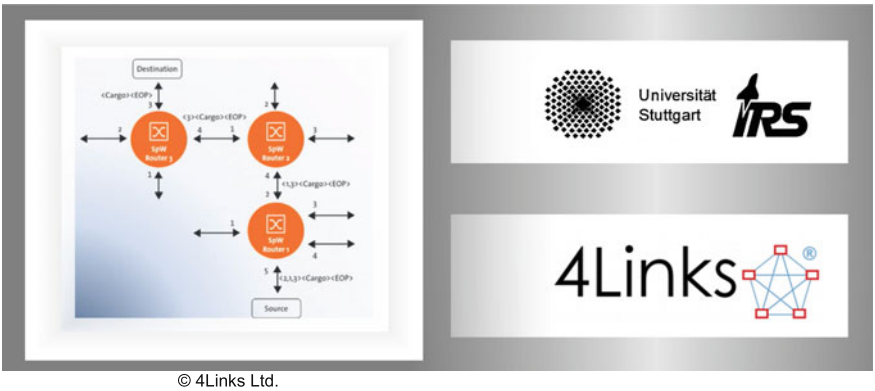
The TCS subsystem relevant TM packet definitions are included in the spacecraft TM table in annex Sect. 17.3.2.

The TCS subsystem relevant Event TM packets are included in the spacecraft Event TM table in annex Sect. 17.3.3.

Details on the TM parameter positions in the diverse packets have to be taken from the MIB.

Chapter 9
Payload Control Subsystem

Philipp Hagel, Paul Walker and Jens Eickhoff



Abstract The payloads themselves are not part of the satellite platform. Nevertheless the options to couple payloads and payload module equipment like a mass memory unit and payload data transmission equipment to the platform computer are relevant for the Flight Operations Manual. Various architectures are explained here. The technical solution for the first FLP-based satellite of the Institute of Space Systems in Stuttgart is covered in a bit more detail. The second part treats more advanced SpaceWire network centric architectures as proposed by 4Links and Airbus DS.

P. Hagel (✉)
Institute of Space Systems, Deutsches Zentrum für Luft- und Raumfahrt,
Bremen, Germany
e-mail: philipp.hagel@dlr.de

P. Walker
4Links Ltd, Milton Keynes, UK
e-mail: paul@4links.co.uk

J. Eickhoff
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_9

Keywords Payload to onboard computer connection schemes · Payload control of the first FLP-based satellite · Modular payload data processing · Network centric architectures · SpaceWire networks

9.1 Aspects of Payload Control and Data Handling

Concerning the topic of payload management on board a satellite three main functions have to be considered:

- Control of the payload in the sense of payload mode switching and processing of the feedback—i.e. payload command and control
- Processing the payload generated mission product-data which might need an on-board compression, preprocessing, quality analysis or packaging for downlink
- The functionality of payload mission product storage and retrieval at time of downlink.

For the latter it is obvious that this function typically is assigned to a dedicated unit, a Mass Memory Unit (MMU). If such a unit also provides an implementation of a file system and in/out streaming functions to/from files, such units are called Mass Memory and Formatting Unit (MMFU).

The other functions—payload control and mission product processing—can be allocated in multiple ways in the on-board architecture. The following points have to be considered for the implementation of the payload control subsystem.

- Payload command and control highly depend on the inherent intelligence of the payload sensor/detector electronics.
- Concerning data management—routing, formatting, storage—the selected solution must consider the data rates produced by each individual payload. This might range from low data rates of a cosmic dust sensor up to the extreme data rates produced by RADAR payloads.
- Concerning the mission product data furthermore the required CPU performance for data processing/compression has to be considered.
- And finally the selected architecture is driven by the number and types of payloads mounted on the platform for a dedicated mission.

The possible solutions generally are, to let a dedicated function be performed by the main platform OBC, or alternatively, by a dedicated payload management "unit" in its most general sense.

In the following sections some elementary solutions are described. In Sect. 9.2 the implementation of the payload control subsystem for the first FLP based satellite—the "Flying Laptop"—is described. Section 9.3 describes the implementation of the architecture followed by Airbus DS for the FLP Generation 2.

9.1.1 Payload Control via the Platform OBC

The simplest solution for payload management is the case where the payloads are directly connected to the OBC (see Fig. 9.1). This approach implies that:

- The satellite will have a lower complexity due to not using any systems in between OBC and the payloads. This will also reduce the mass and the power consumption of the satellite.
- The OBC however needs a built-in Mass Memory Unit (MMU) or needs connection to an MMU to store the generated mission product data.
- The OBC must have sufficient CPU performance to manage payload control and mission product data in parallel to platform control.
- The interfaces of the payloads have to follow certain data bus or interface standards (RS422, SpW or other) to avoid adaptation of the OBC design for each new mission. In particular the interfaces between payload and OBC as well as between OBC and MMU have to provide sufficient data throughput performance.

In conclusion this system design is recommended for missions and spacecraft where the payloads provide a minimum of intelligence to be equipped with a standardized digital interface and where the OBC is powerful enough and has the ability to handle payload data.

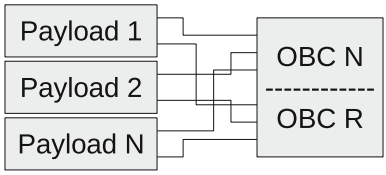


Fig. 9.1 Payloads connected directly to OBC. © IRS, University of Stuttgart

9.1.2 Payload Control via Dedicated PMC

The alternative solution is to manage the payloads via a dedicated Payload Module Controller (PMC)—see Fig. 9.2. This approach was followed for many years for large operational satellites with many instruments like the US NOAA satellites or on the European side the ERS-1/2, Envisat and the MetOp—A/B/C. In these satellites the platform OBCs—with the processor generations available at their time of design—were completely unsuitable to control the up to 12 (Envisat) payloads in parallel to platform management. And this was fact although all payloads themselves already featured a dedicated Instrument Control Unit (ICU).

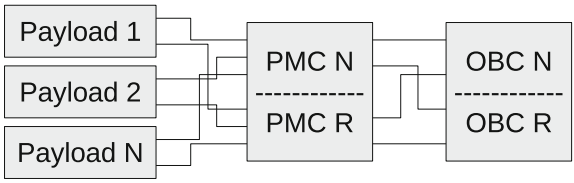


Fig. 9.2 Payloads managed via PMC. © IRS, University of Stuttgart

This concept is also suitable for small satellites and the system characteristics then will be the following:

- The interfaces of the payloads do not necessarily need a certain standardization. Payloads and PMC are
  - either designed together for one mission as complements to each other. This e.g. is the selected solution for the first FLP based satellite from IRS, the "Flying Laptop".
  - Alternatively the PMC is somewhat standardized but has a large scope of interfaces so that several payloads can be connected to it—and interfaces which are not needed for a particular mission simply remain unused.
- A satellite with a dedicated PMC will have a higher complexity due to the addition of another module. This also increases the mass and the power consumption of the satellite.
- The overall payload data handling subsystem complexity increases due to the additional PMC which is to be controlled by the platform OBC w.r.t. system initialization, boot process, reconfiguration and FDIR in general.
- In case of availability of a dedicated PMC the testing of the interfaces of the payloads to the satellite's bus system can be conducted independently from the OBC. Such tests allow to detect problems prior to satellite level FlatSat tests.
- Software for controlling payloads or handling payload data can be patched/updated in flight without interference with the satellite platform because no change to the OBC software needs to be applied.
- The MMU in most cases is connected to or is embedded into the PMC to store payload mission product data. This means the OBC will not have to handle payload data, which keeps this function and associated high speed data interfaces out of the OBC software.

In conclusion this design approach is recommended in case of a mission with multiple payloads which in addition—at least in part—are non intelligent ones and which do not provide standardized control and data handling interfaces.

In such a system design with dedicated PMC, the task allocation for payload control still provides some flexibility.

One solution is that the PMC takes over full control of all payloads and mission product data handling. The platform OBC controls a stable platform attitude like permanent Nadir pointing and the payload mission timeline is executed by the PMC

performing activation, mode control and deactivation of all payloads. The platform OBC only intervenes in case where system resources like power or attitude are lost. This approach is followed by the mentioned big satellites like Envisat, MetOp etc.

The contrary implementation is that the platform OBC performs full execution of the mission timeline and commands the PMC to perform certain payload switching which the PMC then translates to the low level payload electronics commands. The PMC in this approach is still the unit which fully manages the mission product data. This approach is the selected solution for the first FLP based satellite from IRS—the "Flying Laptop".

9.2 Payload Control on the First FLP Based Satellite

9.2.1 PMC Hardware

To distinguish between a general PMC and the very particular implementation of the first FLP mission, the PMC on board the IRS small satellite "Flying Laptop" is called "Payload On-Board Computer" (PLOC). The PLOC hardware architecture is based on the concept as depicted in Fig. 9.3.

It is implemented with complete single redundancy and is operated in cold redundant mode. Both, the MMUs and the processing nodes are redundant, where each processing node however has its own MMU. The processing nodes—called "Central Processing Node" (CPN) are based on FPGAs as besides payload mode

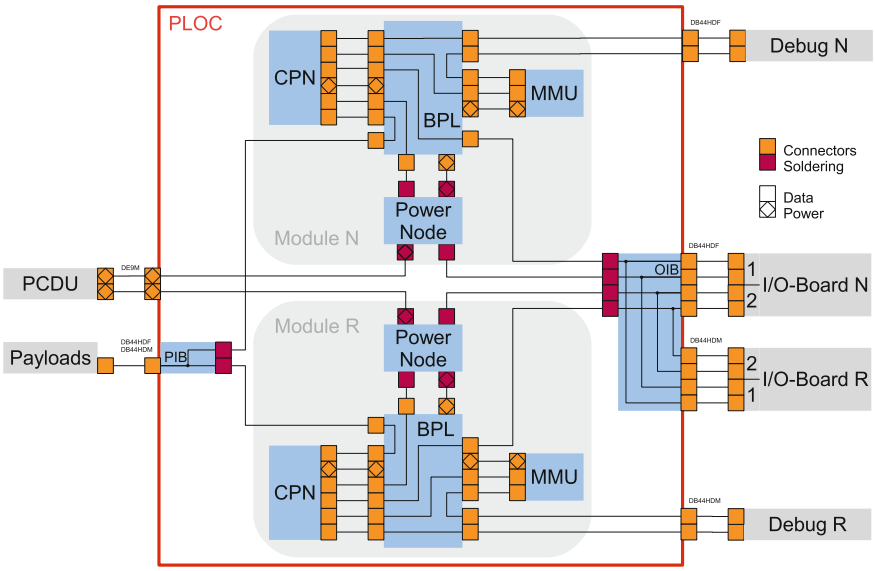


Fig. 9.3 PLOC electrical block diagram. © IRS, University of Stuttgart

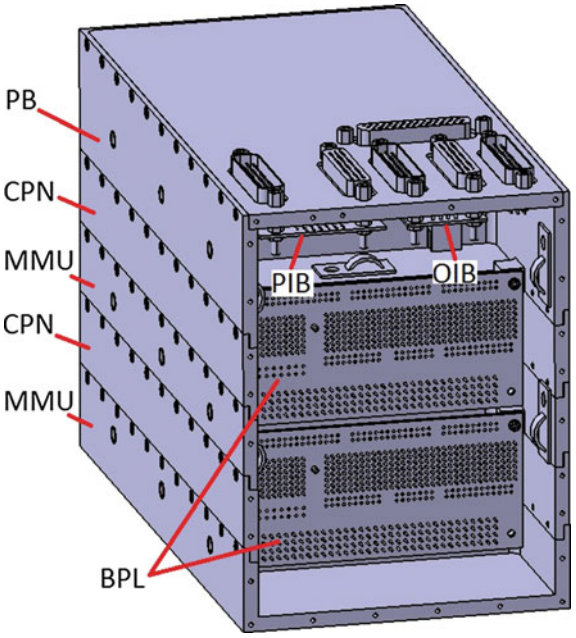


Fig. 9.4 PLOC catia model. © IRS, University of Stuttgart

Fig. 9.5 PLOC FM. © IRS, University of Stuttgart

control they have to preprocess/compress mission product data (e.g. camera raw data to JPEG image format).

The payloads on board the IRS "Flying Laptop" are three sensors and one "transmitter" in the widest sense—a space-to-ground laser terminal. All four payloads are connected to both sides of the PLOC for payload command/control. The mission product data lines of the 3 sensors—2 cameras and an AIS ship routing receiver—also are connected to both PLOC nominal and PLOC redundant.

Figure 9.4 shows the stack of PLOC boards in their cassettes. The topmost cassette holds the Power-Board which supplies all CPN and MMU elements with the required low-level voltages converted down from satellite power-bus voltage. The nominal CPN is mounted in the next lower cassette. As stated the CPN is based on an FPGA. The board has the following features:

- Main FPGA for routing commands and processing payload data
- Housekeeping FPGA for checking and commanding temperature alert, watchdog, boot-up of main FPGA, SW update management
- Flash memory chips to store up to three different images of the main FPGA SW
- Flash memory, SRAM memory and DDR RAM for storing payload data before submitting it to the MMU
- Multiple transceiver, receiver and transmitter chips for interfacing with the different payload interfaces
- Internal 10 and 80 MHz clocks

Following the CPN further down comes the nominal MMU cassette. The MMU in the PLOC consists of an FPGA and several flash memory chips. It can store up to 4 GB of data. Except for very few selected chips, the entire PMC and MMU are based on non radiation hard components for cost reasons. Thus a close monitoring of its health and proper functioning from ground is important once the satellite is in orbit.

The diverse interface printed circuit boards between CPN, MMU and external connection shall not be discussed in the frame of this platform FOM. The PLOC FM is depicted in Fig. 9.5 and further details can be taken from [135].

9.2.2 PMC Mainboard Elements and Function

As already mentioned the PLOC has two FPGAs on each CPN for computational tasks. Here the smaller one, the Configuration Controller (CC), consisting of a Spartan FPGA from Xilinx, is introduced.

The software for the Configuration Controller is stored on a radiation hard EEPROM onboard the CPN. This configuration of the FPGA was developed by "Fraunhofer Institut für offene Kommunikationssysteme"—FOKUS.[1] Since the Configuration Controller is only accessible by a connector directly on the board it is not possible to update its software in orbit.

[1]Formerly "Fraunhofer Institut für Rechnerarchitektur und Softwaretechnik"—FIRST.

The Configuration Controller itself is connected to the platform OBC via several dedicated status lines as well as via two differential lines for external communication using UART protocol. While the status lines are regularly checked by the platform OBC directly, the Configuration Controller is polled periodically by the platform OBC via the differential UART line, to send the CPN board and main FPGA temperature, the consumed current, the voltage and the general status.

The UART line is not only used for housekeeping purpose but can also serve for updating the CPN software of the main FPGA and for handling the different configurations of the main FPGA in flight during the "Flying Laptop" mission. The CPN houses three flash chips to save the configuration of the main FPGA. These memory banks are controlled by the Configuration Controller. The flash chips can be erased and new configurations for the main FPGA can be uploaded onto them. The new configuration can be loaded in segments which is essential considering that each configuration has a size of about 2.6 MB.

At start-up the Configuration Controller will automatically load one configuration onto the main FPGA which enables PLOC to be operational within one second. The PLOC can handle its normal payload control tasks in parallel to simultaneous loading of a new configuration in a non operational Flash memory.

9.2.3 PMC Software

The PLOC is one of the more intelligent PMCs. It is not only able to handle the communication between OBC and payloads but in addition performs payload data storage into MMU and can also perform mission product data preprocessing. The processing of data includes:

- Framing payload data into CCSDS Frames with Forward Error Correction in preparation for transmitting payload data to ground via a dedicated Data Downlink System (DDS) on board the "Flying Laptop".
- Framing payload data with Forward Error Correction for transmitting them to ground via the Laser Link Terminal OSIRIS (payload 4).
- Autonomous triggering of the camera systems for calibration purpose including evaluating of received images.
- Quality analysis of images concerning usability.

9.2.3.1 PMC Command/Control

The PLOC mainboard is the CPN and the CPN's main processing FPGA is a XILINX Virtex II Pro (VTX). It manages all communication between OBC and Payloads as well as processing of all payload data and their routing to MMU

Table 9.1 OBC PLOC communication message

STX	Device ID	Module ID	Message ID	Data	CRC	ETX
1 Byte	1 Byte	1 Byte	1 Byte	N Byte	1 Byte	1 Byte

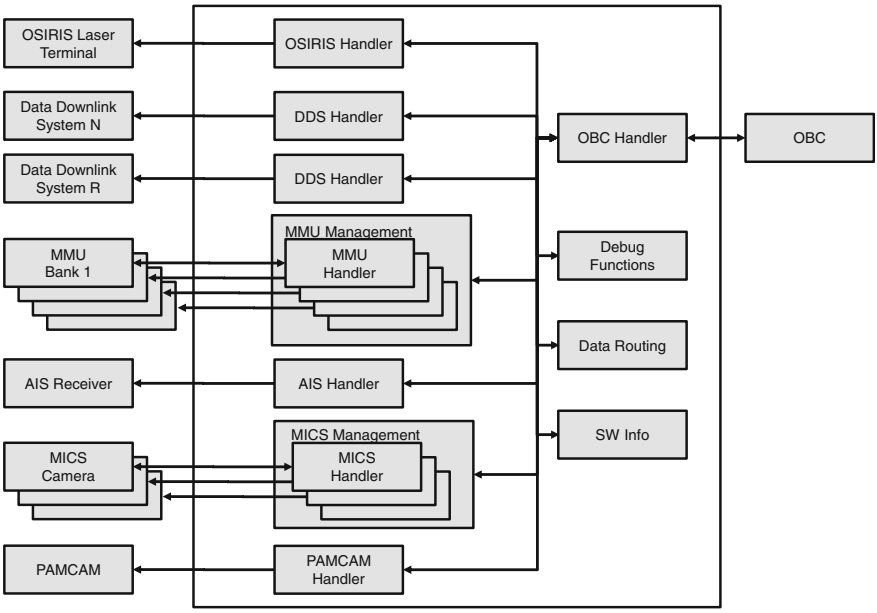


Fig. 9.6 Command/control architecture of PLOC software. © IRS, University of Stuttgart

respectively to the Data Downlink System. For communication between OBC and VTX the following message frame is used (on-board TM, see Table 9.1).

On VTX this message will first be routed to their appropriate payload handler, via the Device ID and the Module ID, where an action, according to the Message ID, will be triggered—see also Fig. 9.6.

For each payload there exists one device handler. In some instances there are multiple modules of one device like the multiple color channels of the multi-spectral camera payload MICS or the different lanes of the MMU. These can be addressed via the Module ID. For all internal "devices" like SW Info, Data Routing, Debugging etc. there exists a dedicated device ID. The module ID serves for identifying to which of these entities the command has to be routed. This is in compliance with the platform OBC software architecture where each device has multiple states, which would not apply to internal entities on the VTX FPGA.

Figure 9.6 shows the commanding architecture of the PLOC software. As can be seen in this diagram, there exists a command route to and from the OBC handler to every other main entity in the software.

The architecture—as being implemented on an FPGA as network of gates—is capable to compute different processes in parallel. This means that the OBC handler is able to accept new commands while others are still being processed. The OBC handler can also handle multiple responses from different devices at the same time.

9.2.3.2 PMC Internal Mission Product Data Handling

A matrix structure of all data sinks, data sources and data buffers is used to transfer payload data from one entity of the top level to another. During operations the matrix can be re-arranged so that each data sink has a single data source. A data source however can have several data sinks. By this means data can also be stored in multiple sinks with redundancy. Figure 9.7 depicts an example configuration where all data coming from the sensor payloads—the multi-spectral camera MICS (red channel as example here), PAMCAM or AIS receiver—are processed on the FPGA in parallel.

All data is handled by pipelining them from one entity to another. If a data sink—like an MMU bank, the DDS transmission system—is busy it signals the source to stop sending data to prevent data loss. As this will not work for payload hardware like a camera or the AIS receiver, intermediate SRAM buffers are implemented to guarantee a continuous data input sink for the hardware generated signals. These synchronous SRAM elements act like FIFO buffers.

As with the commanding architecture, all data sinks and sources have a standardized interface this means that a new entity for data handling can easily be implemented. E.g. an entity for formatting data into CCSDS frames using forward error correction can be implemented to the top level and added to the data matrix. Then the encoding mechanism for example can be placed between SSRAM 3 and DDS N depicted in Fig. 9.7.

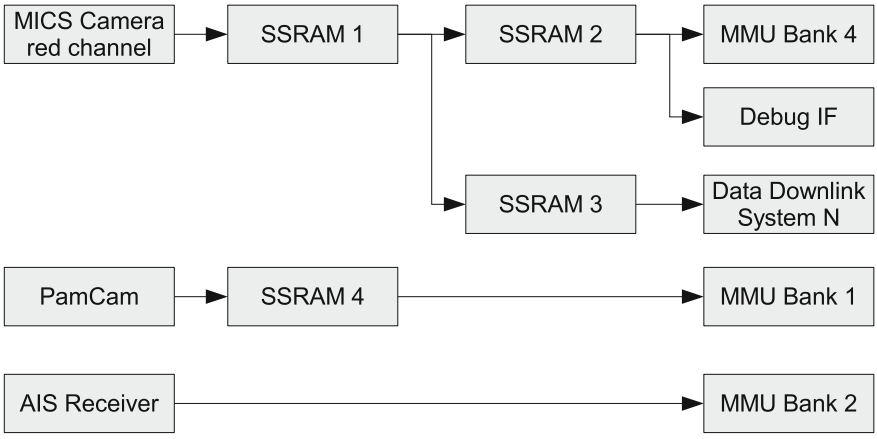


Fig. 9.7 Configuration example for parallel data handling on PLOC. © IRS, University of Stuttgart

9.2.3.3 PMC Error Correction Functions

As there are no payloads on board the IRS "Flying Laptop" which are critical with reference to failure management in the sense that misoperation could damage the payload, the payload FDIR is reduced to pure power and thermal monitoring of the payloads through the platform OBC. Please also refer to Sect. 10.8.

The following failure detections are implemented:

- CRC check during communication between OBC and PLOC as well as between CPN and MMU
- Automatic triggering of panoramic camera PAMCAM (payload 1) initialization to check successful startup
- I2C data-bus acknowledgment check between CPN and MICS camera
- Timeout between CPN and all other devices which will send a response
- Message ID check on each module handler
- Triple module redundancy implementation of OBC handler with a voter to ensure communication is working under any circumstance

If a failure is detected, a corresponding error message is submitted to the OBC. In addition to those automatic detections the following checks can be initiated manually:

- Query of the VTX software Version (works as a simple ping).
- Query info about a certain handler (e.g. amount of data in RAM, amount of messages received by AIS receiver handler, etc.).

9.2.3.4 PMC MMU Functions

The MMU can be accessed via four parallel interfaces. Each line has access to two Flash memory chips with a capacity of 500 MB each. The memory is structured into 250 equally sized sectors. On command from CPN the MMU can save, read or delete data sets stored in one of these sectors. The MMU has three modes in which it can save data.

- Mode one will save all incoming data until there is a break in the incoming data stream.
- Mode two will save all incoming data until an abort command is issued or the chip is full.
- Mode three will save a user defined amount of data.

As result of complexity reduction of the MMU code the MMU needs to be told the address of the memory sector into which it has to store the data.

9.2.3.5 PMC Automated Control Functions

One autonomy function of the PMC is, that images taken from any camera have to be checked concerning their brightness. Too bright or too dark images shall be discarded on board and a second one shall be taken with a different exposure time. If the alternative image is considered acceptable by the onboard algorithm the image data is stored.

The "Flying Laptop" satellite has the ability to collect AIS signals from ships with its AIS receiver (payload 3). Simultaneously it is possible to scan the Nadir field of view for ships by using the MICS camera system—provided the view is not obstructed by clouds. For such a camera based observation the PLOC has to trigger the MICS in regular intervals without guidance from OBC.

Since neither the MMU nor the payload data link budget allow storage of a full time continuous image recording, the images must be checked on board the satellite whether ships can be identified on the received image. E.g. cloud obstructed images will be identifiable due to their low contrasts and can be sorted out. All other images are stored into MMU.

The PLOC automatic operational mode can be activated by command from ground via the platform OBC.

9.3 Payload Control in Network-Centric Architectures

This chapter explains some limitations of the OBC FLP Generation 1 architecture w.r.t. coupling of avionics and payloads and describes the OBC architecture upgrade, which Airbus DS is pursuing with its partners IRS, Uni Stuttgart, Germany, 4Links Ltd., UK and Satellite Services B.V., NL for the FLP Generation 2. The architecture baseline is a SpaceWire network.

This development is also targeting towards the integration of professional SpaceWire based Mass Memory and Formatting Units (MMFU) in contrast to the simpler MMU embedded in the PLOC on board the FLP Generation 1.

Also the integration of dedicated high performance computing nodes for payload data preprocessing on board the spacecraft is considered in this network centric architecture.

To explain the final approach, this short chapter takes the reader through a sequence of examples of SpaceWire networks, starting with the network used on the FLP, and adding SpaceWire routing switches to provide improved connectivity and performance while retaining the fault tolerance of the FLP architecture.

9.3.1 FLP Generation 2 SpaceWire Network

Figure 9.8 shows the SpaceWire network as in the FLP Generation 1 design and in the "Flying Laptop". There are nominal and redundant pairs of OBC Processor-Boards—also called Single Board Computers (SBCs), the platform ground/space link boards (CCSDS Boards), and the I/O-Boards—also called Remote Interface Units (RIU). These are cross-strapped with SpaceWire links inside the OBC so that, even if one of each of the SBCs, CCSDS-Boards, or RIUs is broken, the remaining one of each pair is connected and the system can continue to operate.

The payload computer and its instruments as well as the spacecraft avionics units in Fig. 9.8, however, are connected to the RIUs via a variety of different interfaces, most of which are very slow. Even if some of the data could be much faster, all such data would need to be processed by software on the SBC, which itself would limit throughput. This is a suitable design for a small satellite with low performance instrumentation, but the limitations are obvious.

Figure 9.9 shows the same SBCs, RIUs and Downlinks as Fig. 9.8, but adds a single SpaceWire routing switch. Paths via the routing switch connect all the units together so that again, even if one of each of the SBCs, Downlinks, or RIUs is

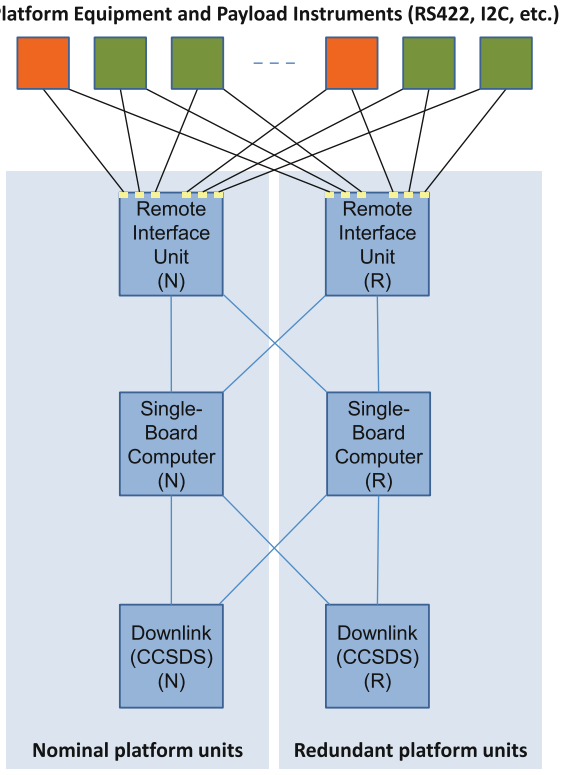


Fig. 9.8 Simplified block diagram of the FLP generation 1 architecture. © 4Links Ltd

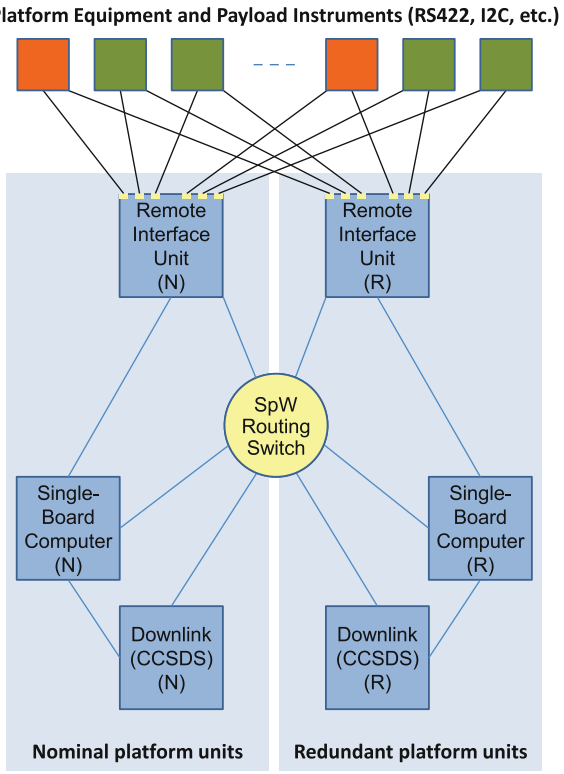


Fig. 9.9 Payloads and avionics on a single routing switch. © 4Links Ltd

broken, the remaining one of each pair is connected and the system can continue to operate. This might be seen as a single point of failure but, if the routing switch is broken, there are still direct connections between the SBCs and the RIUs and downlinks.

If the routing switch and the nominal RIU and Downlink were broken, the redundant RIU and Downlink would be able to operate (and similarly if nominal and redundant were reversed).

While this might be seen as a slight reduction in fault-tolerance compared with the cross-coupling of the network in Fig. 9.8, a benefit from the routing switch is that higher-speed data could now be passed through the routing switch without going via the SBC. This might need a small increase in the functionality of the RIU and/or the Downlink, but it is possible that the RIU could be instructed to send SpaceWire packets to the Downlink rather than back to the SBC. The routing switch also provides a potential connection between the two SBCs.

Figure 9.10 shows the single routing switch replaced by a pair of five-port routing switches. Loss of the redundant routing switch or SBC still enables the nominal SBC to access either RIU and either Downlink, and loss of the nominal SBC and redundant routing switch leaves the redundant SBC able to access the redundant RIU and Downlink either with the direct links or via the nominal routing

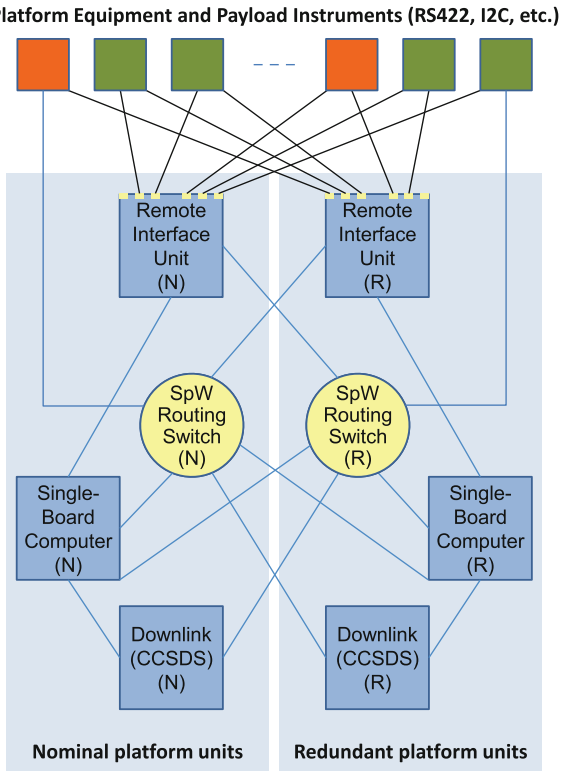


Fig. 9.10 Payloads and avionics coupled via redundant routing switches. © 4Links Ltd

switch. The nominal RIU has a path via the redundant routing switch to the nominal Downlink, (and, again, similarly if nominal and redundant were reversed).

Furthermore this implementation opens up the opportunity to connect SpaceWire equipped payload instruments or intelligent SpaceWire equipped avionics units like GPS or STRs.

Using routing switches with more links, and/or dispensing with the direct connections from the SBCs, might offer improvement.

Figure 9.11 shows what is perhaps the simplest and most regular network, with two routing switches, each connected to all the units that are interfaced by SpaceWire, in the same way as the single routing switch of Fig. 9.9.

As in the network of Fig. 9.8, either RIU can be broken, either SBC can be broken, either Downlink can be broken and (in this case) either routing switch can be broken, yet the remaining units are fully connected and the network will continue to work. There are also redundant paths between the RIUs and the Downlinks (which may be useful for high-data rate instruments), and between the two SBCs for more computing capability (when there is a need and there is adequate power).

Figure 9.12 now for the first time depicts a design where instruments and avionics are separated and where instruments are interfaced exclusively with SpaceWire—together with a Mass Memory and Formatting Unit (MMFU),

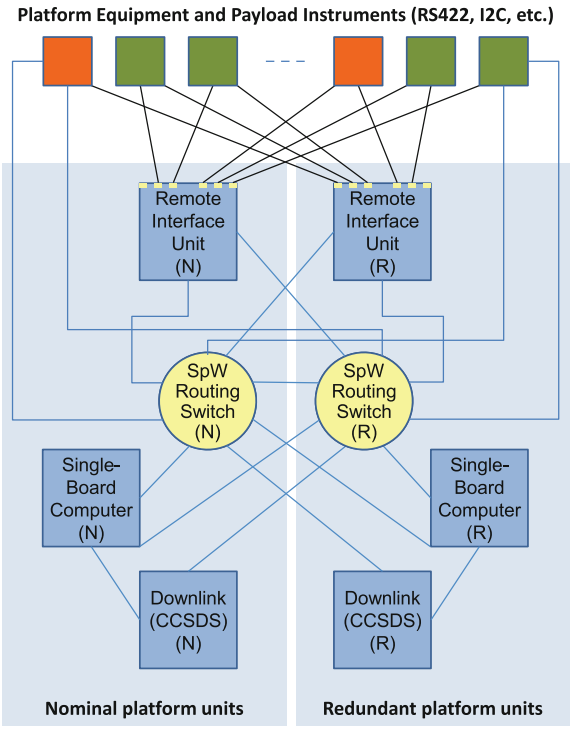


Fig. 9.11 Routing switches with more links improving redundancy. © 4Links Ltd

a Payload Data Processing Unit, and a routing switch to connect these additional components to each other and to the platform routing switches—via which they could be controlled by the SBCs. The payload sub-net additionally can include a fast science data downlink equipment transmitting in X-Band or Ka-Band. The clear separation of the SpaceWire networks also permits to run both sub-nets—avionics and payloads net—with different SpaceWire clock speeds. The platform avionics usually will be fully satisfiable with a SpaceWire data rate in the range of 10–15 MBit/s while the payload sub-net presumably will require much higher throughput performance in the range of 100 MBit/s.

In Fig. 9.12, each Payload instrument still only has a single SpaceWire link. While the Payloads perhaps have less need for redundancy than the Platform, it may often be preferable for the Payloads each to have two SpaceWire ports, and to add another routing switch, as shown in Fig. 9.13.

There are many variations on the network of Fig. 9.13. There may be some payload instruments with a single SpaceWire port as in Fig. 9.12. The Mass Memory and Formatting Unit and the Payload Data Processing Unit's functions might be combined into a single unit, and/or there might be redundant copies of them to provide fault-tolerance and/or higher throughput of payload data.

There may be many more payload instruments, which could be handled either by routing switches with more ports, or by using several four-port or eight-port routing

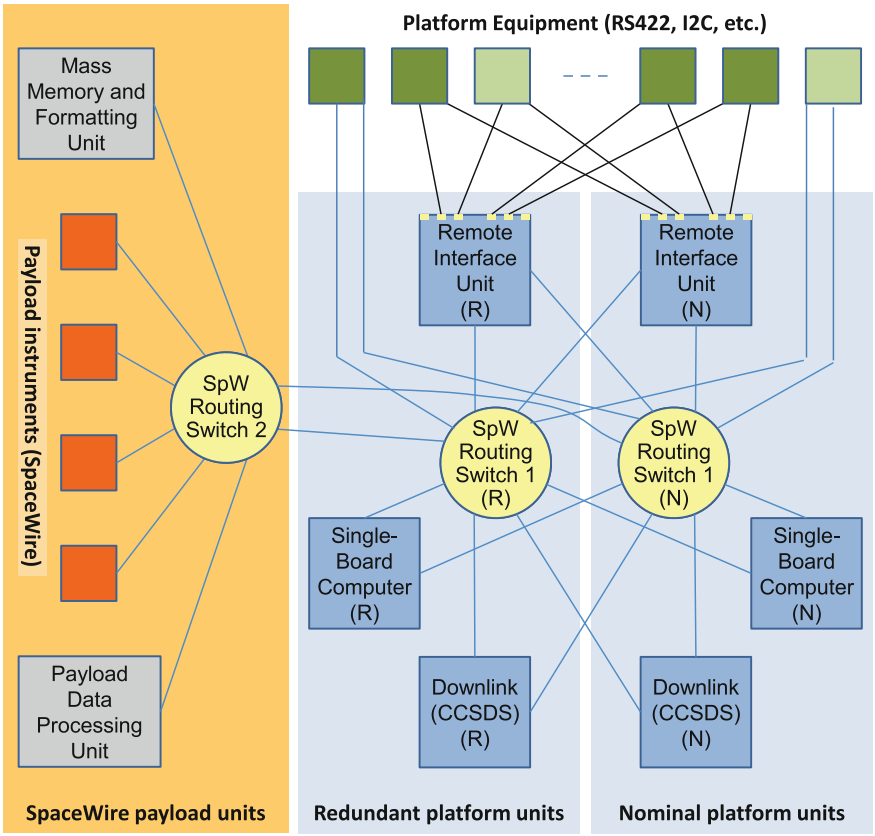


Fig. 9.12 Payloads coupled via separate routing switch. © 4Links Ltd

switches to make a larger routing switch. For some satellites, two routing switches with many SpaceWire ports, could each connect, as in Fig. 9.11, to all the SpaceWire units and/or instruments—whether platform or payload.

There is often a trade-off between using larger routing switches to minimize the number of SpaceWire links, or using more, smaller, routing switches which would need more SpaceWire links but could provide improved fault-tolerance and/or lower total cable/harness mass.

9.3.2 FLP Generation 2 Target Architecture

From Fig. 9.13 it is clearly visible that each Processor-Board—or "SBC"—in such an OBC has to feature a sufficient number of SpW interfaces—or preferably should feature a routing switch directly on the board.

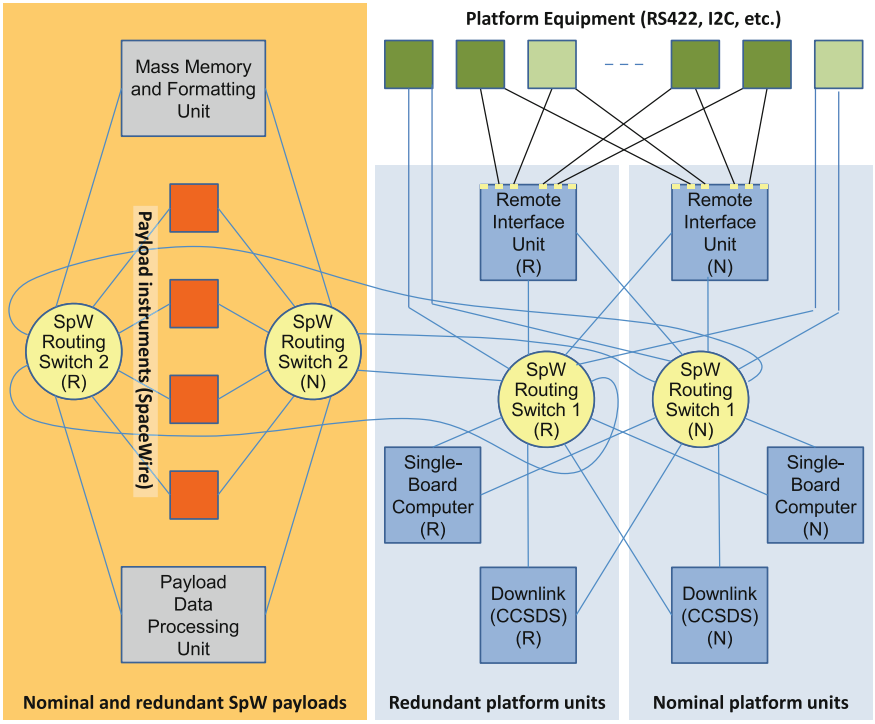


Fig. 9.13 Payloads in separate SpW sub-net. © 4Links Ltd

A particularly elegant design is followed by Airbus DS and its partners for an according upgrade of the FLP Generation 1 OBC. As it was already published in [146] the design foresees to use a Processor-Board featuring a dual-core LEON3FT processor (a Cobham Gaisler GR712 [74]) plus an 8 port SpaceWire routing switch directly mounted on the board (see Fig. 9.14). The Processor-Board (iOBC-A1-LB01-2.0—see [75]) comes from the consortium partner Satellite Services B.V., Katwijk, Netherlands.

The board offers the possibility to connect via the on-board SpaceWire routing switch the classic CDPI units like CCSDS-Boards and I/O-Boards (digital RIUs) plus SpaceWire equipped avionics units like future GPS receivers, STRs, IMUs or similar.

As the GR712 processor features additional SpaceWire interfaces, the payload chain can be connected via a dedicated separate SpaceWire network—which then also can be run at a different clock speed than the avionics sub-net.

In this architecture the payload sub-net routing switches come from 4Links Ltd., Milton Keynes, UK who already produced the I/O-Boards (hardware and firmware) and the CCSDS-Board hardware for the OBC of the FLP Generation 1.

This FLP Generation 2 OBC design—although only featuring one processor per board—still realizes the concept of a PMC-managed payload system

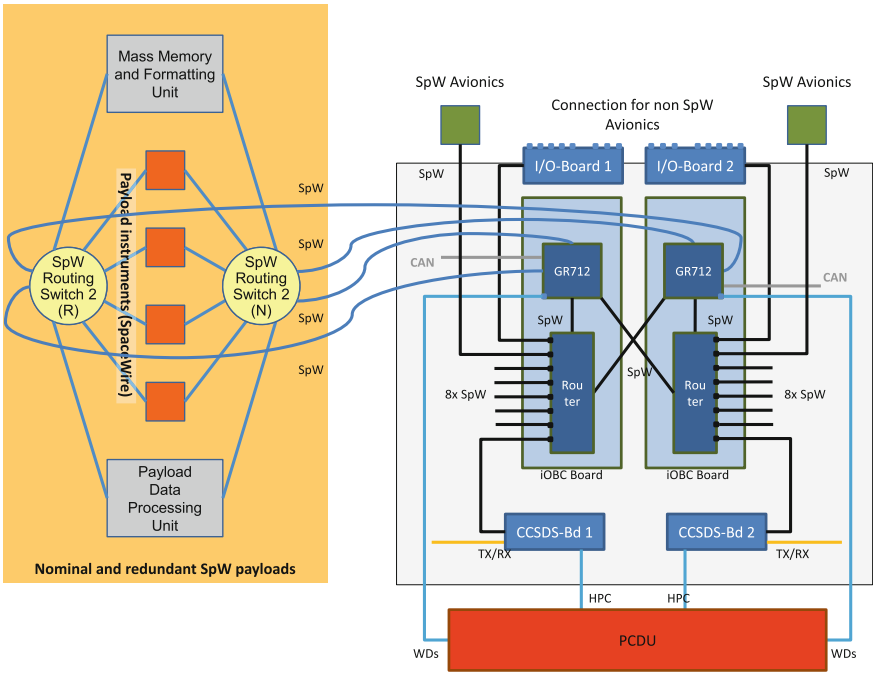


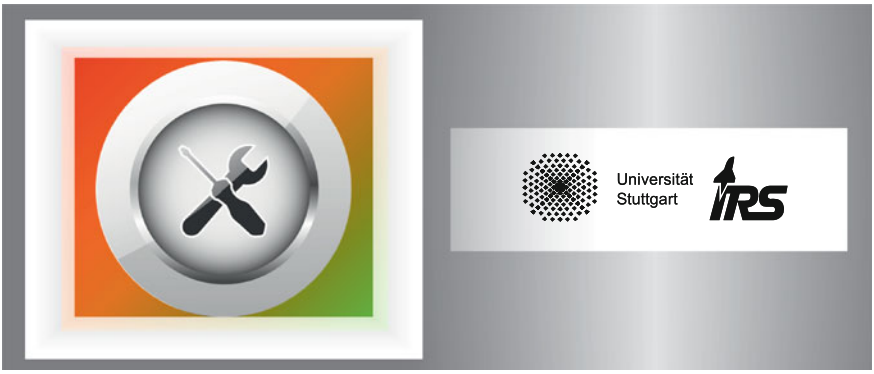
Fig. 9.14 FLP generation 2 system outline. © Airbus DS—from [146]

(please refer to Fig. 9.2), as the GR712 is a dual-core processor. In the design presented in [146] the avionics control software will be run on one core of the GR712 and will be a close derivative of the FLP Generation 1 OBSW. The payload control part will run on the 2nd processor core. Cross-communication from the Core Data Handling to the payload control can be achieved rather easy via the GR712's internal inter-processor bridge.

For the payload data processing chain Airbus DS is aligning this development with an initiative running in cooperation with DLR, where various high speed payload data processing computer architectures are evaluated—like LEON4, PowerPC and ATOM. For more details on this study called "On-Board Computer System-Architektur" (OBC-SA) please refer to [148, 149].

Chapter 10
Failure Detection, Isolation and Recovery Concept

Rouven Witt and Jens Eickhoff



Abstract This chapter is one of the most complex ones and covers the platform Failure Detection, Isolation and Recovery (FDIR). It starts with explaining the FDIR concept and the system redundancies. Next FDIR Events, their flow in the onboard software and their management are explained. The spacecraft reconfiguration functionalities are treated in detail, in particular the reconfiguration of the onboard computer elements and the power subsystem as well as the mode fallback hierarchy for the diverse failure types. The FDIR specifics for each subsystem are explained and also the handling of device hardware failures. Software constants and FDIR limits are provided in the book's annexes.

Keywords Failure detection, isolation and recovery (FDIR) · FDIR concept · Redundancies · FDIR events · Spacecraft reconfiguration · FDIR techniques for all subsystems · Device FDIR · Failure propagation · Satellite safe mode

R. Witt (✉)
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany

J. Eickhoff
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_10

10.1 General Principles

10.1.1 Applying ECSS Terminology

The goal of on-board Failure Detection, Isolation and Recovery (FDIR) is to

- detect failures or malfunctions in devices, assemblies, subsystem or system control
- to isolate the failure root cause to avoid propagation and endangering the mission through follow-up effects and
- to initiate according recovery actions automatically or
- to inform the ground operator that an unrecoverable state has been reached.

Requirements can be expressed formally applying the ECSS-E-ST-70-11C [14] standard for Space Segment Operability. In this standard, different forms of autonomy are assigned to a level number between 1–4, where level 4 corresponds to systems being almost fully capable to perform autonomous operations, for example through automatic re-planning. Different fields that are considered for the autonomy definitions are represented by a letter code in the ECSS standard. Combining the letters and numbers, the pursued or achieved autonomy of a spacecraft can be defined. In the following, the FLP platform goals are brought in reference with [14]—expressing the mission autonomy level for mission execution, data handling and fault management:

Mission Execution

- **Level E2** Execution of pre-planned, ground-defined, mission operations on-board. Capability to store time-based commands in an on-board scheduler.
- **Level E3** Execution of adaptive mission operations on-board. Event-based autonomous operations.

FLP OBSW and FDIR will not be able to plan or re-plan an operational scenario on their own. Such a level of autonomy is beyond the goals of typical FLP based missions. Most mission scenarios will be realized via predefined mission timelines. However, the mission scheduling functions have to be able to react to certain events that are preconditions that have to be fulfilled before a sequence can continue.

Data Management

- **Level D2** Storage of on-board of all mission data, i.e. the space segment is independent from the availability of the ground segment.
  - Storage and retrieval of events and reports (as from D1).
  - Storage management (as from D1).
  - Storage and retrieval of all mission data.

There is a dedicated memory area in the FLP OBC I/O-Boards that is used to store all relevant HK and event data. The storage of HK telemetry will be

configurable from ground through PUS services to allow adaption to particular phases of the mission. During LEOP it will be necessary to store device HK with a higher frequency to allow a detailed observation of on-board processes. Only Safe Mode components are active in LEOP.

During the Commissioning Phase, HK and Event TM of to be commissioned devices will be stored in detail to allow their thorough observation on the first start-up. After successfully completing these critical initial phases, many different TM parameters will be stored due to many active devices during normal operations, but their storage frequency will be significantly reduced.

Fault Management

- **Level F2** Re-establish nominal mission operations following an on-board failure.
  - Identify anomalies and report to ground segment (as from F1).
  - Reconfigure on-board systems to isolate failed equipment or functions (as from F1)
  - Place space segment in a safe state (as from F1).
  - Reconfigure to a nominal operational configuration.
  - Resume execution of nominal operations.
  - Resume generation of mission products.

The main mission statement to be considered for the design of the system FDIR for the FLP is "Single Failure Tolerance", throughout all operation. This means any one single failure occurring on-board must not lead to the loss of the ability to perform operation or even loss of the entire mission. "Reconfigure to a nominal operational configuration" has to be accomplished, which is the main task of the FDIR system. The Safe Mode is the standard fall-back mode for severe events. However, if possible, the operation in Idle Mode or any higher operational mode are to be continued after one failure (see also fallback transitions in Chap. 2 and Fig. 2.1).

10.1.2 FDIR Requirements

The idea of the FLP FDIR concept and implementation follows the paradigms that the OBSW is built on. As described in Sect. 3.1 one of the main ideas of the OBSW implementation as object-oriented is the encapsulation of knowledge. Specific information is encapsulated in dedicated objects. For example a *deviceHandler* stores information about how the communication to its respective device has to look like. This information is not stored anywhere else in the system. All communication will be routed through that *deviceHandler*, except for emergency reasons when direct ground interaction is required.

While only the device handler (DH) knows how the communication with its device has to look like, only the *deviceHandler* object has the ability to decide

whether communication with the device was successful or not. If a *deviceHandler* detects that there was a failure in the device communication, the data from the device will be marked invalid in the *datapool*.

If measurement values of a device are marked invalid, the corresponding subsystem controller will not use these as input data. Instead it will use other available data or will signal that it cannot perform the intended operation or algorithm.

For most of the controllers, a sensor fusion or sensor sanity check will be performed in the controller as shown in Fig. 3.9. If a *deviceHandler* could not detect a failure in its device communication, the protocol was correct. However, the measurement values might still be corrupted. Hence, the sensor sanity check and sensor fusion part of a controller will detect if values are drifting strongly or exceed a limit and will mark the according values as invalid. The controller algorithm again will perform the computation with other available measurement values or will signal that it cannot perform the intended operation. It will most likely also signal that it cannot keep its mode and will be set to the next fallback mode.

This approach has the advantage of not triggering Events of a single failing device on several different levels. FDIR does not need to correlate different Events that point to the same fault because the affected *deviceHandler* and *controller* (CTR) should not mark the same device as broken at the same time. This also reduces the danger of spamming the Event TM packet store.

If a device is reported as faulty, FDIR will attempt to power down the device but otherwise keep operations going. Depending on the device this might be possible if it is the first failure of that kind. After some time the device will automatically be powered up again by its *deviceHandler* and will be re-included into nominal operation. Depending on its behavior, the device will continue operation, or if malicious again, the device will be powered down permanently by FDIR until further examination has been performed by ground. The first failing device should not need to force a transition of the entire spacecraft into Safe Mode.

The goals of the FDIR are

- to detect invalid sensor information,
- the identification of faulty devices,
- the emergency device switch off,
- the S/C reconfiguration to continue an ongoing operational scenario,
- and to inform the ground about taken actions and identified occurrences.

FDIR cases that require a Safe Mode switch are reached when

- the software stops running and resets itself or stops running at all,
- the communication between OBC and PCDU is interrupted such that the PCDU triggers reconfiguration of the OBC,
- several redundant devices are faulty,
- the temperature of Idle Mode devices leave their specified range and need to be powered down,
- the SoC of the battery drops significantly or if

- a fuse switch is triggered automatically in the PCDU due to a malfunction or short in the electrical system.

In case where there is no way to continue nominal operation, the system will switch to Safe Mode with coarse sun pointing and minimized power consumption. This mode will be held until the spacecraft conditions have improved. As soon as the recovery can be confirmed by ground, instructions from ground will be required to switch back to nominal operations from that point. The system will not conduct a switch to a higher mode by itself. In Safe Mode, only the following components are active:

- OBC (Processor-, IO-, and both CCSDS-Boards)
- PCDU,
- TTC (Transmitter, and both Receivers),
- MGM (nominal and redundant),
- MGT (both coils during detumbling) and
- Sun sensors (connected via PCDU).

In case of a catastrophic power emergency, meaning the battery SoC has dropped below a critical level, the entire satellite will be set to "Sleep Mode" with all devices powered down and only the PCDU observing the battery State-of-Charge, equal to the PCDU/System Launch Mode. Recovery will be performed automatically by the internal PCDU controller after reaching a voltage level of 21.5 V through random tumbling and randomly pointing to the sun to continue operations until the MiniOps Mode is reached.

The core systems OBC and TTC will not be powered again before the system has reached this minimum power recovery status.

After OBC boot and OBSW boot the OBSW will automatically initiate the ACS Detumble and furthermore TCS Survival Mode will be entered attempting to reach the Safe Mode.

10.1.3 Fault Tolerance Through Redundancy

To achieve failure tolerance, all components that could fail during launch or operation of a S/C need to be available in redundancy. As the nature of things is that they all could potentially fail, all components (or at least all essential components) need to be available in redundancy. The realization of redundancy on board an FLP based spacecraft is described in Sect. 10.1.4.

However, it is not reasonable to have all systems powered and running at the same time. First of all, this significantly increases power consumption, which is to be avoided as far as possible, not only for a small satellite. Secondly, having two identical units operating at the same time (hot redundancy) might cause interferences on data connections.

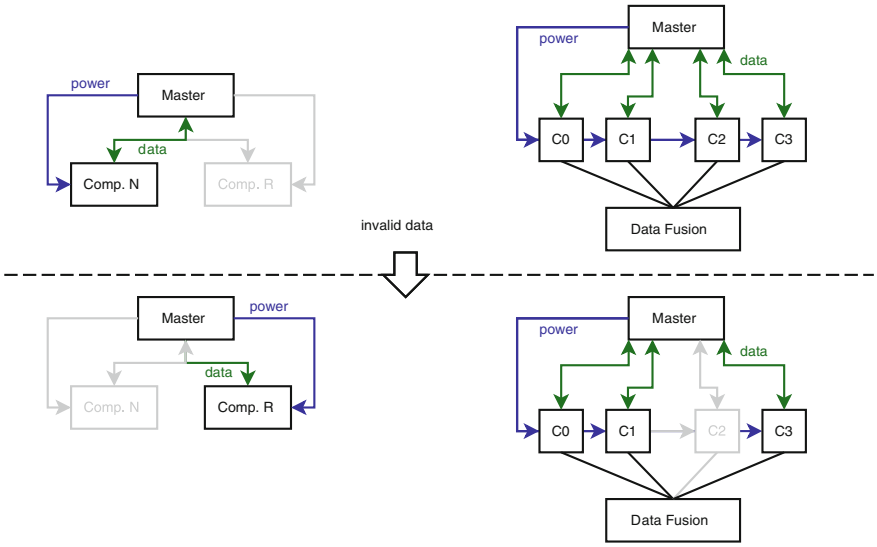


Fig. 10.1 Redundancy switch—cold redundancy and hot 3-of-4 redundancy. © IRS, University of Stuttgart

This means that depending on their importance and task, some units are to be operated in cold and some units in hot redundancy. For both cases there needs to exist an instance that observes the subsystem. This is illustrated in Fig. 10.1. In case of cold redundancy, an independent master unit needs to observe the currently running unit and to determine its correct function. If the correct function is no longer guaranteed, the master unit needs to power down the nominal unit and to power up the redundant unit. For hot redundancy, there needs to be an observing master unit as well, especially if a fusion of data is done. One unit that delivers malicious data could corrupt the complete subsystem's result. So a master unit needs to make sure that all components of a subsystem are working properly. If the correct function of one device is no longer guaranteed, the master unit needs to switch off that unit.

Consequently, this mentioned master unit—the instance that decides to switch units or to power them down—is the *FDIR* object in the OBSW. However, the art of FDIR is to determine, what input of or about a device can be accessed, and how this input can be evaluated to decide whether a device is working properly, or needs to be switched off.

Besides spacecraft FDIR in the form of hardware reconfiguration/hardware FDIR or software-initiated recovery actions/software FDIR, there will be FDIR cases that require ground interaction. This FDIR instance will be referred to as ground FDIR.

10.1.4 Redundancy on FLP

All platform main bus components are available in redundancy. The ACS Safe Mode components MGM and MGT exist in dual cold redundancy, meaning two equally designed units are available for operation. If a malfunction is detected, the redundant unit will be activated and the former active unit will be passivated or powered down. For Idle and Pointing Mode components RW, FOG, GPS and STR different redundancy designs have been selected. RW- and FOG-system are based on four units working at the same time in triangular configuration. A full 4 unit configuration is available in nominal case and the 3-out-of-4 voting is available in case of either unit failing. The erroneous unit can be identified directly and the loss of one unit can be compensated by the remaining three. The GPS receivers are available in triple-redundancy due to the purpose of the "GENIUS" experiment on the first FLP satellite—the "Flying Laptop". This experiment requires three hot redundant receivers. Pure position acquisition is also possible with less receivers—see later Sect. 10.6.2. The STR system is also based on two redundant camera head units, which will be operated at the same time for a higher accuracy of attitude knowledge but can be used in single mode as well.

For payload operations two payload computers (PLOC) and two Data Downlink Systems (DDS) are available in cross-coupling since they are essential in all payload operation scenarios.

During nominal operation, two hot redundant S-band frequency receivers are operated at all times to allow contact in case one receiver is not working properly. The transmitter will only be activated on-board when contact is expected, but can also be powered or unit-switched by ground via High Priority Commands if in range and if deemed necessary. These units' monitoring and redundancy management will be performed by the Onboard Computer.

As explained in Sect. 4.2 the Onboard Computer design is based on two cold redundant Processor-Boards and I/O-Boards and who hot redundant (communication pre-processor CCSDS-Boards). Cross-coupling of the Processor-Boards to the CCSDS-Boards and I/O-Boards allow switching of any of these units in case of a malfunction. If through a malfunction in either Processor- or I/O-Board the communication to the PCDU is interrupted, the latter will initiate a reconfiguration routine to restore OBC functionality. More details are provided later in Sect. 10.2.2.

The PCDU in itself has two internal microcontrollers and a watchdog self-check system to ensure proper operation. PCDU and OBC being the Combined Data and Power Management Infrastructure (CDPI) will be detailed in the following chapters.

10.1.5 Three Stages of Device Failure Detection

There are basically three stages of failure detection for devices depicted in Fig. 10.2:

- The first stage is control and observation of a device's power consumption. As described in detail in Sect. 10.3 the actual measured power consumption of devices is compared to their expected power consumption in the power control system. If a limit is violated, the device is reported to have an unexpected behavior.
- The second stage of device monitoring happens in the *deviceHandler* object. If a device sends a response that is formally incorrect, the *deviceHandler* issues a failure report. But—and this is a paradigm for the FLP FDIR—devices are not shut down necessarily upon their first formal monitoring violation. The mechanism for device failure handling is provided in Sect. 10.4.
- The third step of failure detection is encapsulated in the subsystem controller. Figure 3.9 illustrates that before device measurements are used to execute the control algorithms, a sensor fusion and sanity check is performed to determine their validity. The approaches for the third FDIR step for each subsystem are provided in the corresponding subsystem chapters later.

Whenever one of these three stages detects an unexpected behavior in one of the devices, an Event message is generated and the device is marked as faulty. This avoids that failure messages are generated in several stages of the device failure detection tree. Separating the failure detection for devices into these three steps pays results from the encapsulation of functions into the functionally corresponding OBSW objects. It decreases the complexity of failure handling and simplifies implementation and testing of the failure management system.

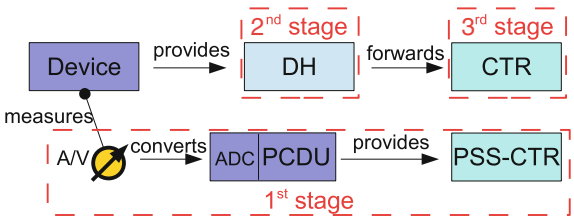


Fig. 10.2 Three stages of device failure detection. © IRS, University of Stuttgart

10.1.6 System Failure Detection

If a device shows an unexpected behavior and needs to be deactivated, the system will attempt to continue operations. As mentioned in Sect. 3.2.3 the knowledge about the device redundancy concept is stored in the subsystem assembly. This means that—as illustrated in Fig. 3.8—the assembly object checks whether there is

still a sufficient amount of devices to perform operations, otherwise it reports that it cannot keep its mode, which will lead to a switch to the next lower S/C mode. Exception of this behavior is the I/O-Board assembly which will always activate at least one instance.

System surveillance such as temperature and power consumption is performed in the controller objects. If temperature or SoC limits are violated, the controllers generate Events and FDIR will take according actions.

10.1.7 Event Utilization

The Packet Utilization Standard [13] defines four different levels of on-board Events, namely normal progress Events, and low, medium and high severity warnings. Event packets are generated by on-board processes identified by their object-ID. If the Event telemetry is activated in the event reporting service (see Sect. 2.4), the Events are also linked down as Service 5 telemetry with the sub-service indicating the Event severity. The Event severity levels are used on the FLP platform as follows:

Severity Level 0 (Normal Progress, INFO) Events indicate an *information* event initiated through nominal actions happening on-board for observation in the ground station. Possible Events show:

- Boot Event from software
- Successful powering of the payload computer
- Successful mode change of the system or a subsystem/device
- Heater activation and deactivation for thermal control
- System or subsystem mode switch.

Severity Level 1 (Low Severity, LOW) Events shall indicate that somewhere in the system, a failure has been detected. The failure will be transmitted to FDIR for further investigations or to determine actions. This type of Event can be reported from any level of OBSW objects (device handlers, assemblies, subsystems, etc.). Possible scenarios are:

- Communication failures with equipment,
- Malfunction of heaters, sensors, etc.

Severity Level 2 (Medium Severity; MEDIUM) Events will be generated by FDIR usually after processing of (several) Severity Level 1 Events and taking recovery actions. Possible scenarios are:

- A device is switched off due to failures.
- Devices have been switched off due to insufficient available power.

Severity Level 3 (High Severity, HIGH) Events will be generated by FDIR and indicate events of the highest possible severity. Events of level 3 indicate possible damage to the mission. Possible scenarios are:

- Battery State-of-Charge is low (Disconnection of non-essential loads).
- Safe Mode needs to be activated.
- Detection of a mission critical failure.

10.1.8 Event Flow

The FLP operation concepts is based on PUS Event Reporting to achieve visibility of processes executed on-board. Following this concept, all failure messages are assumed to be Events of a certain severity, and therefore, failure handling on board the satellite is based on Events, as well. This allows for a common way of handling all Events and makes it easy to create transparency of on-board processes for the ground operator.

In many former FDIR concepts, oftentimes a hierarchy has been realized to handle different kinds of failures on different hierarchy levels. Examples are cited in [158] and in [140]. A failure in such systems is solved on its level or is forwarded to the next higher instance if not solved. For the FLP implementation a different approach has been selected.

The basic concept of the FLP FDIR software implementation is that the detection of failures is distributed throughout all levels of software, all of which have different knowledge, and thus, can detect different kinds of failures. These failures lead to the generation of failure Events at the according point.

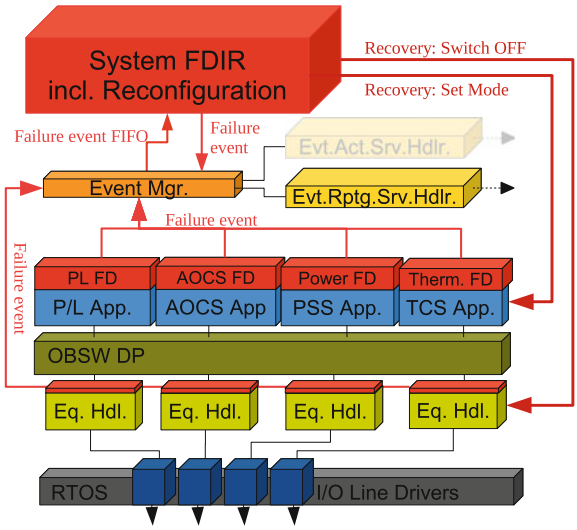


Fig. 10.3 Reduced diagram —Failure Event routing in OBSW. © IRS, University of Stuttgart

A centralized global FDIR instance is responsible for collecting all reported failure Events. Derived isolation and recovery actions will be performed by one centralized object after evaluating all incoming failure Events from one cycle. This is illustrated in Fig. 10.3 which is a reduction of Fig. 3.1, but with more details on the routing of failure events.

For example all formal knowledge about a dedicated equipment—such as for STR or MGM being their respective protocols, boot times, CRC mechanisms, etc—is stored in the corresponding *deviceHandler* software object. So being the first involved instance during device communication, these handlers detect formal errors or whether the device itself reports invalidity within their response message. These *deviceHandler* objects generate a failure detection event with PUS Srv. (5,2) Severity *low,* stating for example "MISSED_REPLY". All device handler objects have the ability to report such a failure Event. All failure Events will be queued up in a Failure Event FIFO where they are waiting to be processed. When a device answers formally correct no failures will be detected by these basic *deviceHandler* objects.

When a device delivers a correct response packet, but the delivered data is corrupted, this has to be analyzed on a higher level of knowledge. This kind of knowledge, such as value limits, redundancy or voting, etc. is stored in the controller sensor management as illustrated in Fig. 3.9. If a sensor is found to be erroneous, meaning the delivered value is not reasonable or voted to be wrong, a failure detection Event is generated. This follows the same principle as for the *deviceHandler* objects except that the knowledge is a different one. All respective objects with higher device knowledge have the ability to generate failure Events. These failure Events also have PUS Srv. (5,2) Severity *low* and will be queued up in the same Failure Event FIFO.

The flow of events can be taken from Fig. 10.3. All generated failure Events will be queued up in one dedicated Failure Event FIFO. This queue will be consecutively processed by the central system FDIR object. This FDIR object examines the Events one after another and determines the corresponding isolation and recovery actions that have to be taken for further system operation continuation or alternatively for a controlled switch to the spacecraft Safe Mode. For most of the possible failure Events, three approaches of handling are considered:

(a) power down a device after a defined number of confirmations
(b) instantly power down devices
(c) instantly switch to satellite Safe Mode

Case (a)

This approach is mainly applied with device communication. If protocol errors are reported, it is considered tolerable to occur from time to time during nominal operation due to environmental influence/disturbance. In such case, no immediate recovery actions are taken. Devices will be only be switched off, if a defined number of such errors is reported within a certain time period. Then, at least one restart is considered reasonable before marking a device as permanently non-healthy.

For device communication a distinction has to be realized since device errors might signal a misbehavior of the device itself. But it is also possible that a device errors is caused by a misbehaving OBC I/O-Board. Therefore, Events from devices will be stored before an actual evaluation takes place. The mechanism for device failures is explained in Sect. 10.4.

Case (b)
There exist situations and resulting Events that make the immediate switch-off of an equipment mandatory. An example would be the equipment temperature exceeding the permitted operational range and the Event COMP_T_OUT_OF_OP_RANGE. If such an Event is identified consequently the device will be powered down.

Case (c)
There will be Events that require an immediate switch to the satellite Safe Mode. An example is reaching of a critical State-of-Charge for the battery.

Events that can not be covered by options (a) through (c) might require particular reactions that can not be generally described and are introduced for their individual subsystems in following chapters.

10.1.9 Failure Event Management

All Events from one cycle will be processed one after the other. In contrast to former designs, only one instance is responsible and has the authority for the execution of recovery actions. This avoids several instances attempting to perform actions that have influence onto another.

An exemplary flow of Events of different severity is provided in Fig. 10.4. The flow considers the following case of a system transition to Idle Mode. As introduced in Fig. 10.3, all Events are routed to the *EventManager* and are forwarded to the configured targets:

- In the transition to Idle Mode, the *System* powers up Reaction Wheel 0.
- *DH::RW0* (→ reaction wheel *deviceHandler* object) reports a successful boot of the device with the Event BOOT_SUCCESS[rw0]. This is an information Event with severity level *INFO* and provides as reporting object the o*bjectID* of the *RW0* instance.
- *DH::RW0* attempts communication with the reaction wheel device.
- The communication is interrupted due to some failure and no response from the reaction wheel device is received.
- *DH::RW0* issues an Event MISSED_REPLY[rw0] with severity *LOW*.
- This Event will be forwarded to *FDIR* which decides to reboot the reaction wheel device. For easier tracking by ground, *FDIR* issues the Event REBOOT [fdir, rw0] which is stored in an Event package store. Additional information in this Event is the reporting object [fdir] and a parameter [rw0]. Since restarting is

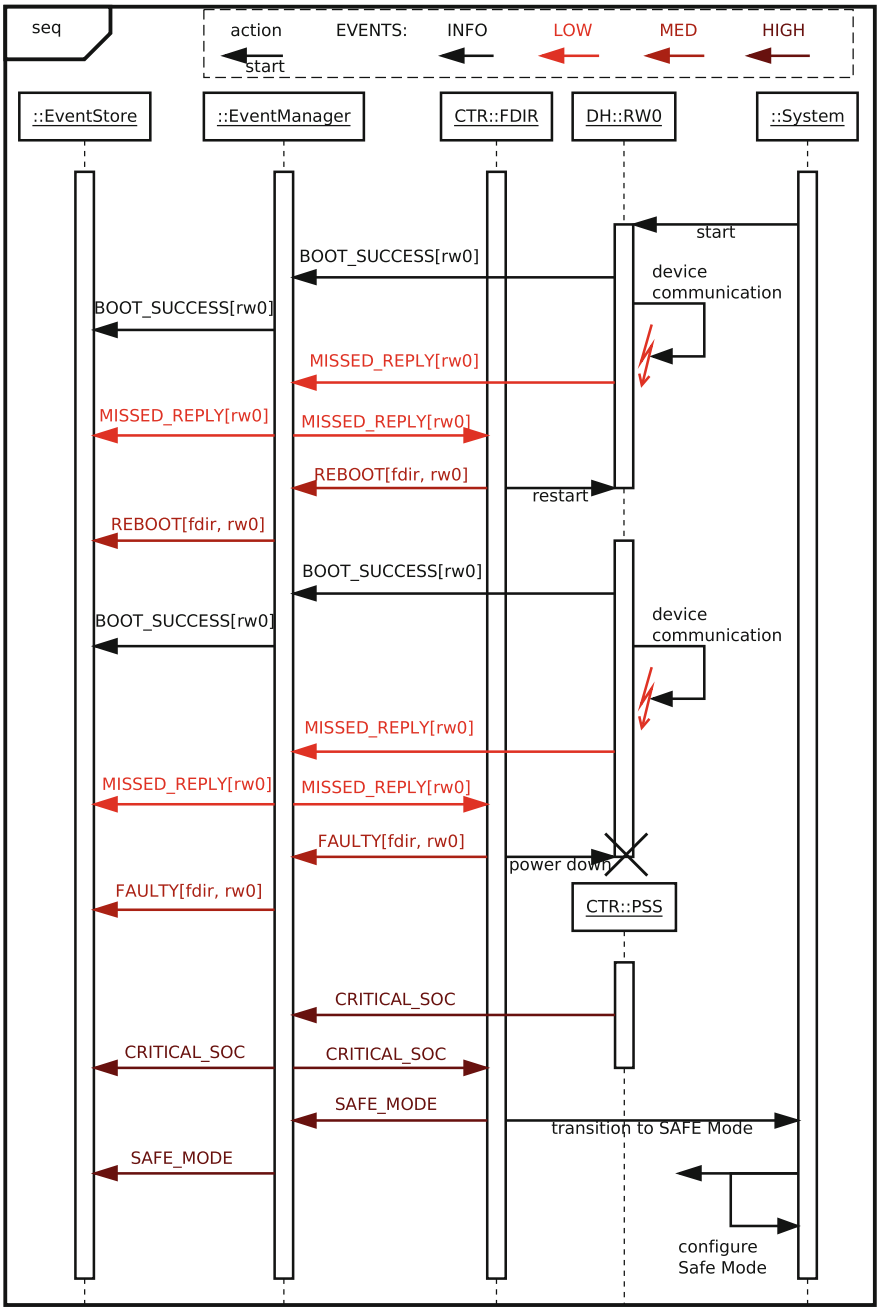


Fig. 10.4 Event flow sequence—Possible scenario. © IRS, University of Stuttgart

not a detection of a failure, but an isolation and recovery attempt, the Event REBOOT[fdir, rw0] has severity *MEDIUM* (compare Sect. 10.1.7).

Note: In the actual implementation, the FDIR instance includes a mechanism that ensures tolerance to sporadic interruptions and will not reboot a device at every interruption but only if device communication could not be established for a longer period (introduced in Sect. 10.4).

- Upon reboot of the reaction wheel device, the DH attempts to re-establish the communication. In the current example, the communication is still interrupted and the *DH::RW0* will issue the Event MISSED_REPLY[rw0] again.
- If the reboot of a device does not prove to be successful, *FDIR* decides that a device is damaged and will power it down permanently. It also generates the Event FAULTY[fdir, rw0]. This is an isolating action and has severity *MEDIUM*. The health state of reaction wheel 0 will be set to *UNHEALTHY*. This state can only be re-established through interaction from ground.
- Reaction wheels 1, 2 and 3 are assumed to work fine.
- The system will still be able to operate in Idle Mode at this point.
- At a later point, the power controller *CTR::PSS* detects a low State-of-Charge in the battery and generates the Event CRITICAL_SOC with severity *HIGH*.
- The Event will be forwarded to *FDIR*.
- *FDIR* decides that the system power consumption needs to be reduced and commands the *SYSTEM* to transition the spacecraft into Safe Mode. For better visibility by ground the *FDIR* logs this action with the generation of the Event SAFE_MODE. This Event has severity level *HIGH* (*called "Software-induced Disconnection of Non-Essential Loads"—SW-DNEL*).
- *System* will automatically command the system to Safe Mode. It shuts down all payload devices and all Idle Mode devices.
- All Events are stored in the *OBSW EventStore* which is a part of the non-volatile memory on the I/O-Board used as TM store. They will be transmitted to ground upon next contact before HK TM is sent.

10.2 Core DHS FDIR

As described in Sect. 10.1.4 all redundancy management for spacecraft equipment except for the PCDU is performed by the OBSW. However, if the OBC itself stops working properly, a proper recovery solution has to be applied. In most industrial spacecraft, a reconfiguration unit is installed in the OBC, that serves as observer for the OBC to function properly. In case malfunctions are detected, the OBC is recovered by switching to redundant OBC elements. Due to reasons of complexity a different approach was developed for the FLP Onboard Computer as was shortly mentioned in Sects. 1.8 and 4.2 and is described in full detail in [47]. The Core DHS hardware here is implemented as so-called Combined Data and Power

Management Infrastructure (CDPI) with shared reconfiguration controller for OBC element reconfiguration and PCDU reconfiguration. In [47] this unit is called Combined-Controller (CC)—please also refer to [39, 40]. In case of the FLP platform this Combined-Controller is physically implemented in the PCDU.

10.2.1 Component Functions During Failure Handling

As already published in [47] four types of errors need to be covered which are explained in the next paragraphs. In any case of reconfiguration of the Data Handling subsystem or of the power subsystem, essential control functions of the satellite are at least blocked or non-operational for a limited time which will degrade the spacecraft attitude and power control. So besides pure failure management in form of switch-over to a redundant unit, the FDIR system must assure shutdown of non essential power loads (payloads, thermal consumers) and must assure to bring the satellite back into a Safe Mode configuration.

Failure Type 1: Failures Identifiable by the Running OBSW
Examples for this type of hardware failures are e.g. a bus controller which is sporadically not responding and requiring retries by the OBSW or the failure case where the OBSW receives Mass Memory EDAC bit error messages. In such case the OBSW can send so-called High Priority Commands (HPC) to the active Reconfiguration Unit for DH/Ops and Power FDIR which then triggers the reconfiguration to the redundant OBC bus controller—or in this case the entire I/O-Board. In other error cases the Reconfiguration Unit may be forced to switch over to the redundant Processor-Board. In the latter case the OBSW has to be rebooted on the target Processor-Board.

To sum it up the triggering of the reconfiguration process is initiated in these cases by the OBSW, the diverse reconfiguration steps with redundant subunit power-up, defective-unit power-down and potential intermediate steps for data shift-over are carried out by the function implemented in the Reconfiguration Unit.

Failure Type 2: Crash of the OBSW and Auto-Reconfiguration
Failure type 2 covers all those errors which coincide with a crash of the OBSW. In such case a Reconfiguration Unit outside the Processor-Board has first to detect the OBSW failure and then has to perform the reconfiguration.

In the simple case of an OBSW crash due to a bug or an IC Single-Event Upset (SEU), the OBSW has to be restarted by the Reconfiguration Unit. The failure detection usually is performed by means of the processor cyclically sending watchdog signals to the Reconfiguration Unit and the latter starting FDIR action in case of the watchdog signal exceeding a timeout.

The reconfiguration activities reach from simple processor soft reset to complete Processor-Board or/and I/O-Board redundancy switchover, depending on the symptoms detected by the Reconfiguration Unit. The number and type of

watchdogs and timers obviously has to be designed appropriately for the Reconfiguration unit being able to determine between different error cases.

Such an automated reconfiguration with broken OBSW is only possible for a limited type of failure cases. The determination of the complex ones and moreover the root cause analysis requires intervention from ground. Which leads over to the next failure class described below.

As summary it can be stated that the Processor-Board shall feature watchdog lines to the Reconfiguration Unit for detection of OBSW crashes or I/O-Board signal routing failures which then can be used for OBSW restart or hardware reconfiguration—the latter only in very limited cases.

Failure Type 3: Crash of OBSW and Reconfiguration from Ground
SW crashes and auto-reconfiguration or OBSW reboot initiation are not sufficient to overcome the problem, the reconfiguration has to be initiated and to be performed from ground. This usually starts with the initiation of power shutdowns for unnecessary loads and continues with deactivation of suspected defective units and activation of hopefully still healthy ones.

All ground commands for these switch-overs have to bypass the OBSW—as it is non-operational. Therefore a special class of HPCs are used (so-called class 1 HPCs or HPC1 commands) which can be identified by the OBC's TC decoder modules and which are not routed to the OBSW on the Processor-Board like normal commands. In a standard OBC architecture these HPCs are directly forwarded to a specific subunit in the Reconfiguration Unit—the so-called Command Pulse Decoding Unit (CPDU). For more details please refer to [140]. The CPDU can decode such HPCs and can send pulse commands to the Power Control and Distribution Unit (PCDU) to trigger LCL relay on/off switching, according to what the targeted unit's status shall be. In the CDPI design, this HPC routing is slightly different—see Sect. 10.2.2.

This allows ground to activate deactivate on-board equipment completely independent from any OBSW to overcome severe failure situations.

Failure Type 4: Power-Bus Undervoltage Failures
The final main type of failures are those leading to a power shortage on board. In such case—independent of whatever the root cause was—the S/C equipment is shut off in several steps. The first units being disabled are payloads and payload data mass memory units. In further sequence then—if the OBSW apparently is unable to manage the problem—the platform units including OBC are shut down. An in case even these measures do not overcome the problem the PCDU finally deactivates itself. For these cases a PCDU is equipped with an auto-activation function for its controller as soon as the S/C power-bus again supplies sufficient voltage—e.g. due to the satellite returning from orbit eclipse phase to sun phase. With increasing power availability the PCDU subsequently activates further S/C platform units, first unit being the OBC, respectively in Safe Mode the redundant OBC (depending on the latest settings in the Reconfiguration Unit). The latter then activates platform AOCS units to achieve a stable SC safe-mode attitude acquisition and potential ground contact through activation of the transceivers.

As summary in these cases reboot and configuration are initiated by the PCDU controller and again PCDU controller and Reconfiguration Unit together manage the recovery.

10.2.2 PCDU as Reconfiguration Unit

As explained in Sect. 4.2.2 and as published in [47, 150], the Combined-Controller in this Combined Data and Power Management Infrastructure (CDPI), also serves as Reconfiguration Unit for the OBC. In this context the PCDU controller is taking over the role of the CDPI Combined-Controller. There is always one operational instance of the OBC Processor-Boards and I/O-Boards required in order to facilitate a working command chain on board the satellite (cold redundancy). Since both CCSDS-Boards are permanently powered, listening to both receivers, the hot-redundant board is instantly available in case of a malfunction of this board type.

However, it is essential to guarantee the detection and reconfiguration of any malfunctioning OBC boards. Industrial satellites usually feature an independent OBC internal Reconfiguration Unit that permanently monitors the operation of the OBC components. As explained in Sect. 4.2 the key idea of the Combined Data and Power Infrastructure is to use the processor of an intelligent PCDU to take over the task of the Combined-Controller type of Reconfiguration Unit and thus to save the development, qualification and manufacturing of one entire OBC ASIC.

1. In failure case 1 the running OBSW triggers the reconfiguration functions implemented in the Combined-Controller (CC), via the CPU-CC links, for example a reconfiguration of an OBC decoder/encoder unit.
2. In failure case 2 the auto-reconfiguration is unchanged. It simply is performed through the Combined-Controller, for example induced through watchdog signal timeouts etc. Such watchdog or alert signal lines from the CPU now have to be routed to the Combined-Controller.
3. In failure case 3 where reconfiguration has to be performed via High Priority Commands from ground, the according reconfiguration command routing goes from the decoders to the CC for reconfigurations command execution in the CC.
4. In failure case 4 (power failures, power-bus undervoltage) the approach of satellite power down and eventual reboot is controlled as in the classic architecture. Just the functions implemented formerly inside the PCDU controller now are located in the CC.

The first essential step in OBC FDIR is failure detection: For the FLP concept, the OBC/OBSW requests important housekeeping data from the PCDU at a regular interval of 10 Hz by a Common Command. This data is recorded by the PCDU within its control loop which is adjusted to match the 10 Hz polling cycle of the OBC. All requested data is submitted to the OBC/OBSW as requested. Among the periodically accumulated data are:

- Status of batteries: voltage/current/battery depth of discharge or state of charge calculated by PCDU
- Status of fuses: on/off
- Status of switches: on/off
- Status of panels: voltage/current
- Temperature sensor data
- Sun sensor data

In case where the PCDU is not polled cyclically as specified, a failure of the OBC system is assumed by the Combined-Controller. Figure 10.5 shows the four OBC boards that are involved in the command chain of Common Commands from OBC to the PCDU. These are the nominal and the redundant OBC Processor-Board as well as the nominal and redundant I/O-Board. Figure 10.5 also depicts the connections between the OBC's CCSDS-Boards and the PCDU for High Priority Commanding, which is explained in the following section in detail.

The affiliation '0' and '1' indicate that the respective units are operated in a hot redundant mode. In contrast, the boards which are operated in a cold redundant mode are affiliated Nominal (N) and Redundant (R).

The HK TM requests are sent from the OBSW on the Processor-Board via the interface on the I/O-Board, as shown in the electrical diagram in Fig. 1.6 and these

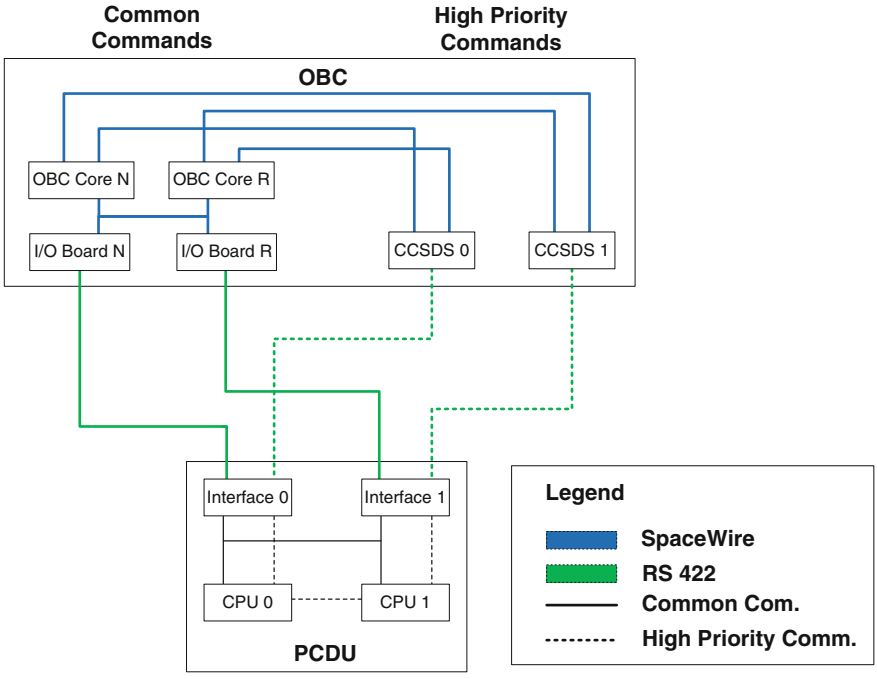


Fig. 10.5 Interface Communication between the OBC and the PCDU. © IRS, University of Stuttgart

TM polling signals are used as watchdog in the FLP Generation 1 CDPI for OBC health detection by the PCDU in its role as reconfiguration unit.

It is detected when TM requests from the OBC are received by the PCDU and a timer is reset inside the PCDU control firmware. If HK TM is not requested for more than 10 s (this value is a default setting and can be adapted in PCDU firmware), a malfunction in the OBC is assumed by the reconfiguration controller in the PCDU.

However, in such case in the CDPI design of FLP Generation 1 it can not be detected directly which of these command chain units—Processor-Board, I/O-Board or OBSW—is the source of the malfunction. Therefore, a particular reconfiguration sequence is defined and implemented in the PCDU (please also refer to Sect. 10.2.12 concerning the improved design for FLP Generation 2).

Therefore, a specific reconfiguration procedure for the OBC is performed to restore the command chain operability by switching through the available instances. After each switch step a delay time is foreseen—e.g. to allow the redundant Processor-Board to boot up entirely—and it is verified by the PCDU whether telemetry polling is resumed again by the OBC. This default hold time can be adapted by command. If polling is resumed, the reconfiguration is considered successful and the further sequence is aborted. If no polling is occurring yet, the next switchover step is performed:

- Turn the switches off for both I/O-Boards and on for the nominal I/O-Board
- Turn the switches off for both OBC Processor-Boards and on for the nominal OBC Processor-Board
- Turn the switch off for the nominal I/O-Board and activate the switch for the redundant I/O-Board
- Turn the switch off for the nominal OBC Processor-Board and activate the switch for the redundant OBC Processor-Board
- Turn the switch off for the redundant I/O-Board and activate the switch for the nominal I/O-Board

For the details and problematic cases of the rconfiguration see Sect. 10.2.3. As soon as a working command chain configuration is found, the PCDU is autonomously set into MiniOps Mode, which corresponds to the Safe Mode at system level. By implementing the autonomous reconfiguration procedure the down time of the satellite system is minimized.

The implementation of the above described concept for the OBC reconfiguration is only reasonable, since the PCDU itself is equipped with an internal watchdog circuit which facilitates the autonomous switching between the PCDU internal, redundant controllers. Thus, the monitoring and switching tasks can be performed without significant delays and thus safeguard a minimum downtime of the satellite system.

Figure 10.6 shows the watchdog functionality for the autonomous switching of the PCDU internal controllers. Both, the nominal and the redundant controller, are operated in a hot-redundant concept with a master and a slave unit at separate electric circuits. The master unit performs all activities, whereas the slave monitors the master.

The master CPU is sending a confirmation signal during its processing cycle in order to permanently confirm its operability. If this condition is no longer met, the switch logic commands to switch the master functionality to the slave unit.

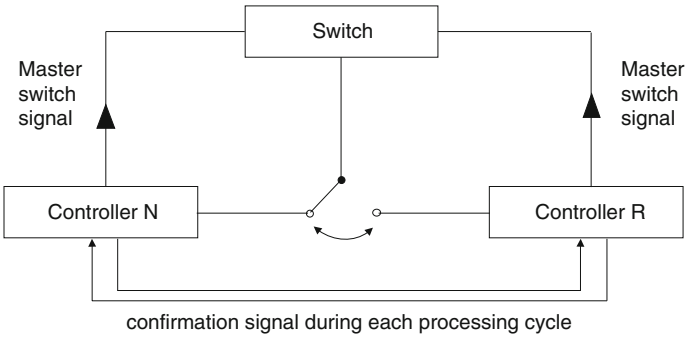


Fig. 10.6 Functional design of the switch logic for the PCDU internal CPUs. © IRS, University of Stuttgart

10.2.3 Reconfiguration Sequence

When communication between PCDU and OBC is interfered, there are several reasons that might have caused this as shown in Fig. 10.7.

(a) The connection between Processor-Board and I/O-Board is interrupted due to cable or connector damage
(b) The I/O-Board has a "software" failure or is significantly damaged
(c) The OBSW has a significant software error and cannot re-establish a stable state

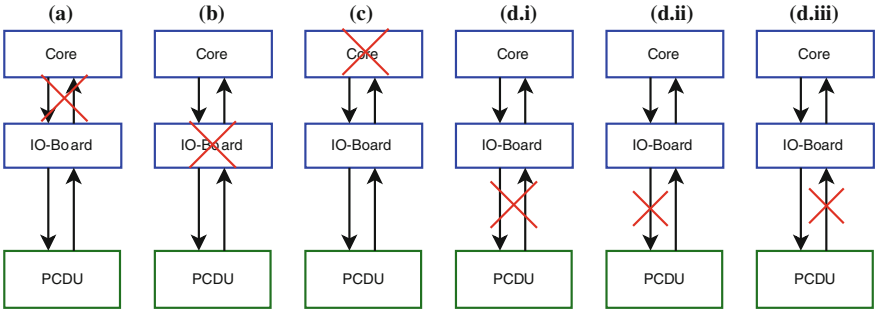


Fig. 10.7 OBC reconfiguration—Different failure cases with OBC and PCDU. © IRS, University of Stuttgart

(d) There is a cable damage between I/O-Board and PCDU:

(i) such that the complete cable is broken (neither back nor forth communication).
(ii) such that the commands do not reach the PCDU.
(iii) such that the commands from the OBC are transmitted to PCDU but the response cannot be received by OBC.

Cases (a) and (b) show the same symptoms in the OBSW and are covered in Sect. 10.2.9. The redundant I/O-Board cannot be activated by software command. Both cases will trigger reconfiguration through PCDU until communication between OBC and PCDU is re-established, either by quick recovery after the reboot of the primary I/O-Board in reconfiguration step 1 or through powering of the secondary I/O-Board in reconfiguration step 3—see reconfiguration sequence in Sect. 10.2.2.

Case (c) will activate the internal reset through the software watchdog if only part of the software is stuck and the watchdog is still running and can perform the SW boot. Or the Processor-Board or SW is stuck such that it cannot recover by itself and triggers the reconfiguration through the PCDU. The software will attempt to store occurring software errors in a Fatal Error Report (see Sect. 3.5) as well as Events in the PROM—depending on which parts of the SW still work.

Case (d) is more complicated. All three sub-cases of (d) will look the same from OBSW perspective. The PCDU device handler in the OBSW will issue HK TM requests to the PCDU but will not receive a response and will generate a MISSED_REPLY Event. This means that the commands have been sent to the I/O-Board but an according response could not be retrieved from the I/O-Board.

There is a significant difference between cases d(i) and d(ii) and case d(iii). In the first two there is nothing that can be done in OBSW except to prepare for the reconfiguration by the PCDU. However, in case d(iii) the PCDU still receives commands from the OBC. That means that a reconfiguration will not be triggered by the PCDU although the communication is not working properly. In this case an FDIR function in the OBSW needs to intervene.

A common approach for not getting responses from the PCDU, but otherwise working properly—case d(iii)—is attempted and provided in the sequence diagram further down in Fig. 10.11.

Before going through the steps of Fig. 10.11 some explanations are provided. The software objects that are included in this procedure are the *FDIR*, the *IO-Assembly* (displayed in Fig. 10.8) handling the I/O-Boards and switches between them, the PCDU device handler as executing object for commands to the PCDU, and the Polling Sequence Table (PST). The PST is important since a running PST attempts active device communication which leads to failure Events if a DH cannot communicate with its device.

What happens when the communication to the PCDU is interrupted—case (d)—is that the PCDU device handler will attempt to send the periodic HK TM request command to the PCDU and will not receive a proper response. As a result the PCDU device handler issues the Event MISSED_REPLY. In the *FDIR*, an Event

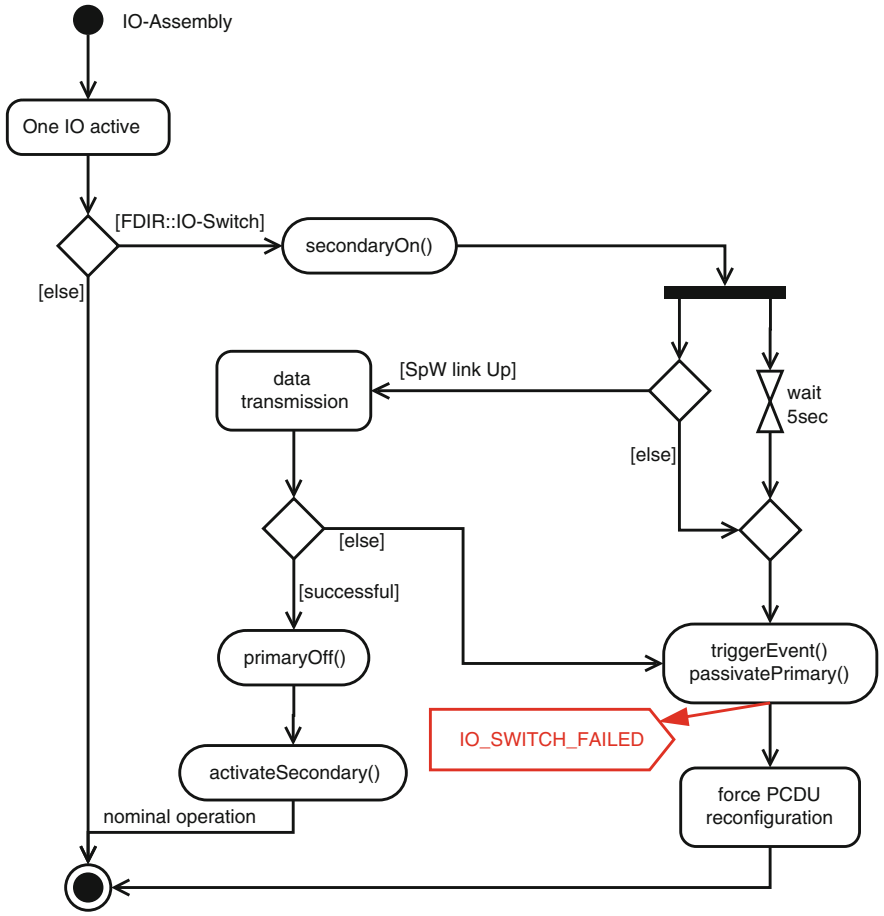


Fig. 10.8 IO-Assembly mode diagram. © IRS, University of Stuttgart

counter is implemented to store the number of times the response from the PCDU has been missed. This is depicted in Fig. 10.9.

This is not the complete algorithm. The implementation includes a mechanism to reset the failure detection for FDIR cases in which the communication was only interrupted shortly and recovers by itself after a short period. The detailed mechanism of increasing and decreasing an FDIR failure counter is provided in Sect. 10.4.2 and Fig. 10.26 and is applied to the PCDU device handler failures accordingly.

When the counter in Fig. 10.9 exceeds the limit, FDIR will command the *IO-Assembly* to attempt an I/O-Board switch—see Fig. 10.9 and later Fig. 10.11. The process for the I/O-Board switch itself is depicted in Fig. 10.10.

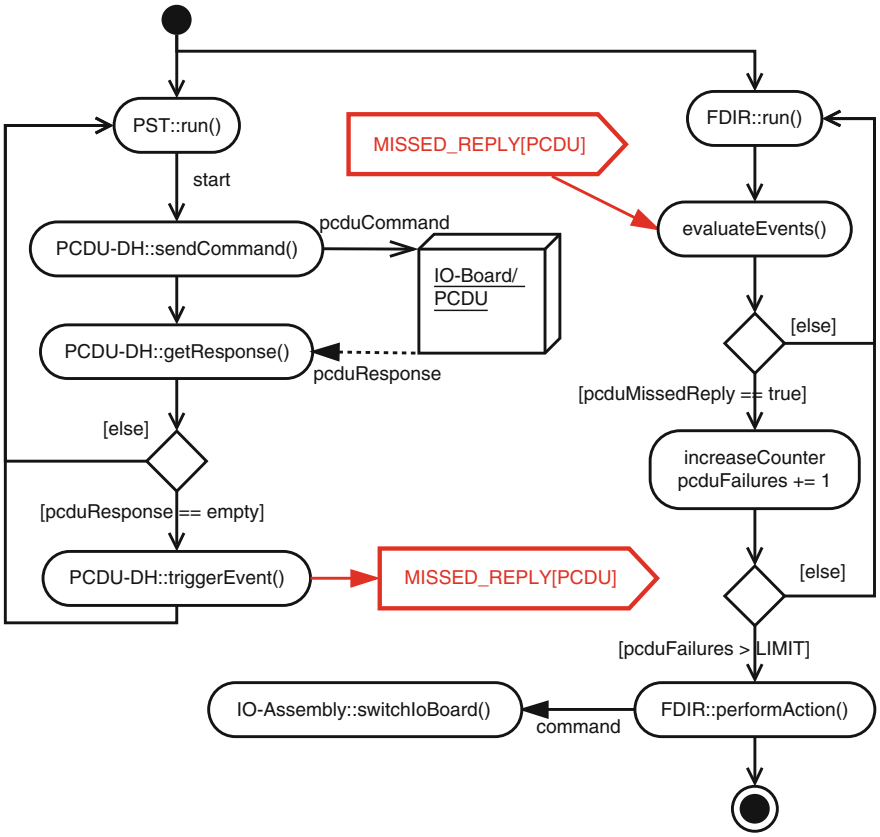


Fig. 10.9 PCDU failure count for reconfiguration. © IRS, University of Stuttgart

- At first the *IO-Assembly* will power the secondary I/O-Board by commanding the PCDU and wait for the SpW link to come up.
- Upon confirmation, the primary I/O-Board is passivated which means the interfaces are deactivated. Due to Y-connections in the wires of the system harness, the transmission interfaces of both I/O-Boards must not be active at the same time. Otherwise signals from the two interfaces could interfere with each other.
- Consequentially, the interfaces of the secondary I/O-Board can be activated and the stored TM from the primary I/O-Board can be transmitted to the secondary I/O-Board.
- At the end of the process, the primary I/O-Board is powered down by commanding the PCDU accordingly.
- The I/O-Board assembly automatically updates the IO status in the *datapool* so that the communication between OBC and devices works with the healthy I/O-Board.

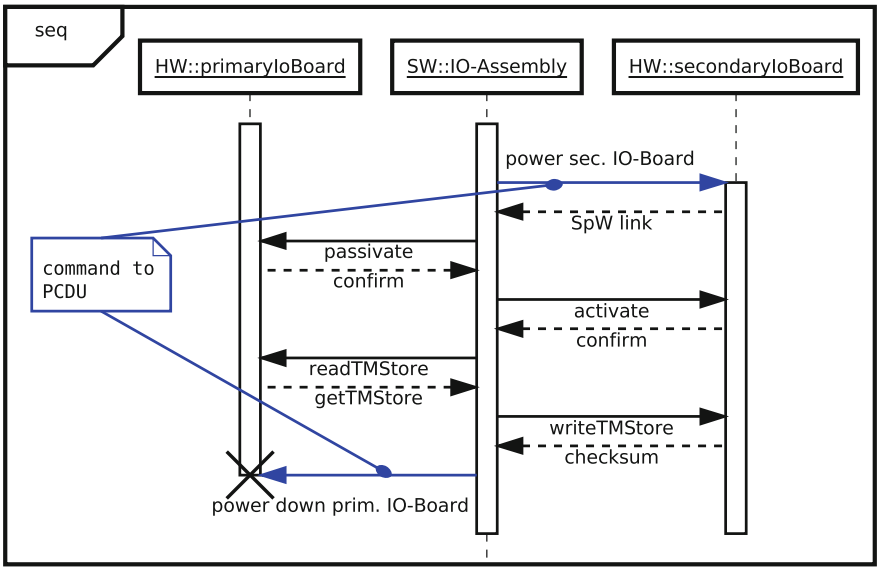


Fig. 10.10 OBC reconfiguration: IO-Assembly switch and configure secondary IO-Board. © IRS, University of Stuttgart

- In Fig. 10.11 it is shown that as soon as the secondary I/O-Board is powered, the PST will be stopped to avoid generating failure events that are expected by the system during this switch.

Reminder
The previous three cases from (d) are considered with (i) no communication between PCDU and OBC, (ii) PCDU does not receive any commands and (iii) PCDU response is not received in OBC. The PCDU will only trigger the reconfiguration, if commands are no longer received, which is not the case in d(iii).

The flow depicted in Figs. 10.11 and 10.13 shall be explained here step by step:

- In the PST (details in Sect. 3.4.3) the PCDU DH is invoked.
- PCDU DH cannot retrieve TM from the PCDU and issues the event MISSED_REPLY.
- After 10 retries, *FDIR* decides to attempt an I/O-Board switch and commands *IO-Assembly*. Simultaneously, *FDIR* generates Event SWITCH_IO. Further MISSED_REPLY signals from PCDU DH are stored in Event TM but will be ignored by the *FDIR* object.
- *IO-Assembly* executes the I/O-Board switch as shown in Fig. 10.10.
- Alternative Branch IF:
  - Secondary I/O-Board can be detected and the PST is paused to avoid generation of DH failures during I/O-Board switch.
  - Secondary I/O-Board is configured as shown in Fig. 10.10.

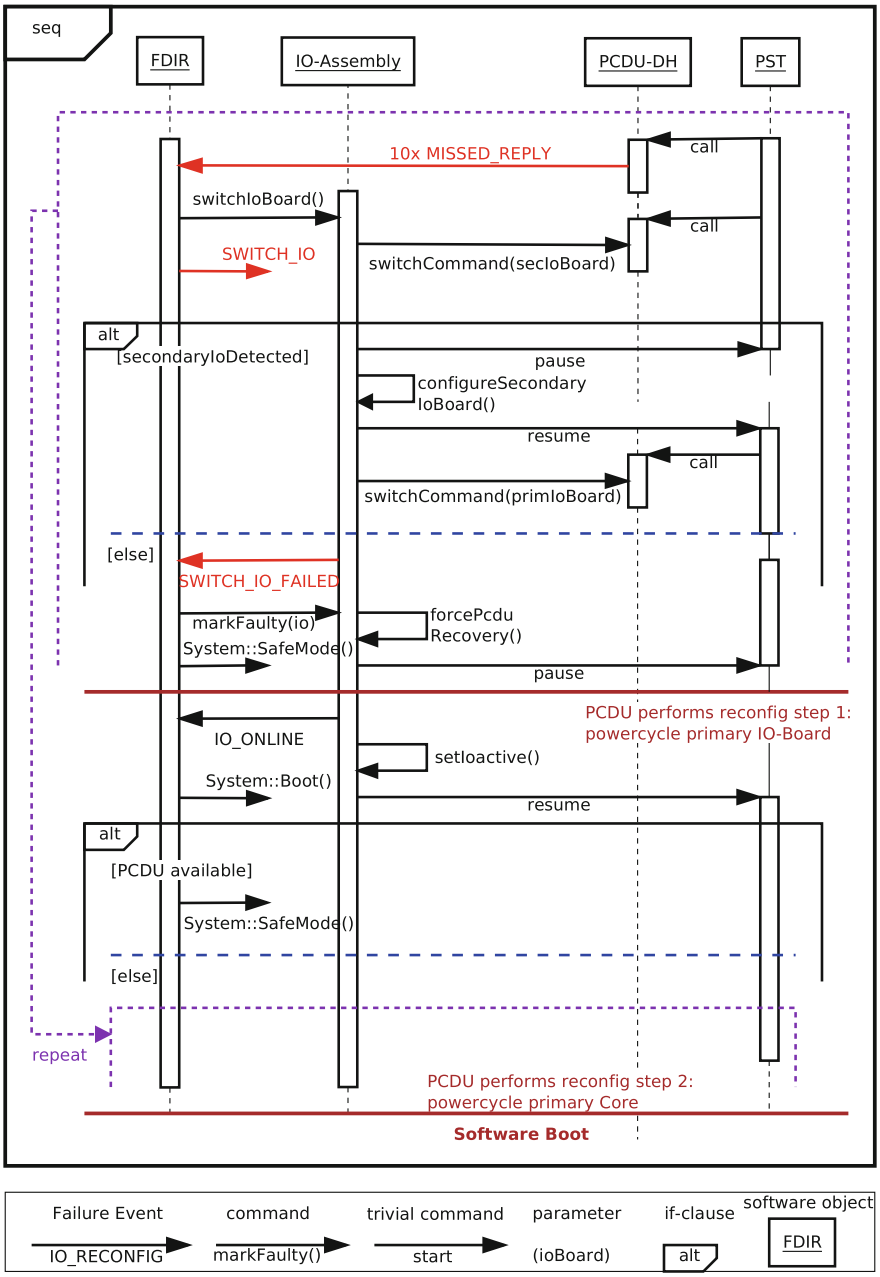


Fig. 10.11 OBC reconfiguration through PCDU—Software perspective—Sequence diagram (1). © IRS, University of Stuttgart

  - PST is resumed, PCDU DH is invoked and former primary I/O-Board is switched off.
  - Nominal operation can continue, ground interaction can command S/C transition from Safe Mode to Idle Mode.

- Alternative Branch ELSE:
  - After expiration of the I/O-Board switch attempt as illustrated in Fig. 10.8, the *IO-Assembly* issues the event SWITCH_IO_FAILED.
  - FDIR marks the primary I/O-Board as faulty and interrupts the communication with the devices including PCDU (compare Fig. 10.9). This intentionally provokes that the PCDU initiates reconfiguration of the OBC.
  - At the same time the PST is paused to avoid generation of device handler failures during I/O-Board reconfiguration through PCDU.
  - *FDIR* commands *System* into Safe Mode and switches off Idle Mode and payload devices to prepare for the state after the reconfiguration. (Reminder: PCDU will be enter MiniOps Mode when communication is re-established. That means only Safe Mode devices are powered).

- PCDU performs reconfiguration step 1 and restarts the primary I/O-Board.
- Upon detection of the SpW link, *IO-Assembly* sends the Event IO_ONLINE and activates the primary I/O-Board.
- *FDIR* commands *System* to boot properly and to resume the PST to attempt proper communication.
- Alternative Branch IF:
  - If the communication to the PCDU has recovered, *System* will operate in Safe Mode.

- Alternative Branch ELSE (repeating sequence from step 1):
  - As before, PCDU DH sends Event MISSED_REPLY.
  - …
  - FDIR marks primary I/O-Board faulty to pause PST and prepare for PCDU reconfiguration step 2—restarting the nominal Processor-Board and software reboot.

- Through reboot of the Processor-Board the software is restarted (Fig. 10.12).
- After software boot, the entire process is repeated and the PCDU device handler signals MISSED_REPLY from the PCDU.
- Eventually *FDIR* interrupts all device communication to prepare for the third step of the PCDU reconfiguration.
- Upon powering the redundant I/O-Board, nominal operation is re-established and *System* can be commanded to transition back to Idle Mode.

The intention of this approach and the early I/O-Board switch-attempt is to avoid falling into a loop trap where PCDU performs the first reconfiguration step over and

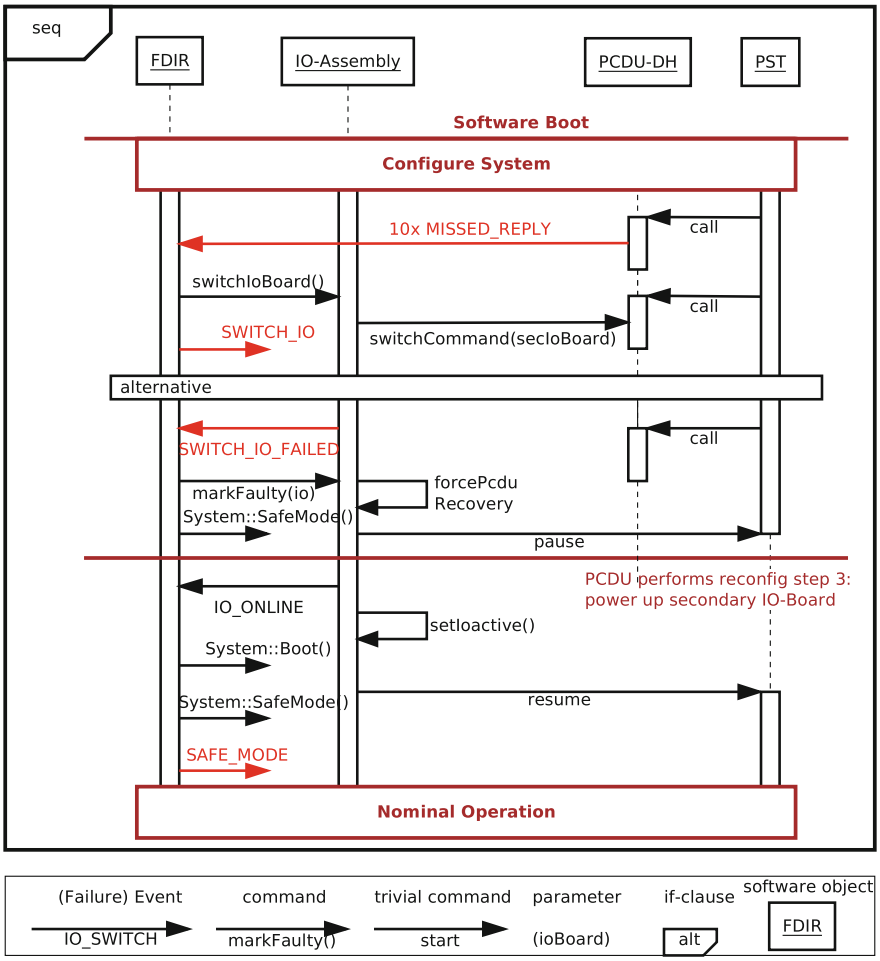


Fig. 10.12 OBC reconfiguration through PCDU—Software perspective—Sequence diagram (2). © IRS, University of Stuttgart

over. In case d(iii) this could happen as described in the following and as is depicted in Fig. 10.13.

(1) PCDU and *System* start in Idle Mode.
(2) *System* cannot detect a response from the PCDU and interrupts the communication to the PCDU
(3) PCDU performs reconfiguration step 1 and restarts the primary I/O-Board.
(4) Upon detection of established SpW link, System initiates Boot and attempts communication with PCDU.
(5) PCDU receives an HK request, enters MiniOps Mode and interrupts the reconfiguration sequence.

(6) =(2) *System* cannot detect a PCDU response and interrupts communication again.

If in the OBSW *FDIR*/*System* interrupts the communication to the PCDU, the reconfiguration will be performed. But after restarting the I/O-Board, the OBC will command the PCDU again to get the system configured, despite not receiving a response. This would interrupt the current reconfiguration sequence until FDIR interrupts the communication again which leads to an endless loop trap. The forced I/O-Board switch, after recognizing the interrupted communication, is introduced to prevent this scenario from occurring.

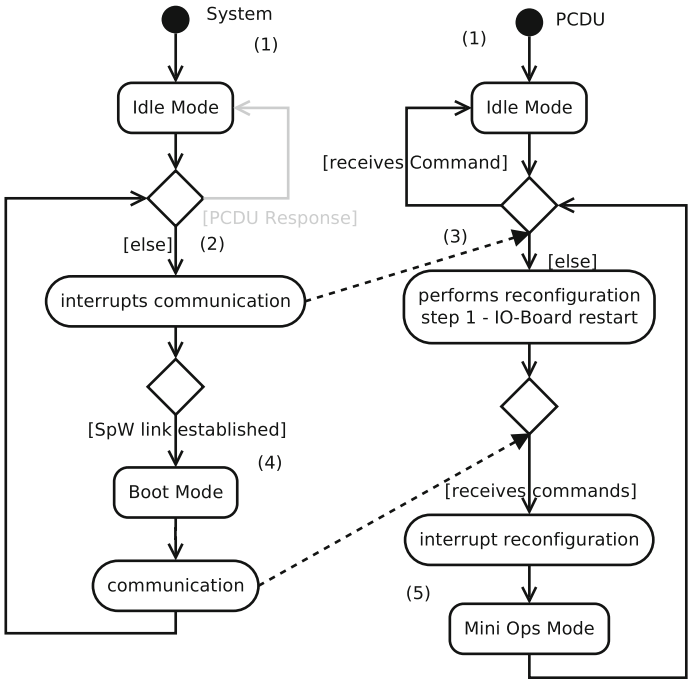


Fig. 10.13 PCDU reconfiguration—loop trap. © IRS, University of Stuttgart

10.2.4 HPC-Based Spacecraft Reconfiguration

The PCDU by means of its Combined-Controller (CC) incorporates the functionality, which is essential for FDIR operations at S/C system level. In order to establish a reliable concept for spacecraft operations and system commandability High Priority Commands (HPCs) are to be applied in case the spacecraft OBSW has crashed and an automatic reconfiguration failed. The concept of HPCs is explained

in [11]. Such commands are applied in case the nominal command chain units, such as OBC Processor-Boards and I/O-Boards, or the OBC's OBSW is not operational. In standard architectures the HPCs are submitted from ground to a dedicated Command Pulse Decoding Unit (CPDU) of the OBC which then commands the PCDU relays switching via dedicated pulse command lines. In case of the integrated CDPI architecture HPCs also bypass the OBSW command chain but do not need a CPDU as they are transmitted directly from the CCSDS-Boards to the PCDU (see Sect. 4.2.2 and [47]).

The following explanations of the CCSDS protocol are limited to the parts that are necessary to understand the HPC transmission and forwarding. Please consult the CCSDS standards [25] for further information. Figure 2.6 in Sect. 2.2 shows an example of the composition of an uplinked TC packet that is transmitted from ground to the satellite. After the decoding of the so-called Command Link Transmission Unit (CLTU), the TC Transfer Frame contains the commands from ground. Only the Frame Header and the TC Segment must be considered to understand the HPC forwarding. The Virtual Channel (VC), in the Frame Header indicates which command chain unit receives the data. The following 4 VCs are available for FLP target spacecraft:

- VC1: nominal command to OBC Processor-Board N
- VC2: HPC1 to CCSDS-Board N
- VC3: nominal command to OBC Processor-Board R
- VC4: HPC1 to CCSDS-Board R

Whereas nominal commands are assigned to the VCs '1' and '3', HPCs are allocated to the VCs '2' and '4'. An HPC that is commanded by ground is referred to as High Priority Command Level1 (HPC1). HPCs Level 2 are processed by the OBC S/W. The TC Segment contains the Multiplexer Access Point Identifier (MAP-ID). A MAP-ID that equals '0' states that the TC Segment contains HPC1s and that containing commands are directly forwarded from the CCSDS-Board to the PCDU. All MAP-IDs unequal to '0' imply PUS packets and are transmitted to the OBC Processor-Board for further processing by the OBC S/W.

As described in Sect. 4.2.2 industrial satellites may feature a so-called Command Pulse Decoding Unit (CPDU) on board to receive HPCs. This unit routes the commands to the respective units. An HPC consists of 2 bytes. The first 8 bits contain information on the channel selection, the second 8 bits on pulse length definition. So it is possible to command 256 units (channel selection) through 256 commands (pulse length definition) by utilizing HPCs.

For FLP, the CPDU is integrated in the PCDU as the only unit commanded by HPCs. Thus, 65536 different commands may be implemented to reconfigure the satellite system by switching LCLs and component switches. The PCDU features a nominal and a redundant RS422 communication interface for the reception of HPCs (see Fig. 10.5) with a baud rate of 115200. All HPC packets are implemented with a 2 byte header that serves as an identifier for the following HPC frame. The composition of the HPC header is as follows: (Table 10.1)

Table 10.1 Header composition of an HPC Frame

Byte Number	1	2
Bit Composition	11111111	01010101

An HPC frame can contain up to 4 high priority commands. Every command starts with a TC Source Packet Header, (TSPH) which is completely dismissed. Each of the HPCs consists of the 6 bytes TSPH, 2 bytes command plus 2 bytes checksum. HPCs can be used to activate or deactivate a single or a specific set of on-board components which cover a specific safety aspect. By virtue of their importance, HPCs are processed immediately after reception at the PCDU and in favor to Common Commands. The most important HPCs are:

- Activate or deactivate the on-board heaters
- Deactivate all non-essential loads for the Safe Mode to save energy
- Reconfigure the command chain

The structure of a single HPC is as follows (Table 10.2):

Table 10.2 Basic HPC structure

	HPC structure				
Byte No.	0..5	6	7	8	9
Meaning	TSPH	CMD ID-H	CMD ID-L	CRCH	CRCL

with

Byte	**Explanation**
Header	0xFF55
TSPH	TC Source Packet Header (0x00)
CMD ID-H	Activate: 0x0F; deactivate: 0xF0
CMD ID-L	HPC No.
CRCH	16Bit CRC (Byte 0–7), higher Byte
CRCL	16Bit CRC (Byte 0–7), lower Byte

Table 10.3 shows an example of an HPC command sequence which may contain up to 4 single HPCs. Table 10.4 provides an overview of all implemented HPCs for the FLP Generation 1.

Table 10.3 HPC frame composition

	HPC sequence structure (up to 4 commands)				
Byte No.	0..1	2..11	12..21	22..31	32..41
Meaning	Header	HPC1	HPC2	HPC3	HPC4

Table 10.4 HPC commands

HPC Number	Action
1–77	Activate switch 0–76 (Turn single components on)
78–144	Deactivate switch 0–76 (Turn single components off)
145	Activate all heater switches
146	Activate nominal core component switches (OBC N, I/O board N)
147	Activate redundant core component switches (OBC R, I/O board R)
148	Activate first core component cross-coupling switches (OBC N, I/O board R)
149	Activate second core component cross-coupling switches (OBC R, I/O board N)
150	Deactivate nominal core component switches (OBC N, I/O board N)
151	Deactivate redundant core component switches (OBC R, I/O board R)
152	Deactivate first core component cross-coupling switches (OBC N, I/O board R)
153	Deactivate second core component cross-coupling switches (OBC R, I/O board N)
154	Deactivate all payload switches
155	Deactivate all switches except for 'Safe Mode' components and 'Survival heaters'
156	Deactivate all heater switches

10.2.5 Software Watchdog

As mentioned in Sect. 10.2.2, there is a reconfiguration function implemented in the PCDU that will take action if the HK data from the PCDU is not cyclically requested by the OBC for 10 s. This might be the case if the OBSW crashed for some reason. The Processor-Board will be power-cycled in such case. However, this is a very harsh way of treating the OBC. In particular since the PCDU reconfiguration begins with an I/O-Board reboot, which takes some additional time.

A preferred and faster way for this is to automatically recognize a software crash and to initiate a controlled reboot. In that case some data might still be written to the I/O-Board to allow analysis of the software crash by ground FDIR. For that reason, a software watchdog (WD) function is implemented inside the software. This function needs to be triggered in every PST cycle to reset a counter.

If the counter is not reset for several seconds and the counter reaches a limit, a reboot is initiated by the *watchdog* software object. If one of the main SW tasks returns a wrong *returnCode* during task processing (see also Fig. 3.10) to a controlling function for whatever reason, the *System* will not reset the WD counter. At the same time, an Event SW_RETURNCODE is generated with the *returnCode* value as parameter for allowing to recapture the process. SW_RETURNCODE Events are generated and stored whenever an unexpected value is returned from a software function. When the WD counter exceeds a configurable limit, it will

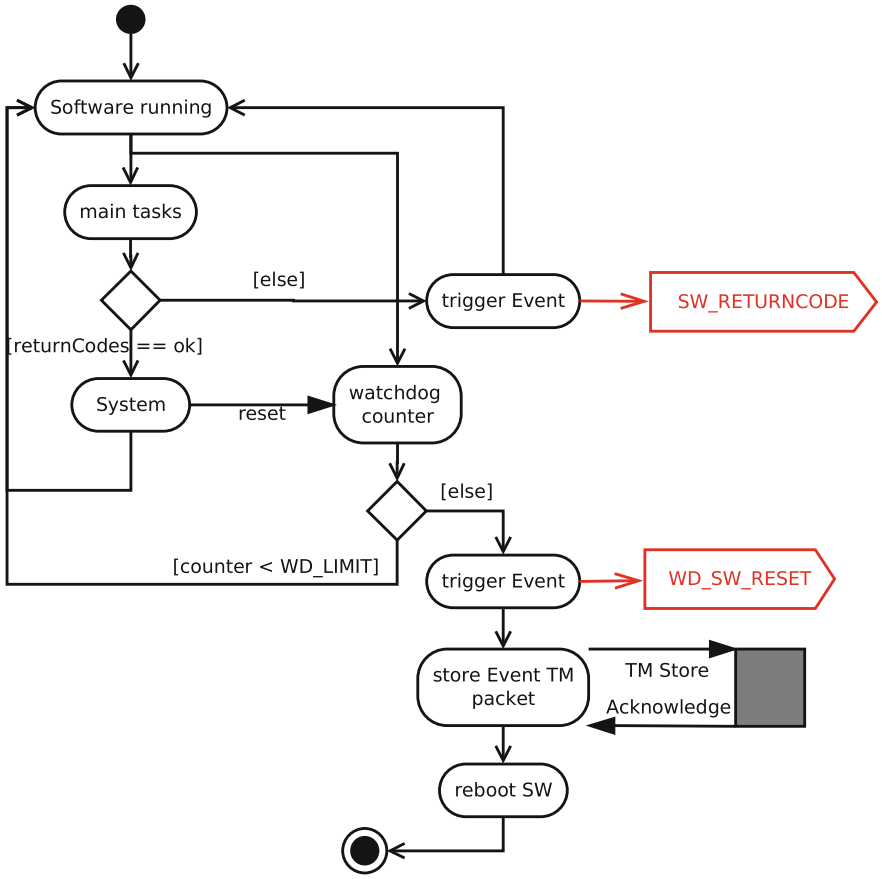


Fig. 10.14 Software reboot through SW watchdog reset. © IRS, University of Stuttgart

initiate a software reboot and a corresponding Event SW_WD_RESET. After the acknowledged storage of that event in the I/O-Board TM store, the reboot will be performed.

If the reboot of the software does not recover the complete software and OBC functionality, the Reconfiguration Unit in the PCDU will trigger a reconfiguration of the OBC. The same is true in case where the software watchdog function shows a malfunction or the controlled software reboot is prevented for some reason. This process is shown in Fig. 10.14.

10.2.6 Software (Re-) Boot

If a software reboot occurs due to a software watchdog reset (Sect. 10.2.5) or a reconfiguration (Sect. 10.2.2), previously stored temporary data is lost. In that case, the software needs to be made aware of the state of the spacecraft. There are several measures to make sure the software is able to identify that and to continue operations.

The Processor-Boards are exactly the same hardware, with the same software image stored in the PROM memory area. There is only one difference to distinguish between the two. A part on the PROM memory (64 Bytes) of the Processor-Boards contains a customized boot loader (see also Sect. 4.3.1). The boot loader stores an identifier to tell which Processor-Board the software is running on. So while booting the software, it can read from that boot loader whether it is running on the nominal or the redundant Processor-Board. It can be assumed that if the software has been rebooted, a failure happened in the software itself, or with the Processor-Board. So all other OBC boards, and one I/O-Board in particular, are supposed to be running properly. The very first boot shall be covered with this approach as well. Following this, it is not required for the OBSW to recognize whether it is the initial boot after launch or a bot after watchdog reset or PCDU reconfiguration. However, the Processor-Board PROM includes a boot counter (see Sect. 4.3.1).

Secondly, there is a defined Spacecraft State Vector (SV) (see Sect. 10.2.11) stored in the I/O-Board that contains the latest configuration of the system. From that state vector the software can determine which devices are active and running and can restore nominal operation as before the reboot.

Thirdly, there is all the data concerning the satellite state, from which operation can be proceeded. There are values for latest position, attitude and especially the latest SoC for the battery, which are important regarding the prevention of major failures or damage.

After a software boot or reboot, the system always attempts a transition into a stable Safe Mode. Failures that lead to a software reboot or reconfiguration are deemed severity level *HIGH*.

The software boot process is displayed in Fig. 10.15. The sequence is as given in the following description:

- Before any actual boot of the main function happens, a CRC check is performed. This will be especially important when a second image and software patching in the PROM comes into play.
- The software starts with creating all required objects and interfaces and assigning them to their respective tasks.
- For visibility on ground and to allow recognition of an autonomous Processor-Board switch, the Processor-Board ID can be read from the PROM and stored in the *datapool* object.
- The *tasks* are started to allow for a certain functionality. However, device communication is not activated yet. The system needs to be configured before.

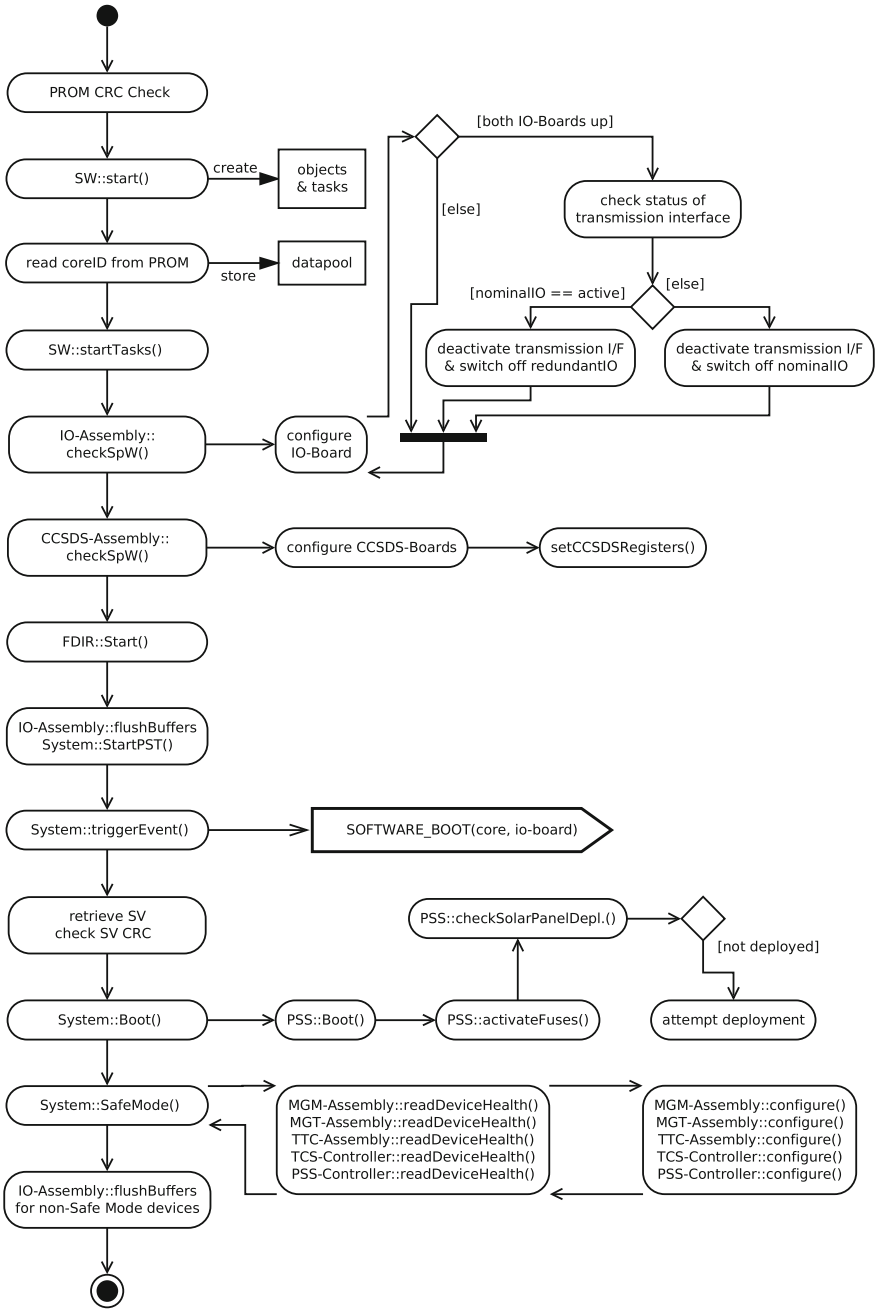


Fig. 10.15 FLP OBSW boot sequence. © IRS, University of Stuttgart

- The *IO-Assembly* recognizes whether one I/O-Board-bound SpW interface has one link (is the case at first boot, recovery or after a software crash during nominal operation) or even two (is the case if the software crashed during data transmission between the I/O-Boards).
- If both I/O-Board links are detected, the assembly checks for which I/O-Board is active. If the nominal one is active, the redundant one is passivated actively. Since the software crashed during data transmission it is not guaranteed, that the transmission was successful. Thus, it is reasonable to continue operations with the previously used—which is the primary—I/O-Board. The command to switch off the secondary I/O-Board is generated and will be transmitted to the PCDU as soon as device communication has started.
- The *CCSDS-Assembly* checks for available CCSDS-Boards and writes necessary registers to operate the board conform with the intended communication based on CCSDS and SCOS-2000.
- At this point, the FDIR processing is started. If failures had been detected before, the software would have restarted, or eventually switched to the back-up software image.
- After one I/O-Board is selected, the transmission buffer is read actively. This leads to the I/O-Board firmware to discard data that are stored from previous communication cycles. No remaining (old) data will interfere when restarting the communication with the devices.
- *System* starts the PST but the DH are not configured yet, so no communication will happen except for the PCDU HK TM requests.
- At this point, *System* creates the SOFTWARE_BOOT Event including the information of active Processor- and I/O-Board.
- Before configuring subsystems, assemblies and device handlers, the previously stored Spacecraft State Vector is retrieved from the active I/O-Board. After successful CRC check, *datapool* values are written, *validFlags* are set, *healthStatus* values of the devices are read and stored. If the CRC check fails, the initial values from SW are still sufficient to achieve a stable system state. This might take longer, however, since devices might be activated that have been switched off before.
- When the state vector (SV) is retrieved, *System* is commanded into Boot Mode (see Fig. 2.1) including booting the controller for TCS and PSS.
- The *PSS_Subsystem* initially activates all fuses to achieve a known state in the PCDU.
- *PSS_Subsystem* checks the status of the solar panel deployment. At the very first boot, the deployment mechanism is activated since at launch the SA-Deployed flag = 0 and the PCDU will report undeployed SA (reed sensor values). If the panels are already deployed (SA-Deployed flag = 1) or PCDU TM indicating successful deployment the flag is set to 1.
- *System* configures the spacecraft to transit to Safe Mode. This includes mode transitioning of all assemblies into On Mode. The assemblies read the respective *healthStatus* for their devices and command the respective *deviceHandler* in On and Nominal Mode (see also Sect. 3.2.3).

- *TCS_* and *PSS_Controller* also read the *healthStatus* of electrical and temperature sensors and heaters to update their internal health information.
- In Safe Mode, only devices MGM, MGT, TTC, OBC, PCDU are powered. But depending on the state before the boot, further devices might have been powered until to Safe Mode transition. Since for example GPS sends data periodically, another flush of non-Safe Mode devices is performed to erase data in their transmission buffer after powering them down.
- Nominal Safe Mode operation is performed until the battery strings are charged and ground interaction leads to a transition into a higher mode.

10.2.7 Data Transmission Between I/O-Boards

For back-up reasons of the data stored in the primary I/O-Board, the data has to be copied from the primary I/O-Board to the secondary I/O-Board. Stored data is housekeeping telemetry that is collected throughout the operation, Events that occurred either through nominal actions or malfunctions, and the Spacecraft State Vector as defined in Sect. 10.2.11 with S/C health state and the *datapool* variables.

To transmit data from the primary to the secondary I/O-Board, the secondary board has to be powered. The *IO-Assembly* includes a function that handles activation and deactivation, and data transmission from one board to the other automatically every 15 min regarding on-board time.

If an Idle Mode or payload device is deactivated by FDIR due to a malfunction in between two such automated I/O-Board synchronization processes, the data in the secondary I/O-Board is not updated directly.

In case the primary I/O-Board now shows a malfunction before the secondary I/O-Board is updated, the OBSW will continue operation with the secondary I/O-Board. This can only occur if a reconfiguration has been conducted by the PCDU. Otherwise the OBSW would have selected the primary I/O-Board (see Sect. 10.2.6). Terminated reconfiguration means that the S/C is in Safe Mode and the PCDU only deactivated Idle Mode and payload devices. Previously active Safe Mode devices are still active. It is possible for the operator to update the health flag of the previously faulty and deactivated device on the secondary I/O-Board. If the operator commands the S/C into Idle Mode without updating the health state, the faulty device is falsely activated. However, this is not deemed dangerous because the system should be able to identify the device as faulty again. The device will be deactivated by *FDIR* and operations can continue.

This I/O-Board synchronization process will not yet be implemented in the OBSW for the first FLP mission “Flying Laptop”. Here it has to be commanded manually by ground.

10.2.8 PCDU Automated Fault Management

If the reconfiguration of the OBC is in progress, as defined in Sect. 10.2.3, and a working configuration is not found immediately, the OBSW might be down for several minutes. During this time, the satellite cannot be properly controlled, and power generation through solar panels cannot be guaranteed. In case where the software/OBC crash happened during a full operational scenario with high power consumption, this might lead to running into an under-voltage failure case.

For such scenarios, the PCDU checks the State-of-Charge of the battery periodically and in case of the satellite bus voltage decreasing below 20 V, it enters the Shutdown Mode. In this mode all components are switched off, except for the PCDU itself—to regulate the battery balancing.

A second observation function concerning power consumption is implemented in the PCDU. One or more devices, depending on their importance, are grouped together and can be deactivated with a single fuse. For all devices a typical value of electric current for their operation is stored and if a fuse detects an over-current for a group of devices, the fuse is automatically opened and the devices are powered down.

10.2.9 IO- and CCSDS-Board SpaceWire FDIR

For the I/O-Boards and CCSDS-Boards, two different failure categories need to be considered.

The one introduced in Sect. 10.4.2 considers failures of the I/O-Board firmware, failures on RMAP level and failures of I/O-Board transmission hardware. Failures for the CCSDS-Board packet transmission via RMAP are covered as well. Particular CCSDS connection failures to the TTC equipment are covered in Sect. 10.5.

SpaceWire communication is implemented in the hardware of the OBC Processor-Board. Observation of the SpW implementation is performed by checking status registers in the SpW hardware. The SpW driver is implemented following [3] where an “Error Recovery Scheme” is defined. Failures are reported as OBSW failure events, but failure handling is already covered in the driver implementation. The Events are forwarded to ground for further analysis. A list of the Events is provided in Sect. 10.4.4. If the SpW link to the operational I/O-Board or CCSDS-Board is lost, the mechanism which was explained above will be triggered, further failure handling for SpW is not required.

10.2.10 Memory EDAC/TM Store Integrity

For the Processor Board, an Error Detection and Correction (EDAC) controller is implemented, that allows correction of a single memory error per chip and detection

of two errors. The EDAC functionality is available for the PROM and the SRAM memory areas. Further information about the EDAC mechanism can be found in [53].

The I/O-Board does not verify RMAP accesses to its memory areas. The RMAP packets stored to the transmission buffers will be verified according to [5]. There is no dedicated EDAC function implemented for the provided I/O-Board mass memory area. For the TM store and Spacecraft State Vector memory on the I/O-Board, a verification mechanism with checksums is used to prevent the software from handling corrupted data.

10.2.11 Spacecraft State and Configuration Vector

A failure case might occur that leads to rebooting the OBSW on the active Processor-Board. Such might be reconfiguration through PCDU or a self-initiated reboot by the software watchdog. In that case all temporary stored data such as position, attitude, battery SoC, etc. are lost. To prevent the system from recomputing its entire status from scratch, a certain memory region on the I/O-Board (all NV-RAM) is used as storage for the S/C state vector. After a reboot of the software, this latest state can be retrieved from the I/O-Board and nominal operation can be reached much quicker.

With a period of 300 s, the Spacecraft State Vector is stored in the primary I/O-Board to keep the data in line with the actual status of the S/C. With a lower frequency of 900 s, this S/C state vector is also transmitted to the secondary I/O-Board for the case of a faulty primary I/O-Board. The safeguard S/C state vector contains the following data:

- LEOP-Completion Flag: Flag indicating to the OBSW that the LEOP phase has been completed.
  Default Value = 0
  Value change by GndOps
  Normal Flight value = 1
  OBSW will keep TTC TX on as long as Flag = 0
  Available as System TM
- Separation-Completion Flag:
  Flag indicating for the OBSW that launcher separation is completed. Available as System TM
- SA-Deployed Status Flag:
  Flag indicating for the OBSW that solar array is properly deployed. Available as System TMDevice health statuses
- Configuration tables that are stored in the (sub-)system objects (e.g. Safe Mode components for the system, sensor redundancies for temperature sensors)
- Values of adaptable configuration variables (monitoring limits etc.)
- Configuration of CCSDS-Board
- Battery SoC
- Next ground contact times

- Remaining I/O-Board storage volume
- HK definitions and activated HK TM packets

All entries of the Spacecraft State Vector are attended with a timestamp and a *validFlag*. The complete state vector also has a timestamp which declares the time at which the state vector was written. This is required to inform a newly booted OBSW which data is the more accurate and which data can be used or needs to be ignored if both I/O-Boards are powered (see Sect. 10.2.7). Additionally, there is a checksum that can be compared when retrieving the SV to ensure that the SV has been transmitted completely.

In case of an OBSW crash, an OBSW Fatal Error Report will be attempted to be generated and stored in the PROM (see Sect. 3.5). This death report mainly consists of unexpected *returnCodes* of internal functions. These *returnCodes* will be stored in separate TM packets and serve as a kind of debug output of the software. On next ground contact the report will be downloaded. This report allows for further root cause analysis of the OBSW crash on ground and how a change of values or a OBSW patch might avoid the same crash happening again in the future.

10.2.12 SW Watchdog for FLP Generation 2

To simplify the reconfiguration scheme for the OBC boards (see Sect. 10.2.3) and to make it clearly identifiable for the PCDU Combined-Controller whether an I/O-Board or the Processor-Board is the root cause for lacking communication to the PCDU, the FLP Generation 2 computer architecture is equipped with separate watchdog lines from Processor-Boards to PCDU.

These watchdog signal lines are marked in light blue in Fig. 9.14. The watchdog counter signals are sent cyclically over RS422 serial lines from Processor-Board to PCDU. The PCDU TM polling by OBSW still goes via I/O-Board.

Thus the PCDU can uniquely identify which communication partner is non-operational and can simply reconfigure the affected unit without the complicated workflows of Figs. 10.11 and 10.12.

10.2.13 Events in the Core DHS FDIR

Event 10.1: SOFTWARE_BOOT

	Event Name	Event Severity	Event Parameter
	SOFTWARE_BOOT	INFO	Core ID, I/O-Board ID
Event description	This event will be generated at software boot		
Reaction	–		

Event 10.2: IO_ONLINE

	Event Name	Event Severity	Event Parameter
	IO_ONLINE	INFO	I/O-Board object-ID
Event description	This event will be generated if the SpW link to an I/O-Board is detected		
Reaction	–		

Event 10.3: MISSED_REPLY

	Event Name	Event Severity	Event Parameter
	MISSED_REPLY	LOW	Device object-ID
Event description	This event will be generated if a deviceHandler Object does not receive a response from its device		
Reaction	FDIR counts the events and after a certain amount of events has occurred within a certain time period, the device will be rebootet. If the event keeps occurring, FDIR will shut the device down and mark it PERMANENTLY_FAULTY. Only ground can reactivate the device		

Event 10.4: SWITCH_IO_FAILED

	Event Name	Event Severity	Event Parameter
	SWITCH_IO_FAILED	LOW	I/O-Board object-ID
Event description	FDIR has decided to switch the I/O-Board. This event will be generated if the switch to the secondary I/O-Board has failed		
Reaction	FDIR marks the primary I/O-Board as faulty. This interrupts the communication with the PCDU and triggers the reconfiguration sequence in the PCDU		

Event 10.5: SW_RETURNCODE

	Event Name	Event Severity	Event Parameter
	SW_RETURNCODE	LOW	returnCode
Event description	This event will be generated if an unexpected returnCode from a function or task is returned to the system. Event will be stored in Event TM for software debugging		
Reaction	Software watchdog will restart software		

Event 10.6: SWITCH_IO

	Event Name	Event Severity	Event Parameter
	SWITCH_IO	MEDIUM	I/O-Board object-ID
Event description	This event will be generated if FDIR decides to switch to the secondary I/O-Board		
Reaction	–		

Event 10.7: SAFE_MODE

	Event Name	Event Severity	Event Parameter
	SAFE_MODE	HIGH	
Event description	This event will be generated if FDIR decides to transition the system into Safe Mode		
Reaction	–		

Event 10.8: WD_SW_RESET

	Event Name	Event Severity	Event Parameter
	WD_SW_RESET	HIGH	
Event description	This event will be generated if a the software watchdog reboots the SW		
Reaction	–		

Event 10.9: IO_SWITCH

	Event Name	Event Severity	Event Parameter
	IO_SWITCH	MEDIUM	Device object-ID
Event description	This event will be generated if FDIR decides to switch to the secondary I/O-Board due to PCDU or I/O-Board failures		
Reaction	–		

Event 10.10: SPW_SPWSTR_EE

	Event Name	Event Severity	Event Parameter
	SPW_SPWSTR_EE	LOW	
Event description	Early End of Packet		
Reaction			

Event 10.11: SPW_SPWSTR_IA

	Event Name	Event Severity	Event Parameter
	SPW_SPWSTR_IA	LOW	
Event description	Invalid Address		
Reaction			

Event 10.12: SPW_SPWSTR_WE

	Event Name	Event Severity	Event Parameter
	SPW_SPWSTR_WE	LOW	
Event description	Write Synchronization Error		
Reaction			

Event 10.13: SPW_SPWSTR_PE

	Event Name	Event Severity	Event Parameter
	SPW_SPWSTR_PE	LOW	
Event description	Parity Error		
Reaction			

Event 10.14: SPW_SPWSTR_DE

	Event Name	Event Severity	Event Parameter
	SPW_SPWSTR_DE	LOW	
Event description	Disconnect Error		
Reaction			

Event 10.15: SPW_SPWSTR_ER

	Event Name	Event Severity	Event Parameter
	SPW_SPWSTR_ER	LOW	
Event description	Escape Error		
Reaction			

Event 10.16: SPW_SPWSTR_CE

	Event Name	Event Severity	Event Parameter
	SPW_SPWSTR_CE	LOW	
Event description	Credit Error		
Reaction			

Event 10.17: SPW_SPWSTR_TO

	Event Name	Event Severity	Event Parameter
	SPW_SPWSTR_TO	LOW	
Event description	Tick Out		
Reaction			

10.3 Power FDIR

For reasons of readability in this chapter the sensors for electric current and electric voltages are referred to as V(olt)-sensor and A(mpere)-sensor. In diagrams, I0, I1, I2, IX are used for electric currents and U0, U1, U2, UX for voltages with X representing a random number 0 or 1 or 2 in case a diagram shows a general

approach valid for multiple electrical current (voltage) sensors. They are exchangeable with "A-sensor (V-sensor)".

The PCDU implements automatic emergency safety check functions to prevent the system from major failures. These safety checks include

- SoC observation,
- under-voltage protection,
- fuse current surveillance and fuse deactivation.

These checks are essential to trigger reactions in case where the OBSW is not controlling the satellite properly, which could happen during OBC reconfiguration (see also Sect. 10.2.8).

These safety checks are also active during normal operations. However the PCDU does not distinctly report to the OBSW when limits were exceeded and reactive measures were taken. The PCDU generates internal Events for these cases and stores them in its memory. However that information is not included in the HK TM that is periodically polled by the OBSW. This means that the OBSW and a ground operator are not immediately informed about PCDU reactions and thus it could happen that they do not know about the exact current state of the system. For FLP Generation 2 a PCDU firmware update is foreseen to provide full parameter visibility to the OBSW.

Detailed knowledge about the Power Supply System (PSS) including failure detection is represented in the OBSW by the *PSS-Controller* as was already mentioned in Sects. 3.2.4 and 5.6.

10.3.1 Battery State-of-Charge Surveillance

The *PSS_Controller* executes an algorithm for the battery SoC determination in every cycle, based on the voltage and electric current values provided by the sensors in the PCDU. The State-of-Charge threshold which is defined in the PCDU to trigger reactions is also represented in OBSW, but at system boot or reboot the OBSW it is not immediately operational. The PCDU uses voltage thresholds at the PCDU main switch as indicator for emergency shutdown.

OBSW lower limits values are defined to be higher than the trigger limits of the PCDU which during nominal operation leads to the OBSW triggering actions—

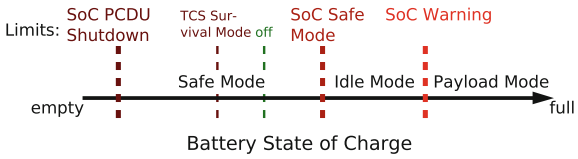


Fig. 10.16 Battery State-of-Charge Limits and Operational Ranges. © IRS, University of Stuttgart

such as Software Disconnection of Non-essential Loads (SW-DNEL)—before the PCDU algorithms do. This allows for the generation of according failure Events and for their storage in the Event TM store for ground visibility instead of a hard reset performed by the PCDU without further notice.

Thresholds for the State-of-Charge surveillance that trigger reactive actions, are presented in Fig. 10.16. The limits are determined to prevent the battery from taking damage through essential discharging.

SoC Warning at 55 % battery SoC
This limit is defined for any high-power payload scenario. It will most likely be generated when several payload components are operated simultaneously, increasing the overall power consumption of the satellite. Exceeding the threshold a warning Event—SOC_WARNING Event (RID 4012)—is issued that the battery needs recharging before further payload operation can be performed.

FDIR will initiate the deactivation of all payload devices to decrease the power consumption. The spacecraft will enter Idle Mode, and will try to take a steady configuration. The satellite is oriented towards the solar panels towards the Sun to recharge the battery.

The threshold is chosen such that the system can easily survive another half orbit without charging in case where the S/C enters the eclipse right away. This is sufficient due to a reliable determination of the Sun direction even in the eclipse. The battery charging continues after the eclipse end.

SoC Safe Mode at 45 % battery SoC
If for some reason the satellite will not be able to point towards the Sun and therefore cannot recharge the battery the SoC will drop further. At reaching the next threshold *SoC Safe Mode, FDIR* will trigger the SOC_CRITICAL Event (RID 4013) which leads to deactivation of all Idle Mode components to further decrease the system's power consumption.

This situation should only be reached if more than one malfunction has occurred in the system. The satellite will then operate in Safe Mode and will attempt to recharge the battery while coarsely pointing towards the Sun using only ACS devices MGM and MGT.

The SoC level for *SoC Safe Mode* is selected such that a complete orbit could be survived in Safe Mode without recharging the battery. The assumption is, that it takes a maximum duration of one orbit to properly detumble the satellite and to re-establish a system state which enables steady charging of the battery.

TCS Survival Mode at 35 % battery SoC
If the SoC falls below a further threshold the battery approaches a critical SoC limit that could lead to physical damage. In such case the Thermal Control System is switched into the Survival Mode to further decrease the system power consumption and reduces thermal conditioning to only the Safe Mode components. Payload devices are no longer heated and their damage is accepted in this mode to increase chances for a survival of the mission. The TCS Survival Mode is explained in Sect. 10.7.3.

TCS Survival Mode Off at 39 % battery SoC
To avoid frequent triggering of the TCS Survival Mode an SoC limit above 35 % is selected for re-entering the TCS Nominal Mode. If the battery is charged in the TCS Survival Mode up to 39 % the TCS is switched back into nominal operation. Since the S/C might be tumbling in that mode it is hard to determine a period of time that it will take to reach 39 % SoC. However, this limit is deemed sufficient to prevent immediate activation of the TCS Survival Mode again and to a dangerous loop.

SoC PCDU Shutdown at 25 % battery SoC
PCDU shutdown This is the lowest State-of-Charge level that will be detected by OBSW control and equals a battery voltage of approx. 20 V. If this level is reached, severe malfunctions have occurred on board the satellite and complete loss of the mission cannot be excluded.

When violating this threshold *FDIR* will isse the SOC_SHUTDOWN Event (RID 4014) and will write according safeguard memory entries to log the failure. This will include the battery SoC values (see Sect. 10.2.11). Finally the OBSW will command the PCDU into Shutdown Mode.

In this mode, the PCDU will deactivate all loads including the OBC and TTC systems to achieve an absolute minimum power consumption. If the OBSW is no longer able to perform this task, at a minimally lower level of 19 V bus voltage, the PCDU will perform the same deactivation of itself.

The satellite will not be actively orientated as long as the PCDU is in Shutdown Mode but it is assumed that the battery will be charged slowly during the uncontrolled tumbling since it will occasionally point towards the Sun. The PCDU will remain in standby with a very low power consumption but will still perform battery charge control.

Battery heater disabling Note that when the PCDU is commanded to Shutdown Mode, it also deactivates automatically the battery heaters' bi-stable relays (see Sect. 5.5.1) so that the battery is not heated anymore. As mentioned before, as long as the OBSW is active, the battery temperature will always be kept above 0 °C by the battery heaters, but the battery cells can be operated as long as their temperature remains above −30 °C. In the scenario considered here, the risk of severely damaging the battery by operating it outside of its operational temperature range needs to be traded against the risk of damaging it by discharging it below its lowest operational voltage (19 V) due to the battery heaters being activated.

Battery damage by undertemperature As the battery is thermally very well isolated against the satellite system, its heat capacity is fairly large and some heat is dissipated by the battery electronics at all times, cooling of the battery from 0 °C to −30 °C will take in the range of 24 h when no power is generated by the panels, depending on the satellite attitude and other factors. When a reasonable minimum of power is generated by the panels, more heat will be dissipated by the battery system, further reducing the risk of it reaching a critically low temperature.

Battery damage by undervoltage On the other hand when *FDIR* has commanded the PCDU to Shutdown Mode due to a low SoC and no power is generated by the panels in the following hours, the battery voltage may reach its lowest operational value of 19 V faster than the battery temperature reaches −30 °C. If the battery heaters were activated in such case, they would increase the permanent damage to the battery caused by discharging it below 19 V. Therefore, the bi-stable relays are deactivated when the PCDU is commanded to Shutdown Mode, since deep discharging would occur before the battery cools down below its operational limit. The likeliness of damage caused by deep discharging is considered more severe.

Note that power generated by the solar panels within a few hours after shutdown should prevent any permanent damage to the battery. Depending on the duration and severity of any non-operational conditions of the battery (be it a low temperature or a low voltage), permanent battery capacity loss will be the result of such conditions.

PCDU recovery Upon reaching a voltage level above 21.5 V the PCDU will enter Recovery Mode 0 during which the PCDU heaters will be activated to achieve a PCDU temperature of at least −40 °C. Furthermore in this PCDU mode the SoC estimation for the battery is resumed. With proper values from the SoC estimation the PCDU will boot and enter Recovery Mode 1.The transitions for the PCDU from Shutdown/Boot/Launch Mode to MiniOps and Safe Mode can be taken from Sect. 5.5.3—Fig. 5.12. The SoC level for the PCDU to re-initiate the boot up sequence is selected such that a direct fallback into shutdown mode is avoided.

The PCDU will deactivate its heaters as soon as its temperature rises above the minimum operational limit of −40 °C. It is assumed that the dissipated heat of the operating PCDU is sufficient to keep the PCDU in its operational temperature range.

Depending on the duration during which the satellite was uncontrolled, the temperature in the system might have dropped. Therefore from Recovery Mode 1 onwards also the OBC and TTC will be heated until reaching their minimum operational temperature. For the OBC this is controlled by bi-metal switches in the housing. Having reached the required temperature, the PCDU will identify the current dropping to 0.0 A for the OBC and TTC heater circuits. In the following PCDU MiniOps Mode the OBC and TTC will be activated as well as the ACS Safe Mode component power lines. The OBC and OBSW will boot and take over control of the satellite again and will attempt a detumbling of the satellite to enter Safe Mode and eventually to re-establishing a steady state.

10.3.2 Voltage Levels and Fuse Currents

It is not only the battery SoC that needs careful monitoring to avoid to damage devices or the spacecraft system. The diverse on-board equipment devices must be protected against overvoltage and overcurrent. For this the FLP PCDU contains latching current limiters (LCLs) which are called “Fuse” by the manufacturer.

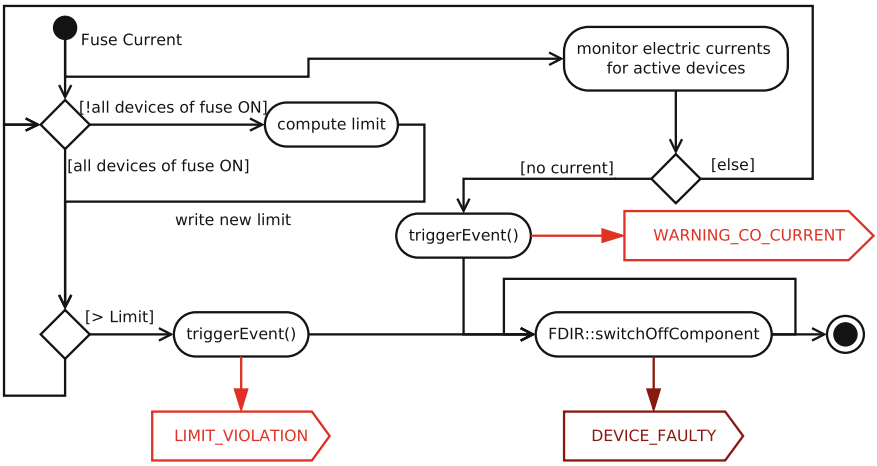


Fig. 10.17 Fuse Current Limit. © IRS, University of Stuttgart

Due to volume constraints inside the PCDU box of FLP Generation 1 not all devices on board the spacecraft will be secured via an individual fuse. Some of the fuses are used for two or three independent—meaning not redundant—devices. Even if multiple devices are secured by one fuse, each device has its individual power switch inside the PCDU.

For multiple devices connected to one fuse the maximum permissible power value is derived from the combined power consumption of all (refer to Table 17.10 in the annex for fuse/device assignment and power consumption limits). The PCDU in its HK data can only provide the currents per fuse, not per connected device. Therefore, an additional analytical mechanism is required for device protection.

The *PSS_Controller* stores a table with the maximum characterized power consumption for each device. Depending on which devices on a particular fuse are powered the upper limit is computed as depicted in Fig. 10.17. The provided fuse current is compared against the limit and upon exceeding the limit an Event is generated. *FDIR* will then deactivate all devices connected to that fuse since with only a LIMIT_VIOLATION event as input *FDIR* cannot isolate the malfunctioning one.

If only one device is connected to a particular fuse or if only one device on a fuse for multiple unis is powered, the approach to deactivate all of them is still appropriate. No different method is required in the logic.

The operator on ground then will need to perform further investigation to determine which of the devices showed the increased power consumption. If a device shows no power consumption at all despite a closed PCDU switch and not open fuse the *PSS_Controller* will trigger the Event WARNING_NO_CURRENT which leads to *FDIR* switching off this device. The *PSS_Controller* needs the following information:

- All switch statuses
- All fuse statuses
- Fuse ↔ device assignment
- Switch ↔ device assignment
- Nominal/maximum device power consumption table

The power monitoring is performed by the *PSS_Controller* in the OBSW. It has the task to observe the electric fuse currents and to initiate isolation and recovery actions before the PCDU built-in functions (fuses) need to.

10.3.3 Failure Handling for PSS Devices

In the PCDU one battery string is charged through the power output of one dedicated solar panel. There is no cross-coupling between battery strings and solar panels—see Fig. 5.9 for details. Charging can be activated by closing the Charge-Switch. A battery string having reached its maximum charge status—i.e. exceeding the maximum permissible voltage—is reported to the PCDU by the battery electronics. The PCDU therupon opens the Charge-Switch of the affected string.

For the solar panels and cells, electric voltage and current values are observed at all times by the *PSS_Controller*. However since there is no way to recover from solar cell defect these control mechanisms only raise informational Events TM for the ground operator to potentially make adaptions to the power control algorithms.

The deployment of the two side panels of the solar array is detected by the PCDU and is stored in an SA-Deployed flag in the OBSW. At OBSW boot/re-boot the *PSS_Controller* reads the values of the solar panel deployment Reed sensors to get information about the state of the solar panel deployment mechanism during the OBSW boot process. Solar panel deployment will be attempted after a specified time of 180 s after the separation is detected. If the solar panel deployment has not been successfully conducted after five attempts, ground interaction is required.

One difficulty in modeling failures in the PCDU is that it is hard to distinguish between failures in the sensor, and failures in the device which this sensor is helping to observe. This problem occurs at two different points in the PCDU system. Figure 5.9 shows that each of the lines to the battery strings (top) and the solar panels (left side) are equipped with only one V-sensor and one A-sensor. If one of the sensors returns an unexpected value it is not clear immediately whether the sensor is wrong or the device is damaged. The following sections provide the considerations to evaluate sensor measurements that show unexpected behavior.

10.3.3.1 Overall System Power Budget

The PCDU HK TM packet includes the electric currents and voltages of all solar panels (SP) and battery strings (Bat) as well as an electric current and voltage for the main power-bus. The values are tracked at the measurement point marked with "UMPB" (main power-bus) in Fig. 5.9 on the right side at the bottom. The total power budget can be calculated in two ways:

$$P_{in} = 3 \cdot (U \cdot I)(SP) + 3 \cdot (U \cdot I)(Bat) \tag{10.1}$$

$$P_{out} = I \cdot U(MPB) \tag{10.2}$$

The voltage drop over the PCDU main switch is in the range of some 100 mV and thus is negligible for the energy budget calculation on board. As the BCRs dissipate some power the input power will be marginally higher than what the *PSS_Controller* will compute using bus voltage and current.

If the comparison proves wrong, a failure in a sensor can be the source. A failure of the main power-bus sensors cannot be isolated because no reasonable limits can be determined for these without complex considerations. The electric voltage depends on the input via battery or solar panels and the electric current on the main power-bus strongly depends on the current spacecraft configuration and maneuver.

10.3.3.2 Solar Panel Failure Handling

Solar panels in this context also include their respective A- and V-sensors in the PCDU since they are required to determine the health state of the panels. Figure 10.18 illustrates the case of sensors showing unreasonable values.

Due to the nature of the ADC converter in the PCDU the values for electric current and voltage will always include noise. Therefore the value is assumed not to be equal to '0' even if there is no current or voltage present. Mentioning a value of '0' in the following tables actually means a value between 0−dx…0+dx with dx being specified as dx = 0.2 V.

Tables 10.5 and 10.6 show the conflict of detecting a failure if a sensor does not provide a measurement due to a broken wire, which actually is the same as showing '0'. The illustration in Fig. 10.18 shows the mechanism for treating the cases of Tables 10.5 and 10.6.

The *sunFlag* is set in the *ACS_Controller* according to the algorithm described in [125]. If *sunFlag* is '0', the S/C is orbiting through the eclipse.

The Sun vector cannot not be determined reliably in the pointing modes. However it is not required for such a mode since attitude determination is conducted with FOG and STR systems. Therefore failure detection of solar panels is not performed during pointing modes since they are not permanently directed towards the Sun. The same is true for the S/C being in the eclipse. The failure

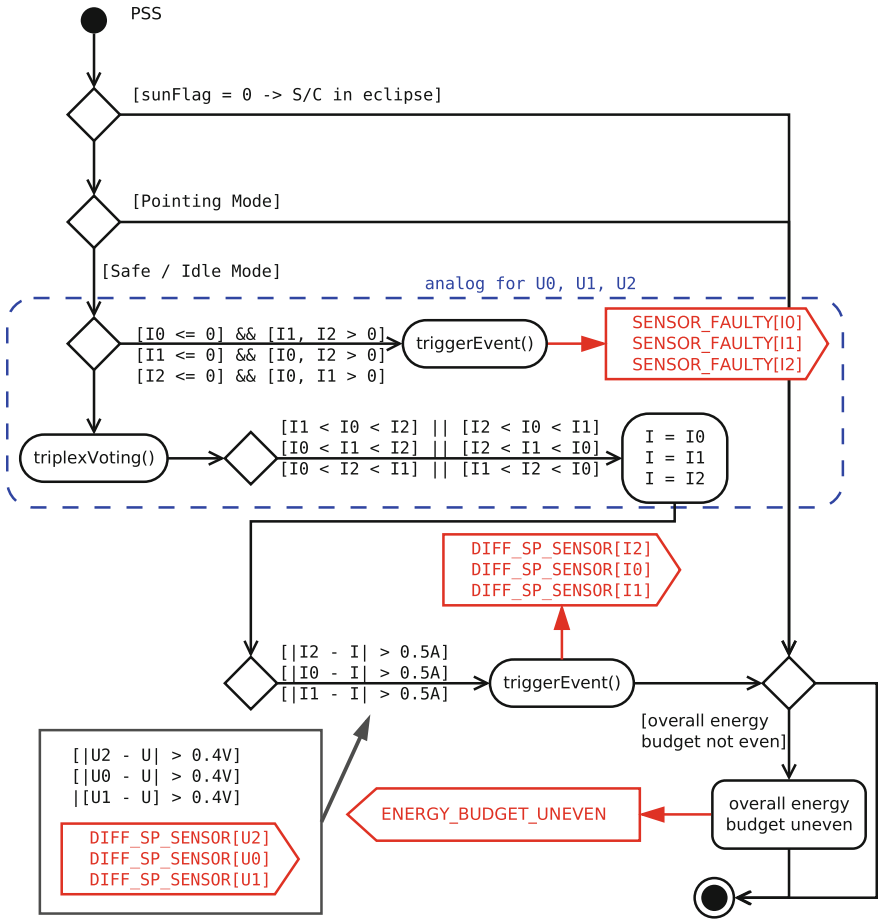


Fig. 10.18 Solar Panel and Sensor Failure Detection. © IRS, University of Stuttgart

detection is only performed during Idle and Safe Mode. The flow in Fig. 10.18 is described hereafter:

- If the satellite is in the eclipse or in a pointing mode (Inertial, Target, Nadir, Coarse Nadir) no failure detection for solar panels is performed but the total energy budget is calculated as described in Sect. 10.3.3.1.
- If the satellite is in Safe Mode or Idle Mode the goal is to direct the solar panels towards the Sun. Therefore—and unless a detumble maneuver has to be performed—the values of the electric current and voltage should be greater than '0'.
- If a sensor is registered to show '0' while the other two sensors show a value greater than '0', this sensor is reported as faulty.

Table 10.5 Failure modes of solar panel A-/V-sensor (1)

Occurrence	A-sensor shows '0'	
Assumption	Sensor is correct	Sensor is broken
Cases	Solar panels not directed towards Sun → check attitude/pointing	Battery strings are still charged correctly → overall energy budget "computed input from solar panels ↔ measured ouput on S/C bus" does not come out even
	Satellite in eclipse → check sunFlag	
	(line to) solar panels broken/malfunctioning	

Table 10.6 Failure modes of solar panel A-/V-Sensor (2)

Occurrence	V-sensor shows '0'	
Assumption	Sensor is correct	Sensor is broken
Cases	Solar panels not directed towards Sun → check attitude/pointing	Battery strings are still charged correctly → overall energy budget "computed input solar panels ↔ measured ouput S/C bus" does not come out even
	Satellite in eclipse → check sunFlag	
	(line to) solar panels broken/malfunctioning	

- If a different case occurs, a triplex voting is performed as shown in Fig. 10.18. The value in the middle of the two extremes is taken as basis for the voting mechanism. If the electric current of one sensor differs from the voting reference by more than 0.5 A the identified deviation is reported with an Event. If the voltage of one sensor differs from the basis voltage for more than 0.4 V the identified deviation is reported with an Event.
- If the overall budget is incorrect it is reported.

The evaluation of Fig. 10.18 in *FDIR* is detailed in Fig. 10.19. The counters are introduced because Events might be triggered—e.g. through particle impact on the solar panels—that induce a short-term higher/lower voltage output. As a tolerating measure the devices are not marked as faulty instantaneously but after multiple occurrences of such Events.

- If both current and voltage for the same solar panel are reported to be '0' the failure counter for the solar panel is increased until reaching 2.
- If only one sensor is reported either to be '0' or to be voted out its failure counter is increased until reaching 4.
- If either counter reaches its limit the corresponding device is marked as faulty by *FDIR*.

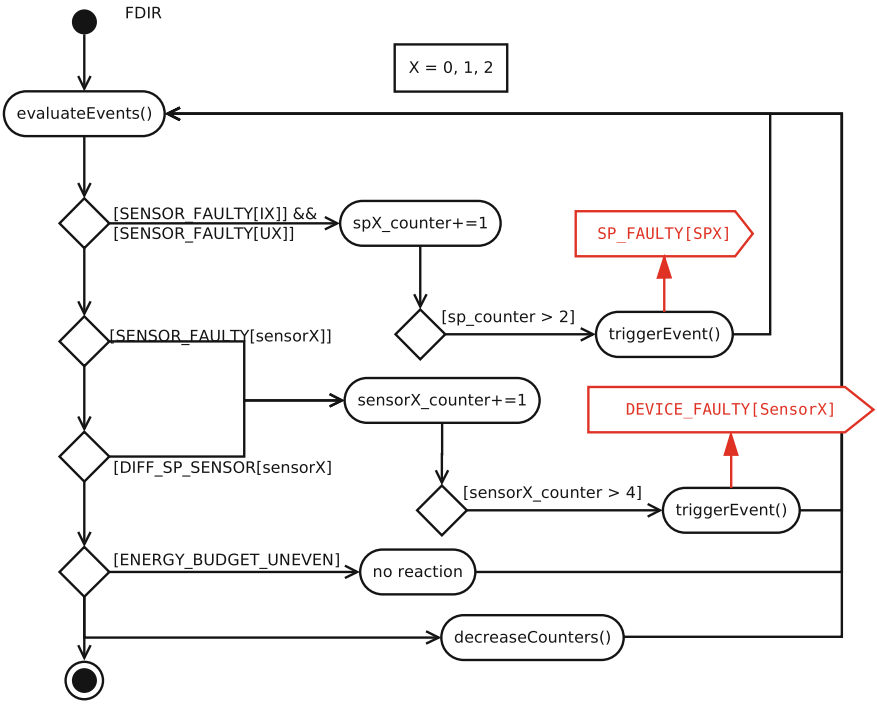


Fig. 10.19 Solar Panel Failure Evaluation in the System FDIR. © IRS, University of Stuttgart

- In case the energy budget is detected to be uneven, ground interaction will be required. No reaction will be performed by OBSW *FDIR*.
- If no failures are detected the counters are decreased. This is a simplified version of the device failure mechanism from Sect. 10.4 to reduce complexity.

10.3.3.3 Battery Failure Handling

In this section the potential failures of battery strings are considered. Battery strings, in this context also include the respective A- and V-sensors since they are required to determine the string statuses. The consideration is comparable to the one in the previous section for the solar panels and their sensors.

Table 10.7 shows the PSS Event LIMIT_VIOLATION[IX, HIGH]. It will be generated if one of the A-sensors shows an electric current exceeding 12 A. In a nominal case this value should never be reached. Two scenarios are possible root causes for this symptom. In both cases further investigation is required by ground to recover the system.

Table 10.7 Failure modes of battery string A-/V-sensor (1)

Occurrence	A-sensor shows >12A	
Event	LIMIT_VIOLATION[IX, HIGH]	
Assumption	Sensor is correct	Sensor is broken
Cases	Sensor shows too high charge current. Risk of battery heating up and damaging	Sensor shows wrong measurement
Reaction	Battery charging will be disabled by deactivation of BCR	Risk is too high, battery charging is disabled. ground interaction required

Table 10.8 Failure modes of battery string A-/V-sensor (2)

Occurrence	A-sensor shows < -12A	
Event	LIMIT_VIOLATION[IX, LOW]	
Assumption	Sensor is correct	Sensor is broken
Cases	Sensor shows too high a discharge current. Might be a short circuit in the system. This should be isolated by PSS fuse current surveillance or PCDU fuse deactivation	Sensor shows wrong measurement. Other sensors should show correct values
Events	FUSE_DEACTIVATED or FUSE_OVERCURRENT	DIFF_BAT_SENSOR[IX]
Reaction	None necessary	Both events will lead to marking the sensor as faulty

(a) The sensor works correctly and the battery is charged with a current greater than it is supposed to be. In such case the battery could overheat and be damaged. *FDIR* will disable the charge process by deactivating the corresponding BCR.
(b) The sensor itself is defective and provides a corrupted value. Here the risk of battery damage through heating up is too high and *FDIR* will also disable the charge process.

Table 10.8 shows the PSS Event LIMIT_VIOLATION[IX, LOW]. This Event will be generated when the A-sensor of a battery string shows a value below −12 A. This implies that the discharge current for a battery string is too high. Two different root causes might lead to this Event.

(a) The sensor is working correctly and the discharge current is too high. This could result from a short circuit somewhere in the system and the battery (all strings) being discharged quickly. No direct reaction is required because this should lead to a subsequent step of failure detection.
(b) Either the *PSS_Controller* detects one or more devices drawing excessive power and issues an Event FUSE_OVERCURRENT[fuseID] which results in the corresponding switches being deactivated by *FDIR*.

(c) Or the OBSW is too slow with the deactivation and the PCDU fuse surveillance already undertook the deactivation of the corresponding fuse by itself. In that case, the *PSS_Controller* will generate a FUSE_DEACTIVATED Event.
(d) The sensor is broken and shows an incorrect value. In such case the other two battery A-sensors should show a correct value and a significant difference between the A-sensor values should be detected. This will lead to the Event DIFF_BAT_SENSOR[IX]. If both Events are recognized by *FDIR* the identified sensor will be marked as faulty.

The cases of Tables 10.8 and 10.9 are illustrated in Fig. 10.20 with the *PSS_Controller* and in parallel with the corresponding *FDIR* reactions.

- Starting at the top left the *PSS_Controller* checks for a charge current that is too high. This could happen if the regulation with the BCRs is defective. However it is not likely to occur. If an electric current exceeding 12 A is detected a LIMIT_VIOLATION event is generated. Its parameters comprise the sensor that caused the limit violation and a parameter to indicate that the measurement violated the upper limit. Upon this limit violation—since unlikely to appear but dangerous for the battery string—*FDIR* disables the charge process by deactivating the BCR for that battery string.
- If a limit violation of the lower limit –12 A is detected too much power is drawn from one or all battery strings. This could happen in case of a short circuit somewhere in the system. This is covered by the fuse surveillance of OBSW or PCDU and does not need a particular reaction of *FDIR*.
- If the measurements are within –12 … +12 A, a triplex voting is performed to identify measurements that are outside of the allowed range. It is expected that the A-sensors show similar values at all times within a range of ±2 A. In the triplex voting the value in between the two others is chosen as basis. If one of the other two sensor measurements are ±2 A below or above the basis value, this one is reported. This should not happen and indicates a failure with the A-sensor. It is marked as faulty by *FDIR*.
- In case where a FUSE_OVERCURRENT Event is reported, *FDIR* will disable the fuse.

Table 10.9 Failure modes of battery string A-/V-sensor (3)

Occurrence	V-sensor shows significant undervoltage	
Event	BATTERY_STRING_UNDERVOLTAGE	
Assumption	Sensor is correct	Sensor is broken
Cases	Battery String is not charged anymore	The overall energy budget should show a significant difference between input and output
Events	No additional event	ENERGY_BUDGET_UNEVEN
Reaction	Needs to be observed by ground FDIR.	Both events will lead to marking sensor as faulty

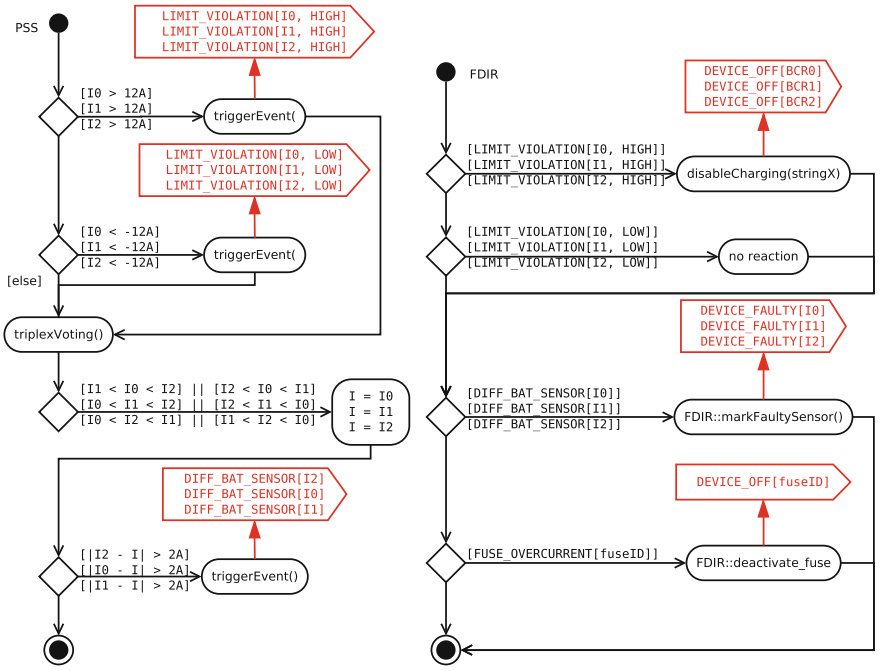


Fig. 10.20 Battery and Sensor Failure Detection. © IRS, University of Stuttgart

Table 10.9 shows the PSS Event BATTERY_STRING_UNDERVOLTAGE. For each battery string separate Events are generated considering their importance and sensitivity to damage. It will be generated if the sensor of a battery string shows a voltage below 20.5 V. Again two scenarios are to be considered:

(a) The sensor works correctly and one of the strings seems not to be charged anymore. This will require ground investigations to decide further actions. No action will be performed by *FDIR*.
(b) The sensor shows a wrong value. In that case the battery is fine but the balance of the overall power consumption should indicate a power difference between power input and output (see also Sect. 10.3.3.1). This generates a second Event ENERGY_BUDGET_UNEVEN. If both Events are generated within one PST cycle it is assumed that the sensor is broken and the battery works fine. The sensor will be marked as faulty by *FDIR* as shown in Fig. 10.21.

In Table 10.10 the Event BATTERY_STRING_OVERVOLTAGE is described. It will be generated if the V-sensor of a battery string shows that the voltage level of a string is measured to exceed 26 V. Nominally this Event should never occur because the BCRs limit the voltage. However it could also mean that the sensor shows a corrupted value. The two following scenarios are possible (see also Fig. 10.22):

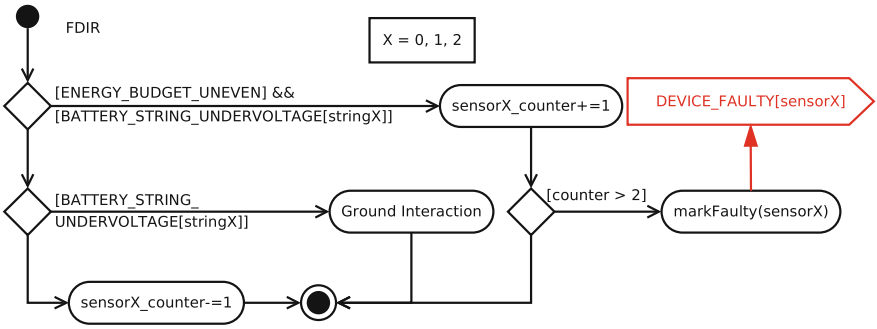


Fig. 10.21 Battery string undervoltage Event and isolation. © IRS, University of Stuttgart

Table 10.10 Failure modes of battery string A-/V-sensor (4)

Occurrence	V-sensor shows significant overvoltage >26 V	
Event	BATTERY_STRING_OVERVOLTAGE	
Assumption	Sensor is correct	Sensor is broken
Cases	Battery string shows overvoltage	Sensor shows wrong measurement. Other sensors should show correct values
Events	none	DIFF_BAT_SENSOR
Reaction	BCR will be deactivated	Both events will lead to marking sensor as faulty

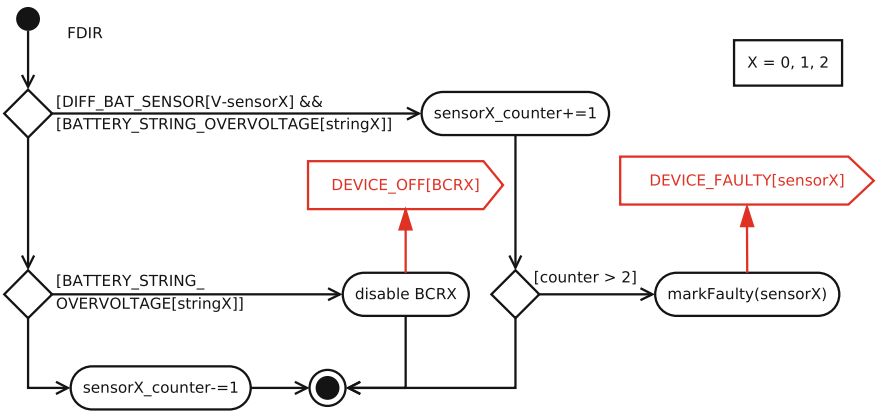


Fig. 10.22 Battery string overvoltage Event and isolation. © IRS, University of Stuttgart

(a) The sensor shows a correct value and the voltage level of one string is too high. If no other Event is generated this will require ground investigations for further actions. The charge process will be disabled and the BCR will be deactivated.
(b) The more likely case is that the sensor is broken and shows a wrong value. In that case the sensors of the two other strings will show a correct value and the Event DIFF_BAT_SENSOR[V-sensorID] is generated. Upon detecting both Events *FDIR* will mark the corresponding V-sensor as faulty.

If there is a significant difference between all sensors for both the A-sensors and the V-sensor of a battery string it is very likely that the battery string itself is faulty or the line to the battery string is damaged. In that case a DIFF_BAT_SENSOR Event is reported for both A- and V-sensor of a string. *FDIR* disables charging the string and provides a DEVICE_FAULTY Event to support further ground analysis as shown in Fig. 10.23.

Finally there exists the potential failure case of the battery temperature exceeding its permitted upper limit, which is defined to be 45 °C, i.e. 10 K below the operating limit given by the manufacturer of the cells [58]. Therefore each battery string is equipped with two temperature sensors for thermal monitoring. In case where the temperature limits for safe battery operations are violated this is detected—first by the *TCS_Controller* which then throws a COMPONENT_TEMP_OOL_HIGH event (RID 5804) containing the object-ID of the affected string (see Sect. 10.7).

In such case *FDIR* disables the string charging by disabling the charge activation flag of the affected string in the PCDU to prevent irreversible damage of the cells. This constitutes a severe *FDIR* case and ground intervention is needed. If no ground intervention happens quickly after the error occurred, the battery will be discharged subsequently and the SoC will fall continuously, which in turn causes loads to be deactivated at different levels (see Sect. 10.3.1). This however has the positive

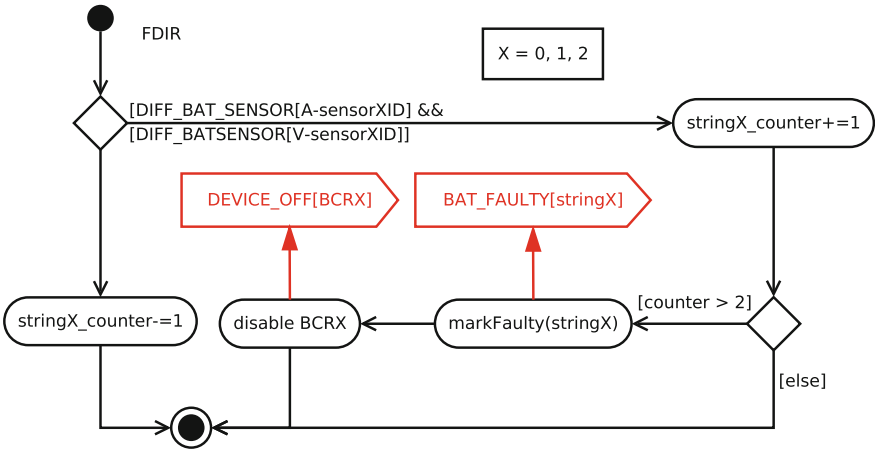


Fig. 10.23 Battery string faulty. © IRS, University of Stuttgart

effect to reduce the heat dissipated within the satellite system. The charge activation flag must be reset by ground or is reset automatically by the PCDU when it reboots, which will be the case once the SoC drops below its lowest limit of 25 %.

10.3.4 System Reactions

Section 10.3.1 introduced thresholds for the energy that is stored in the battery. The survival of the battery is essential for an operational mission. Depending on the SoC level according reactions have been defined to decrease the overall power consumption and to allow for re-charging of the battery strings.

In Sect. 10.3.2 the monitoring of single devices and fuse currents was introduced. Depending on the occurring Event, reactions will be executed as defined or are initiated automatically. If the electric current of one device or of a device-group on a fuse is too high the fuse is disabled. This leads to the device not being available for further operations. If the corresponding assembly of the device will report that it cannot keep the current mode because of missing redundancies this leads to a spacecraft mode not being executable anymore. The *System* will automatically transition into a defined fallback mode according to the *PSS_Controller* mode table.

If as explained in Sect. 10.3.3 a solar panel or battery string is identified as being faulty, the system will be set into Safe Mode immediately. A reduction of the overall power consumption is attempted to allow for ground to analyze the problem and to elaborate adequate actions. A switch into the Safe Mode is also performed when the charge process behaves unexpected and is disabled. If the voltage of a battery string is determined to be too low the balancing in the PCDU automatically draws power from the healthy strings. Events that initiate an immediate Safe Mode switch are shown in Fig. 10.24. However, the *PSS_Controller* needs to make sure that at no point all BCRs are marked faulty together since this could cause a mission loss. There exists a safeguard mechanism that prevents *FDIR* from disabling all BCRs at the same time.

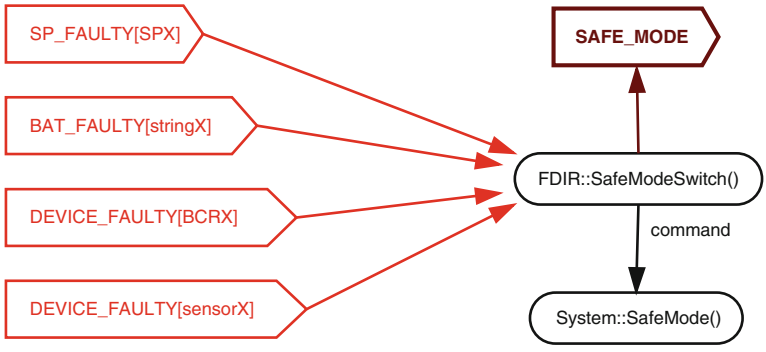


Fig. 10.24 PSS Safe Mode switch Events. © IRS, University of Stuttgart

10.3.5 PSS FDIR Events

Event 10.18: SOC_WARNING

	Event Name	Event Severity	Event Parameter
	SOC_WARNING	LOW	–
Event description	This event will be generated if the State-of-Charge of the battery drops below 55 %		
Reaction	FDIR initiates transition into Idle Mode to switch off payload components		

Event 10.19: SOC_SAFE_MODE

	Event Name	Event Severity	Event Parameter
	SOC_SAFE_MODE	HIGH	–
Event description	This event will be generated if the State-of-Charge of the battery drops below 45 %		
Reaction	FDIR initiates transition into Safe Mode to reduce the overall power consumption		

Event 10.20: TCS_SURVIVAL_MODE

	Event Name	Event Severity	Event Parameter
	TCS_SURVIVAL_MODE	HIGH	–
Event description	This event will be generated if the State-of-Charge of the battery drops below 35 %		
Reaction	FDIR initiates transition of TCS into Survival Mode to further reduce the overall power consumption. In this mode, payload is not heated anymore, payload damage is accepted to increase chances for mission survival		

Event 10.21: TCS_SURVIVAL_MODE_OFF

	Event Name	Event Severity	Event Parameter
	TCS_SURVIVAL_MODE_OFF	LOW	–
Event description	This event will be generated if the State-of-Charge of the battery exceeds 39 % after the TCS Survival Mode had been activated		
Reaction	FDIR commands TCS into nominal mode again to prevent the system from cooling off too far		

Event 10.22: SOC_SHUTDOWN

	Event Name	Event Severity	Event Parameter
	SOC_PCDU_SHUTDOWN	HIGH	–
	Event Name	Event Severity	Event Parameter
Event description	This event will be generated if the State-of-Charge of the battery drops below 25 %		
Reaction	FDIR commands the PCDU into Launch Mode to decrease the system power consumption to the lowest possible value. The PCDU switches off the complete system until a higher SoC is reached		

Event 10.23: WARNING_NO_CURRENT

	Event Name	Event Severity	Event Parameter
	WARNING_NO_CURRENT	LOW	Device object-ID
Event description	This event will be generated if a device is switched on but shows nopower consumption		
Reaction	FDIR switches off the component		

Event 10.24: LIMIT_VIOLATION

	Event Name	Event Severity	Event Parameter
	LIMIT_VIOLATION	LOW	Object-ID, HIGH/LOW
Event description	This event will be generated if an upper or lower limit is exceeded in a sensor measurement. The objectID indicates the according sensor		
Reaction	FDIR counts the events to reach a decision or marks the sensor as faulty immediately		

Event 10.25: DEVICE_FAULTY

	Event Name	Event Severity	Event Parameter
	DEVICE_FAULTY	MEDIUM	object-ID
Event description	This event will be generated if FDIR decides to switch off a device and mark it as faulty		
Reaction	–		

Event 10.26: DEVICE_OFF

	Event Name	Event Severity	Event Parameter
	DEVICE_OFF	MEDIUM	object-ID
Event description	This event will be generated if FDIR disables a device or BCR		
Reaction	–		

Event 10.27: SENSOR_FAULTY

	Event Name	Event Severity	Event Parameter
	SOC_SAFE_MODE	LOW	Sensor object-ID
	Event Name	Event Severity	Event Parameter
Event description	This event will be generated if a sensor shows unexpected behavior		
Reaction	FDIR counts the events to reach a decision or marks the sensor as faulty immediately		

Event 10.28: DIFF_SP_SENSOR

	Event Name	Event Severity	Event Parameter
	DIFF_SP_SENSOR	LOW	SP Sensor object-ID
Event description	This event will be generated if a A-(V-)sensor around the solar panels shows a different behavior than the other two solar panel A-(V-) sensors		
Reaction	FDIR counts the events to reach a decision or marks the sensor as faulty immediately		

Event 10.29: DIFF_BAT_SENSOR

	Event Name	Event Severity	Event Parameter
	DIFF_BAT_SENSOR	LOW	Bat Sensor object-ID
Event description	This event will be generated if a A-(V-)sensor around the battery strings shows a different behavior than the other two battery string A-(V-)sensors		
Reaction	FDIR counts the events to reach a decision or marks the sensor as faulty immediately		

Event 10.30: ENERGY_BUDGET_UNEVEN

	Event Name	Event Severity	Event Parameter
	ENERGY_BUDGET_ UNEVEN	LOW	Sensor object-ID
Event description	This event will be generated if the calculation introduced in 10.3.3.1 - Overall System Power Budget is not even		
Reaction	FDIR uses this event to reach a decision for faulty sensors. If no further event is noted in the PST cycle, no reaction is initiated but the ground operator is informed		

Event 10.31: SP_FAULTY

	Event Name	Event Severity	Event Parameter
	SP_FAULTY	MEDIUM	SP object-ID
Event description	This event will be generated if a solar panel is assumed to be faulty		
Reaction	FDIR switches the system into Safe Mode immediately		

Event 10.32: BAT_FAULTY

	Event Name	Event Severity	Event Parameter
	BAT_FAULTY	HIGH	BAT object-ID
Event description	This event will be generated if a battery string is assumed to be faulty		
Reaction	FDIR switches the system into Safe Mode immediately		

Event 10.33: BATTERY_STRING_UNDERVOLTAGE

	Event Name	Event Severity	Event Parameter
	BATTERY_STRING_UNDERVOLTAGE	HIGH	Battery string object-ID
Event description	This event will be generated if an unexpected low voltage level is detected for a battery string		
Reaction	If the overall energy budget is uneven, this event is assumed to show a sensor failure. The sensor is marked as faulty. Otherwise the string is not charged anymore. No reactions from FDIR are undertaken		

Event 10.34: BATTERY_STRING_OVERVOLTAGE

	Event Name	Event Severity	Event Parameter
	BATTERY_STRING_OVERVOLTAGE	MEDIUM	Battery string object-ID
Event description	This event will be generated if an unexpected high voltage level is detected for a battery string. This event is not expected during nominal operations since the solar panels do not provide a voltage level higher than 26 V. Therefore, the sensor should be faulty		
Reaction	If a difference in the voltage sensor measurements is detected as well, the sensor is marked as faulty. If no other event is detected, no reaction is required		

Event 10.35: SAFE_MODE

	Event Name	Event Severity	Event Parameter
	SAFE_MODE	HIGH	SP object-ID
Event description	This event will be generated if FDIR decides to command a transition into Safe Mode		
Reaction	–		

10.4 Equipment FDIR

10.4.1 Device Failure Versus I/O-Board Failure Symptoms

As described in Sect 3.2.2, formal information about the communication with devices on board an FLP-based satellite is stored in the software within d*eviceHandler* Objects. This information includes

- the communication protocol used by the device,
- the verification mechanism for the device response packets e.g. an algorithm to calculate a checksum,
- the position of the HK data in the packet, their format and calibration functions and
- the knowledge to interpret validity information which is delivered by the device itself in its response packet.

Checksum and protocol information is only available for devices that provide their data in digital form. As mentioned before, the PCDU is utilized as converter for all analog input from electric and temperature sensors. Sanity checks for these sensor measurements are performed in the subsystem controller objects in the OBSW. There exist no separate *deviceHandlers* for those sensors.

In this and the following chapter the term "device" always refers to components that provide digital information and whose response packets are interpreted by a *deviceHandler*. As a first point of contact for incoming data from devices, a *deviceHandler* object performs the formal checks of the response packet. On this level of knowledge, there are basically three types of error checks performed which may lead to possible failure Events:

(1) The message from the device hardware does not conform to the protocol that is expected
(2) The received message is empty
(3) Or a parameter in a device command response (set by the device's microcontroller/CPU) signals that the measurement of the device is not valid (e.g. for the STR).

If one of these cases occurs, the device handler object generates a *Failure Event Message* (see also Sect. 3.3.3) with the according *eventID*, the listed object-ID as the reporting object and the severity level *LOW*. The event will be forwarded to the global *FDIR* object as previously shown in Fig. 10.3. It indicates that something unexpected has occurred with the reporting device and reactive measures should be taken.

However, the challenge of failure handling on device level is that all devices that offer communication with the OBC/OBSW are not directly connected to the Processor-Board but instead through the I/O-Board, as can be seen in Fig. 1.6. This means that a failure event reported by a *deviceHandler* object could be induced by the device itself and some also by a malfunction in the I/O-Board as well. An

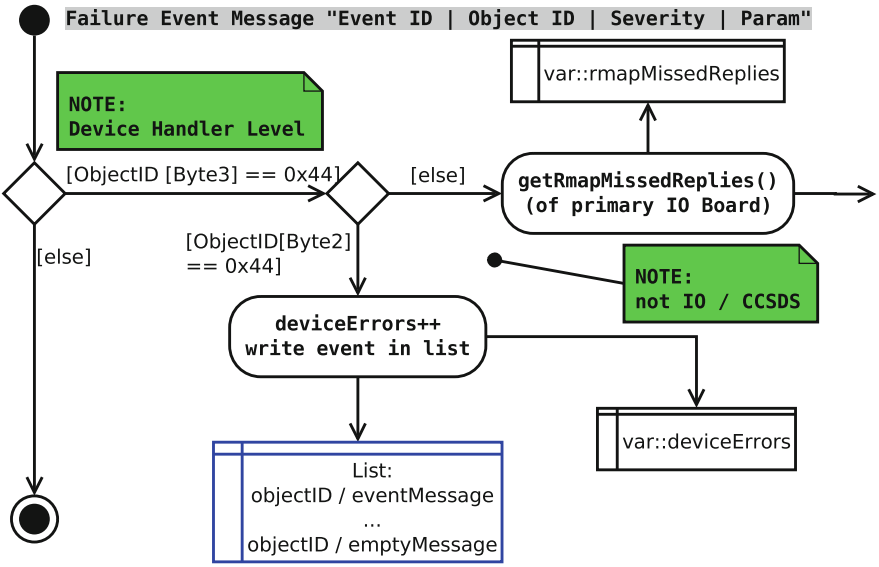


Fig. 10.25 Event message routing on deviceHandler object level. © IRS, University of Stuttgart

indication for a typical I/O-Board malfunction would be several *deviceHandler* objects reporting a missing response packet at the same time. The failure source cannot be determined uniquely considering a single event message at a time. Evaluation of failure messages on this level is performed after all event messages in one PST cycle have been forwarded and processed. The mechanism for device failure Event handling is explained in the following paragraphs.

The Fig. 10.25 depicts the Event message routing on *deviceHandler* object level:

- The first information required from the Event is the domain the Event was generated in. The 3rd Byte of the reporter object-ID is checked for the value '0 × 44'—compare to Table 3.1. This value indicates that a *deviceHandler* is the source of the Event. Only if this value is found, the following mechanism will take action. For the object-IDs see Sect. 3.2.8.
- A second check provides the information whether the Event was generated by a common device handler, or by a SpW device-managing handler. If the 2nd Byte of the reporter object-ID is '0 × 44' as well, the source is a common device handler.
- In that case the Failure Event Message will be stored into a two-dimensional {key/value}-list (in C++ called <map>) with the corresponding key being the device's object-ID. This will be performed for all Failure Event Messages of devices reported in one cycle.
- In the meantime a counter *deviceErrors in the OBSW* will be increased incrementally for every Failure Event Message stored in the list. After storing all

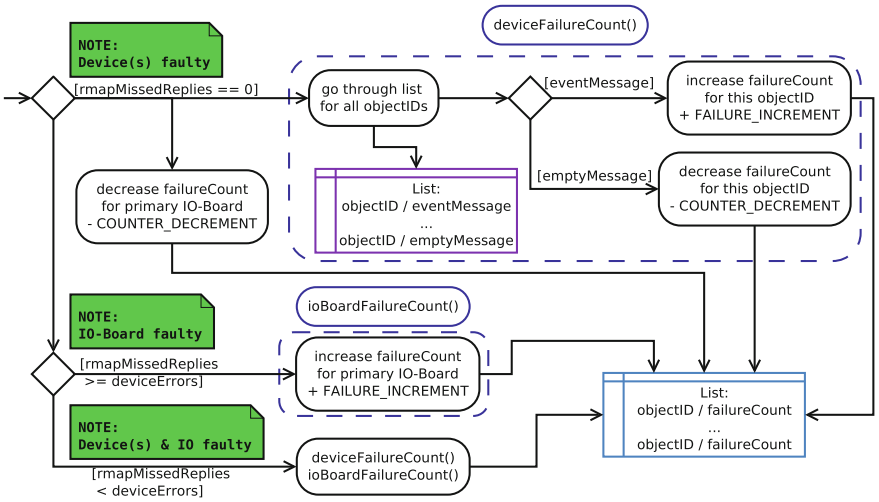


Fig. 10.26 Evaluation of deviceError and rmapMissedReplies counters. © IRS, University of Stuttgart

failure events, a function is executed that reports back *rmapMissedReplies,* the number of RMAP replies that have not been received during the past cycle.

Depending on the two values of *rmapMissedReplies* and *deviceErrors*, the evaluation of the device failure events can be performed. Figure 10.26 shows the three possible cases that need to be evaluated. Several SW-variables are included in the explanation of the mechanism.

- deviceErrors: counts the total number of device errors within one PST cycle.
- rmapMissedReplies: denotes the number of RMAP-packets that have been expected by the SW RMAP implementation to be provided by the I/O-Board, but have not been received. Expected RMAP packets—received RMAP packets.
- list {*objectID*/eventMessage}: A list of all *deviceHandler* objects, identified by their *objectID*, and a corresponding *eventMessage*. If an event is received for a certain *objectID,* this event is stored in the list. Otherwise, the message field is cleared for this *objectID*.
- list {*objectID*/failureCount}: A list of all *deviceHandler* objects, identified by their *objectID*, and a corresponding *failureCount*. The *failureCount* is incremented when FDIR identified a failure of that device in a PST cycle.
- FAILURE_INCREMENT: Constant variable.
- COUNTER_DECREMENT: Constant variable.
- FAILURE_INCREMENT and COUNTER_DECREMENT are variables to adjust the failure evaluation mechanism.

In Fig. 10.27 and based on that later in Fig. 10.30 the failure evaluation mechanism is visualized. The overall paradigm applied here comes from aircraft avionics architectures (see [145]). Failures in the device lead to a steep increase of

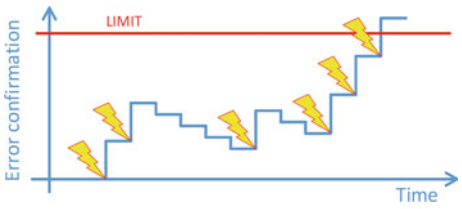


Fig. 10.27 Device failure evaluation mechanism [*from* 145]. © IRS, University of Stuttgart

the device failure counter by the value given through FAILURE_INCREMENT. Phases in which the devices work properly slowly decrease the counter by the value given through COUNTER_DECREMENT. If the failures are too numerous, the limit is exceeded and the device is rebooted or deactivated.
In the evaluation scheme three cases have to be determined:

1. The value of the variable *rmapMissedReplies* equals '0':
   This means that no malfunction has been reported for the I/O-Board. All events that have been reported for devices are really caused by these devices. In this case, a *deviceFailureCount()* function will be executed that steps through the previously filled {key/value}-list. If an event has been reported for an object-ID, a failure count is increased by FAILURE_INCREMENT in another {key/value}-list that stores a failure counter for each possible *objectID*.
   If there is no event for an object-ID, the failure count will be decreased by COUNTER_DECREMENT.
   The values of FAILURE_INCREMENT and COUNTER_DECREMENT serve the evaluation mechanism for unhealthy devices and are chosen such that a fine adjustment of the mechanism is possible without much effort.
2. The value *rmapMissedReplies* is equal to or greater than *deviceErrors*:
   This means that all reported device failures are caused by a malfunctioning I/O-Board, and the devices themselves are not faulty. Only the failure counter for the I/O-Board will be increased.
3. The value *rmapMissedReplies* is smaller than *deviceErrors*:
   This indicates a mixture of the two cases above, the I/O-Board is faulty, but also one or more device(s) is/are faulty as well. It can not be reasonably determined, which of the devices' failures are caused by the faulty I/O-Board so as general approach the failure counter is increased for all reported devices.

10.4.2 Device Failure Isolation and Recovery

It is expected that devices will show random errors during operation because of environmental effects or degradation and wear-out effects. The FLP Generation 1 payload computer in particular is not hardened against radiation and is expected to

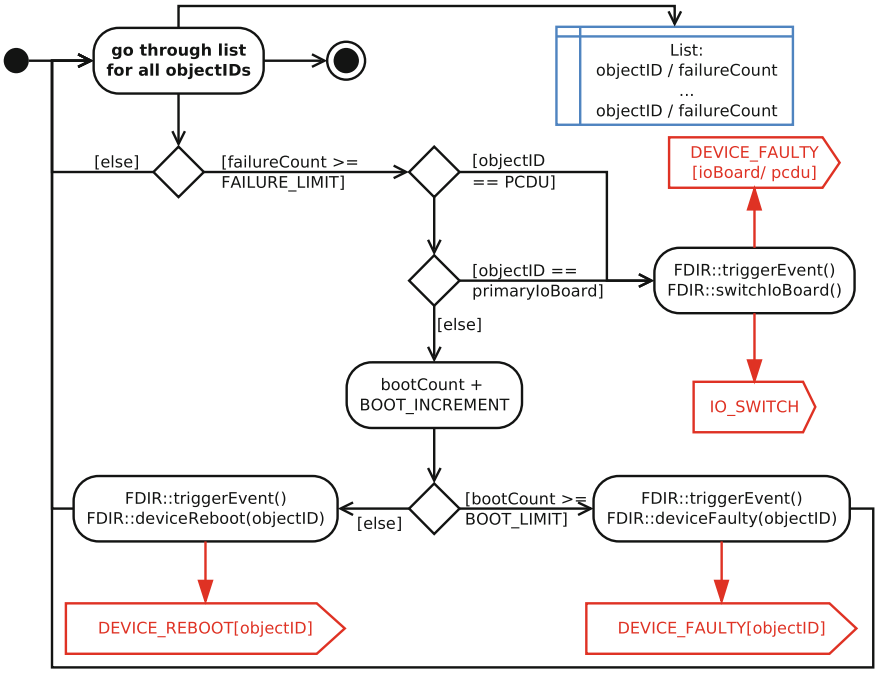


Fig. 10.28 Device failure evaluation. © IRS, University of Stuttgart

show distortion. Therefore the general situation is somewhat different compared to a commercial spacecraft which would not include such experimental components.

Thus, it is not considered reasonable to switch off a standard device which was marked faulty after only one malfunction symptom. Some devices—when malfunctioning due to a Single Event Upset (SEU)—can only be recovered through power-cycling, which is normally not the preferred method of treating the device. The failure evaluation paradigm based on the failure count already was sketched out in Fig. 10.27. It shall be further detailed now:

When the counter for the device exceeds a certain limit, recovery/reconfiguration is initiated by *FDIR*. After the above described functions of memorizing the Events and increasing the failure counters, an evaluation function as depicted in Fig. 10.28 is executed to determine the necessary actions of *FDIR*.

This function steps through the {*objectID*/failureCount}-list and evaluates the value of the *failureCount* variable for each object-ID.

- If *failureCount* exceeds FAILURE_LIMIT, *FDIR* determines the device that is faulty and distinguishes between the following cases:
  - If the faulty device is either the primary I/O-Board or the PCDU, the event DEVICE_FAULTY is triggered with [pcdu] or [ioBoard] as parameter. Both

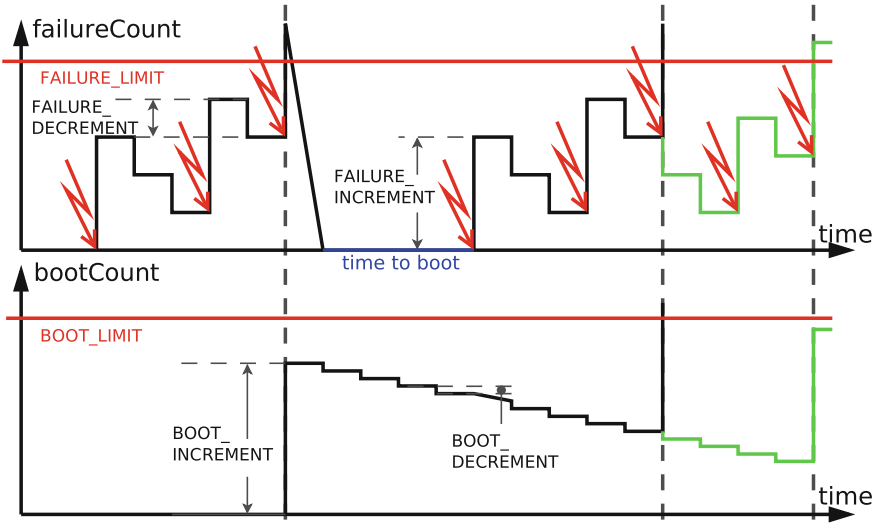


Fig. 10.29 Device Failure Handling—Extended Failure Confirmation Mechanism. © IRS, University of Stuttgart

cases trigger the same reaction that is the attempt to switch the I/O-Board as described in Sect. 10.2.3.

– If the faulty device is a standard device like an ACS sensor, a mechanism for shutting down devices by *FDIR* is chosen that is based on Fig. 10.27.

The approach followed is that two counters are defined to initiate different actions:

- Whenever a device reports a failure event, *failureCount* is increased by FAILURE_INCREMENT as previously introduced. This is also shown in Fig. 10.29.
- If *failureCount* exceeds FAILURE_LIMIT, the device is assumed to be faulty and cannot recover by itself. However, it worked for some time, so it is not necessarily deemed yet to be fully defective.
- Exceeding the first limit will initiate the system to deactivate the device's power switch in the PCDU and reactivate it in the next cycle, meaning to power-cycle the device and take it back into operation.
- Upon power-cycling the device, the device *bootCount* as second counter in turn is increased by BOOT_INCREMENT.
- *failureCount* is reset to '0' and the device is rebooted.
- Depending on the behavior of the device after the boot, *failureCount* is increased when the next failure is reported.
- bootCount in the meantime decreases by BOOT_DECREMENT in every FDIR step.
- The scenario shown in green color in Fig. 10.29 represents the following flow:

  - The device shows few failures after the reboot;
  - bootCount decreases significantly in that period;
  - After four failures, failureCount exceeds FAILURE_LIMIT;
  - bootCount is increased by BOOT_INCREMENT, but does not yet exceed BOOT_LIMIT;
  - The device is power-cycled and rebooted, but not marked faulty.
- The scenario shown in black color in Fig. 10.29 represents the following flow:
  - after three more failures in the device in a short period, FAILURE_LIMIT is exceeded again;
  - bootCount has not decreased significantly, at the same time it is increased by BOOT_INCREMENT and then exceeds BOOT_LIMIT;
  - the device is marked faulty and is shut down permanently. It will be marked as FAULTY (see Sect. 3.2.2) and will not be reactivated by OBSW.

The benefit of this mechanism is a quite freely possible adjustable number of failures that are tolerated for each devices in a certain period of time. These settings even can be adjusted individually per instance of a device type—considering the degradation of the equipment over mission lifetime.

10.4.3 Failure and Boot Counter Evaluation

Basis for the following values is the decision to tolerate at maximum one reboot of a device per 15 min = 900 s without marking it faulty. A device is rebooted (FAILURE_LIMIT is exceeded) after

- a maximum of 10 failures in 9 min if the device recovers in between the failures
- or 5 successive failures.

For devices for which the data is polled at 5 Hz (refer also to Sect. 3.4.3) this means within 15 min there are (900 s * 5 Hz) = 4500 times of data polls.

Concerning the effect of a reboot and marking the device FAULTY it has to be considered that the *FDIR* is scheduled with 10 Hz so within 900 s it is processed 9000 times. If *bootCount* is incremented to 9000 after the first boot, it will be reset to zero after 15 min.

The increase of *failureCount* is explained by Fig. 10.30 and with the following settings:

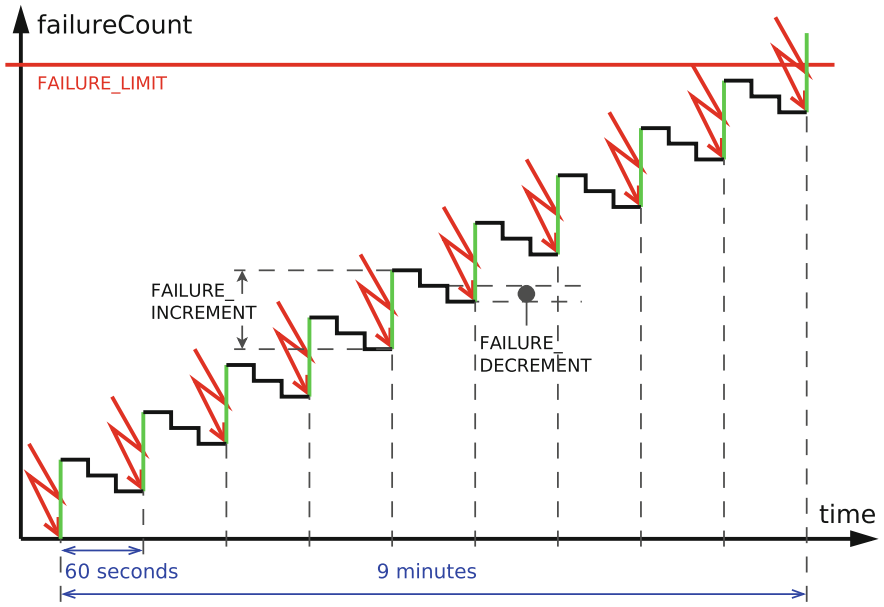


Fig. 10.30 Device Failure Evaluation Mechanism—Determination of the Constants. © IRS, University of Stuttgart

$$
\begin{gathered}
10 \cdot FAILURE\_INCREMENT - 9 \cdot 60 \cdot FAILURE\_DECREMENT \geq FAILURE\_LIMIT \\
5 \cdot FAILURE\_INCREMENT \geq FAILURE\_LIMIT \\
\rightarrow FAILURE\_LIMIT = 4600 \\
\rightarrow FAILURE\_INCREMENT = 1000 \\
\rightarrow 10 \cdot 1000 - 9 \cdot 60 \cdot FAILURE\_DECREMENT \geq 4600 \\
\rightarrow FAILURE\_DECREMENT \leq \frac{10000-4600}{9 \cdot 60 \cdot 10} = 1 \\
BOOT\_LIMIT = 9009 \\
BOOT\_INCREMENT = 9000 \\
BOOT\_DECREMENT = 1
\end{gathered}
\tag{10.3}
$$

With these settings, it is allowed for a device to provoke a failure every 500th response packet. For 5 Hz polling rate, this is 1 wrong packet per 100 s.

The values of these limits, increments and decrements can be configured at software compile time to adapt the device failure event handling mechanism to effects yielded by the system tests.

10.4.4 Device Handler FDIR Events

Event 10.36: DEVICE_REBOOT

	Event Name	Event Severity	Event Parameter
	DEVICE_REBOOT	MEDIUM	Device object-ID
Event description	This event will be generated if FDIR decides to reboot a faulty device		
Reaction	–		

Event 10.37: DEVICE_FAULTY

	Event Name	Event Severity	Event Parameter
	DEVICE_FAULTY	MEDIUM	Device object-ID
Event description	This event will be generated if FDIR decides to switch off a faulty device		
Reaction	–		

10.5 TTC FDIR

Possibilities for Failure Detection, Isolation and Recovery of the TTC system are very limited. Under nominal conditions both receivers need to be active in hot redundancy, otherwise a single-failure could lead to a loss of the mission. The same is true for the CCSDS-Boards of the OBC. As can be seen in Figs. 1.6 and 6.1, the CCSDS-Boards and the TTC receivers and transmitters (or transceivers) have a cross-coupled connection.

10.5.1 Receiver Failure Handling

There exists only one way for on-board FDIR to detect a malfunction of a receiver. If power consumption of one receiver rises over its maximum limit (see Table 17.10 in the annex) the fuse for this receiver will be triggered and the receiver will be switched off. Each receiver has its individual fuse as can be seen in Table 17.10.

Following the hot redundancy approach, receivers do not differ in their function. Both forward received TC packets to both CCSDS-Boards. The only difference might be a minimum output signal time offset due to electronic parts manufacturing

variation or due to degradation. This means that as long as one receiver is working properly, TCs will be forwarded to both CCSDS-Boards and will reach the OBSW, but the OBSW is not able to detect, which receiver submitted it to the OBC. TCs corrupted by a defective receiver obviously will be ignored by the OBSW. Except this, it is not possible to detect a malfunction in a receiver as long as the power consumption stays in its expected range and an unhealthy receiver fails passive.

A different way to detect failures in the receivers is available for the ground station and is explained for the complete TTC subsystem in Sect. 10.5.3.

A worst case scenario for a receiver malfunction is the one, where a defective receiver does not fail passive but still forwards data to the CCSDS-Boards. Should this receiver forward data faster than the healthy one, and with a correct start sequence at the beginning of the CLTU, with data being corrupted somewhere in the CLTU to the malfunction, these corrupted data will be forwarded to the CCSDS-Boards.

The CCSDS-Boards, upon detection of a start sequence (see Fig. 10.31) will ignore data sent by the second (healthy) receiver input channel. If the decoding CCSDS-Board detects an uncorrectable error in an extracted frame from the defective receiver during the unpacking process, it ignores the rest and discards the

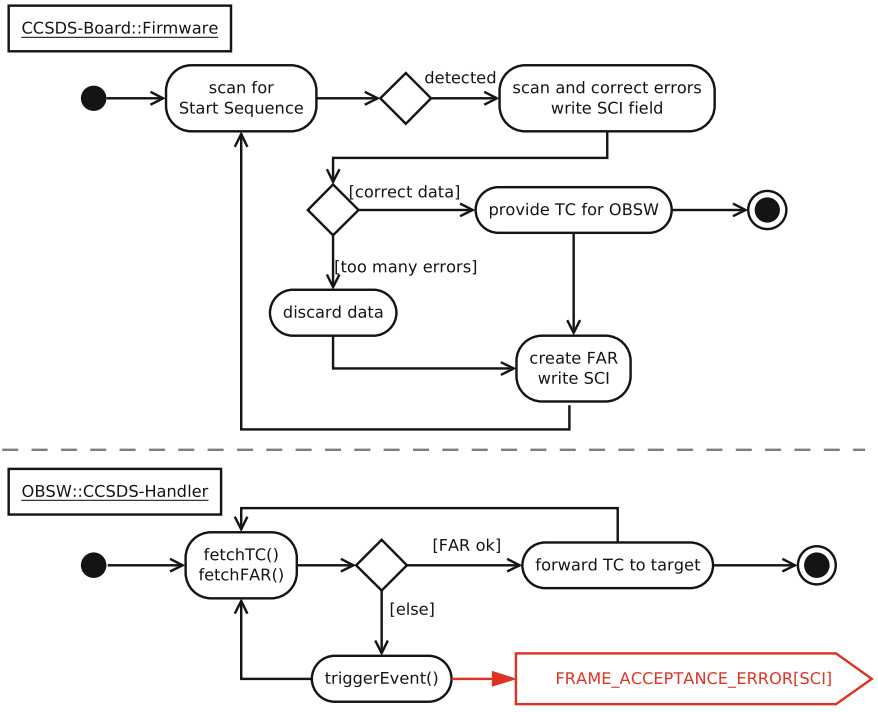


Fig. 10.31 CCSDS-Board input channel selection and OBSW Frame Acceptance Error. © IRS, University of Stuttgart

entire packet. After that, the CCSDS-Board starts scanning for a start sequence again. In that context a worst case scenario could mean, that an active failing receiver could prevent TCs from reaching the OBSW. A possibility to cover this problem is given in the Frame Acceptance Report (FAR), that is created by the CCSDS-Board automatically [55].

This report can be read by OBSW via SpaceWire RMAP access and yields a Selected Channel Input (SCI) field, which provides the channel on which the CCSDS-Board received and accepted data. This Event is forwarded to ground in a live Event TM packet and ground can identify the blocking receiver and can deactivate it via HPC1 command to the PCDU.

Orbit simulations can be performed quite accurate on ground. Therefore, it is always possible to predict a ground contact phase very precisely, and these ground contact time slots should preferably be uploaded to the OBSW cyclically for the next several ground contact slots—e.g. always for the next 24 h ahead (also refer to the mission planning topic later in Chap. 14). Knowing when ground interaction should happen, the system can determine that apparently TCs have not reached the satellite and can issue an according Event TM to ground. However, this should not happen as long as so far both receivers have been working properly. If one receiver fails before or during a contact phase ground should again use the Frame Acceptance Report (FAR) to determine which receiver apparently is not performing properly—see Fig. 10.32.

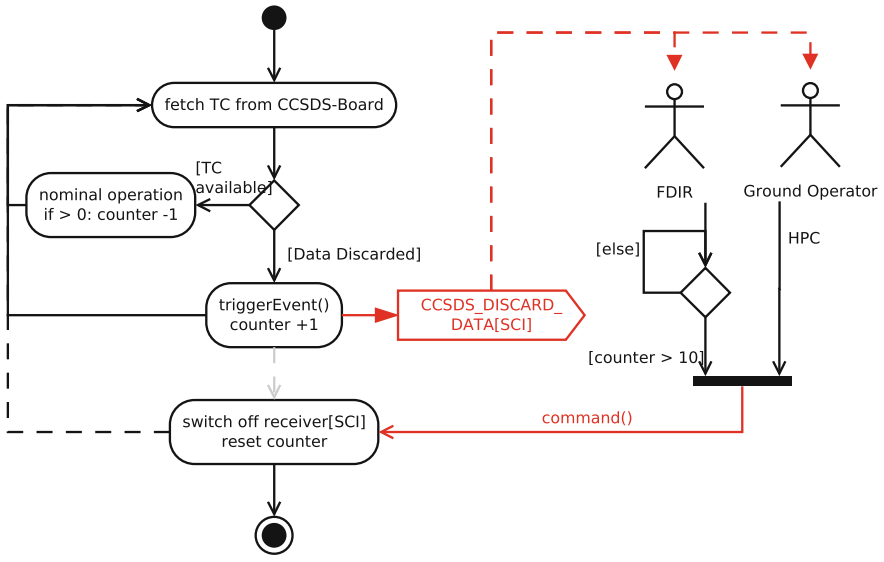


Fig. 10.32 Unhealthy receiver switch-off by FDIR or Ground. © IRS, University of Stuttgart

In case where even the second receiver fails, FDIR will realize a TC timeout after a certain period of (current limit 120 s) where contact was expected to have been established. This will lead to the currently active receiver being marked unhealthy, which will be switched off. Since the system will always try to achieve a state where it might be reached from ground, FDIR will automatically switch on the currently deactivated receiver—one receiver always must be active.

By doing this, there is a final chance that the receiver previously marked unhealthy has recovered its health state. If that is not the case, FDIR will toggle between the receivers during all future contact phases as long as power is available. If contact cannot be established anymore, the mission is lost but this case goes far beyond the required single failure robustness paradigm.

10.5.2 Transmitter Failure Handling

The two transmitters should not be powered at the same time since their signals are routed to two shared antennas and both transmitters sending will lead to a corrupted signal that cannot be interpreted on ground. Additionally for legal reasons a transmitter may only be powered when in range of a ground station. Therefore, the system may only power up a transmitter when the satellite approaches the ground station.

During nominal operation, this is performed via time-tagged TCs in the OBSW, that power the transmitter. As as second measure, the transmitters are powered by the TTC *COMM_CONTROLLER* object (Fig. 6.3) upon detection of a carrier lock due to RF-signal from ground station. The carrier lock is set when the receiver recognizes a signal on the TC carrier frequency. However, it cannot be determined on board the satellite whether the transmitter properly sends TM or not. This is for the ground to decide when they expect to receive TM.

In case of TM reception problems on ground the initial recovery attempt for the transmitter would be the deactivation of the nominal transmitter by direct command. The *TTC_Subsystem* thereupon will automatically activate the redundant transmitter (see also Sect. 6.3) and TM will be sent again.

Similar as for the receivers, one way of detecting a failure in the transmitter is to observe the power consumption and to deactivate a transmitter when it exceeds the limit provided in Table 17.10 in the annex.

10.5.3 Transceiver Health Monitoring

The TTC subsystem cyclically reports certain TTC equipment housekeeping parameters as telemetry to ground. The very specific parameters however are subject to analysis by ground personnel. Please also refer to the explanations to

Tables 6.13 and 6.14 as well as to the reference health data (BOL values) provided by the supplier in annex Sect. 17.5.2

10.5.4 CCSDS-Board FDIR

A possibility to determine a malfunction in a CCSDS-Board is provided through interaction with the ground station. TCs are sent with a dedicated Virtual Channel identification (VCID) in the packet. The receivers both forward all TCs, but the CCSDS-Boards filter for their VCID. A CCSDS-Board only forwards TCs with the correct corresponding VCID. A ground operator by this means can identify if TCs are not acknowledged by the spacecraft.

The CCSDS-Boards share a cross-coupled link between each other (see Fig. 1.6). On this link, Command Link Control Words (CLCW) are exchanged between the CCSDS-Boards. CLCWs provide information about TC decoding in the CCSDS-Boards. The CLCWs are inserted into all TM Transfer Frames of each board and are sent to the ground for further evaluation. For each uplinked TC Frame these exist according CLCWs. Table 10.11 shows the content of a CLCW. Most important for failure monitoring by ground is the third byte providing the status of carrier and bit lock.

The Control Word Type furthermore contains information whether the content is submitted from the board targeted to by the selected VC in the uplinked spacecraft TCs—or whether it is included in the second board's TM idle frames. In the latter case the nominal first board has a transmission problem for CLCWs (and subsequently for all TM). More details about the failure monitoring with CLCWs can be taken from [55]. The CCSDS-231.0-B-2 standard [27] also covers the use of the TC frame counter when the telecommunication is run in Acceptance Check and Data Mode (AD).

Table 10.11 Content of a Command Link Control Word (CLCW)

Byte Number	**CLCW Register bits**	**CLCW contents**						
First	[31:24]	Control Word Type	CLCW Version Number	Status Field	COP In Effect			
Second	[23:16]	Virtual Channel ID	Reserved Field					
Third	[15:8]	No RF Available	No Bit Lock	Lock Out	Wait	Retransmit	Farm B Counter	Report Type
Fourth	[7:0]	Report Value						
Fifth	N/A	[RS232 Break Command]						

10.5.5 Communication System Events

This section collects events that are involved in the communication system FDIR.

Event 10.38: FRAME_ACCEPTANCE_ERROR

	Event Name	Event Severity	Event Parameter
	FRAME_ACCEPTANCE_ERROR	LOW	CCSDS-Board object-ID, Selected Channel Input SCI
Event description	This event will be generated if a CCSDS Board identifies a corrupted input TC frame and drops the discards the data		
Reaction	Event will be routed to ground immediately since it can only occur upon ground contact. Ground operator needs to decide whether VC switch is required		

Event 10.39: CCSDS_DISCARD_DATA

	Event Name	Event Severity	Event Parameter
	CCSDS_DISCARDED_DATA	LOW	CCSDS-Board object-ID, Selected Channel Input SCI
Event description	This event will be generated if a CCSDS-Board reports that it discarded data		
Reaction	Event will be routed to ground immediately since it can only occur upon ground contact. Ground operator needs to decide whether VC switch is required		

10.6 ACS FDIR

This chapter describes implemented failure detection methods for the ACS devices. Later in the chapter furthermore the ACS mode transitions are explained, which are initiated by the *ACS_Controller* as reaction to detected unit malfunctions.

The ACS hardware of the FLP Generation 1[1] consists of the sensors MGM, FOG, SUS, GPS and STR. Except for the SUS, these sensors deliver measurements that are processed directly by the OBC. Failure detection requires permanent observation of equipment on-board telemetry. As illustrated in Sect. 3.2.4 and Fig. 3.9 the monitoring of these measurements is encapsulated in the subsystem controller object—in this case the *ACS_Controller*—as described in Sect. 7.11. For

[1]For the FLP Generation 2 the ACS can be upgraded to a full AOCS with a propulsion module (see Sect. 7.10) which however is not treated here in the FDIR chapter as no such spacecraft/mission is implemented yet at time of publishing.

the sensors, a sensor signal fusion and comparison of their redundant units provides the basis for failure detection. For the actuators—Reaction Wheels (RW) and Magnetotorquers (MGT)—the redundancy management in FDIR cases is more complex and is described later in this chapter.

10.6.1 Magnetometer Failure Detection

Two magnetometers are available in hot redundancy on board the FLP platform. Their single magnetic field values are monitored and undergo a limit check as a first step in the *ACS_Controller*. The monitor limits are set in the OBSW to a design value of 66 µT for one axis, which is the maximum expected magnetic field intensity around Earth (see [141], Chap. 7).

A limit violation only can occurs if induced by a magnet on ground during tests, but not in flight. Therefore such a limit violation in orbit indicates a failure in the device. This can only be determined during periods in which the magnetotorquers are not powered. Otherwise the MGM measurements will be biased by the MGTs' magnetic field. This implies that the monitoring limit check in the OBSW has to be activated and deactivated depending on the MGT activation status.

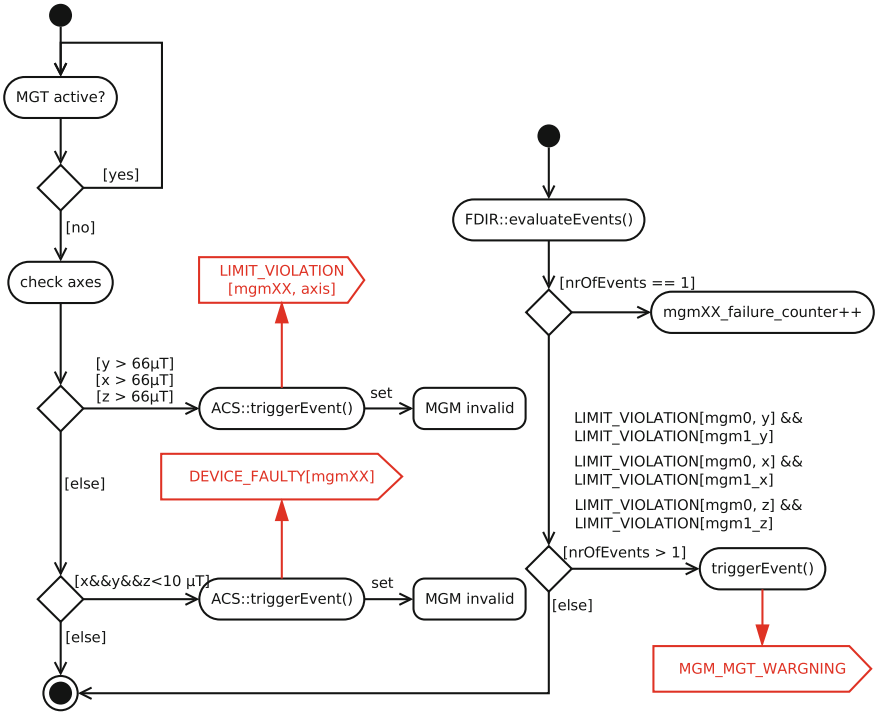


Fig. 10.33 MGM failure detection. © IRS, University of Stuttgart

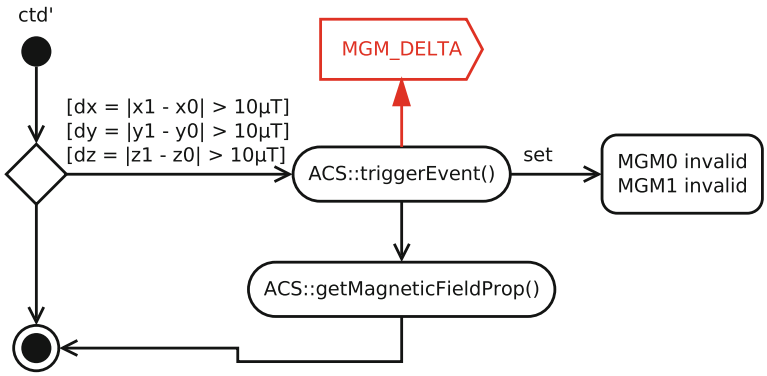


Fig. 10.34 MGM cross monitoring and comparison to propagation. © IRS, University of Stuttgart

If both MGM sensors report a limit violation for the same axis, this indicates that it is caused by an MGT coil still being active and the alternating operations timing between MGT and MGM being incorrect.

According to information from the MGM manufacturer it is furthermore not reasonable that all axes show a very small magnetic field value (<10 µT) or near-zero at the same time and the absence of value changes in one axis (not displayed in the following image) also indicates an MGM internal failure.

The process of failure detection for the MGMs is depicted in Fig. 10.33 and is performed during every measurement cycle of the ACS.

As a second step, the two MGM sensors can be used to detect a failure in the measurements by comparison of the single magnetic field value along each axis of the two devices. But this comparison is not sufficient to isolate the failure, since it cannot be recognized which of the two devices is mistaken.

In case where such a measurement signal incompatibility occurs, an according Event is reported to ground and will need further investigations there. To back-up this case, a propagator for Earth's magnetic field is implemented in the ACS algorithms to provide the magnetic field vector if no MGM is available for some time—see Fig. 10.34.

10.6.2 GPS Sensor Failure Detection

As explained in Sect. 7.5 the FLP Generation 1 is equipped with three DLR GPS receivers that are running in hot redundancy since they are designed as non radiation hard devices. From communication with the manufacturer it is also known that in radiation induced error cases (SEUs) the GPS receivers often show higher power consumption and corrupted communication (not only corrupted navigation data). These symptoms are already covered with the mechanisms in Sects. 10.4 and 10.3.

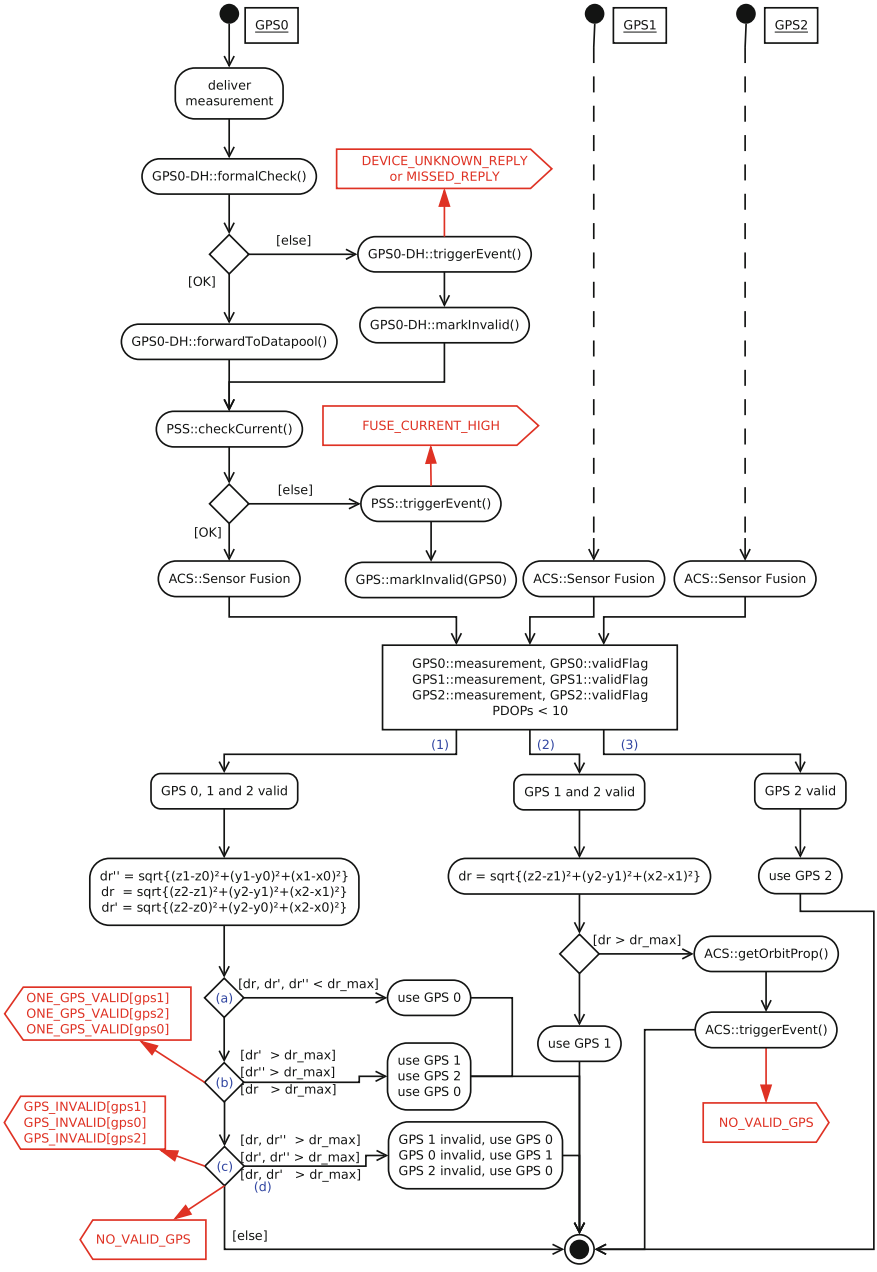


Fig. 10.35 GPS Sensor Monitoring. © IRS, University of Stuttgart

In case where no other instance reports a failure and marks the sensor measurements as invalid, the data from the GPS receivers are compared against each other.

Having three receivers available a 2-to-1 sensor voting can isolate a malfunctioning device. If voting identifies one instance as malfunctioning, *FDIR* will restart the device and will check whether it thereafter is operating properly again. If not, *FDIR* will power off the device permanently. The number and logic of retry cycles is explained further below in this section. Corresponding Event TM allows ground to follow the actions taken on board. The voting process is illustrated in Fig. 10.35. It is deployed in detail for GPS 0, but the same logic implies for GPS 1 and GPS 2:

- GPS 0 delivers its measurement and the *deviceHandler* receives the data.
- The *deviceHandler* performs the formal check for validity, protocol and/or CRC as already described in detail before in Sects. 3.1 and 10.4.
- If the formal check proves NOT to be successful, the *deviceHandler* triggers an FDIR Event—either that it did not receive a response (MISSED_REPLY) or that the response is not correct (DEVICE_UNKNOWN_REPLY).
- If the response is ok, the GPS sensor data is written into the *datapool* and is made accessible for further operations.
- A second instance in device monitoring is the *PSS_Controller*. It observes the power consumption of the devices and signals if a device exceed the permissible electric power consumption. The corresponding Event is FUSE_CURRENT_HIGH.
- Measurements and *validFlags* are provided to the *ACS_Controller* which performs sensor redundancy management as described in Fig. 3.9.
- Depending on the number of valid GPS measurements, the ACS computes the satellite position distances between the individual GPS measurements.
- Branch (1)—all GPS sensors are valid up to this point:
  Three distances are calculated for each sensor pair. The cases are explained below with the nominal case being (a). The following failure cases (b), (c) and (d) are illustrated in Fig. 10.36.

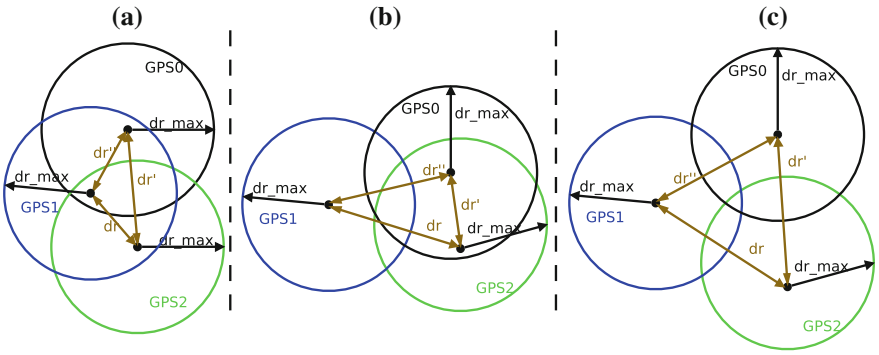


Fig. 10.36 GPS monitoring—permitted deviation between the measurements. © IRS, University of Stuttgart

(a) All measured positions are within the maximum distance and are valid. In case all measurements are deemed correct, the first instance is selected by default.
(b) One of the distances (dr′) is greater than the maximum permissible deviation. This means that the middle one is assumed to be the correct one. The valid one is reported to *FDIR* (GPS_VALID[gps1]) which then knows that the two others are faulty.
(a) Two measurements (dr, dr″) exceed the maximum permissible deviation. It is assumed that the instance involved into the calculation of these two is the faulty one (here GPS1). By default the first remaining valid instance is selected (here GPS0). If that is identified as faulty for other reasons (e.g. power consumption) the second instance is selected. The faulty instance is reported with GPS_INVALID[gps1].
(c) If all measurements are above the limit, the controller signals all GPS sensors to be faulty NO_VALID_GPS.

- Branch (2)—One GPS sensor invalid, two sensors valid:
  - The distance between the sensor measurements is calculated and checked against an upper limit. If it is above the limit, none of the measurements is selected and both sensors are reported to be faulty with NO_VALID_GPS.
  - Otherwise the first or second instance is selected by default.
- Branch (3)—One sensor valid:
  - If only one sensor is deemed valid at this point, and has passed the previous checks, it is assumed to be valid. This is an unsafe case but with three sensors, this should not occur regularly or for longer periods. Through the Events and GPS TM the ground has full visibility and can examine this further if necessary.

NO_VALID_GPS: For the case where there is no valid GPS measurement available, the ACS algorithms include an orbit propagator for the computation of the S/C position. The propagator takes over immediately if no valid GPS position information is available. In the first hour the propagator (using linear extrapolation) has an accuracy of <0.1° [125]. If after a while there is still no GPS information the ACS subsystem and as result the entire satellite system will be switched into Safe Mode. The mechanism is the same as introduced in Sect. 10.4.3 with the settings:

$$\begin{gathered} FAILURE\_LIMIT = 4600 \\ FAILURE\_INCREMENT = 1000 \\ FAILURE\_DECREMENT = 1 \\ BOOT\_LIMIT = 30000 \\ BOOT\_INCREMENT = 10000 \\ BOOT\_DECREMENT = 1 \end{gathered} \tag{10.4}$$

After 5 successive reports of a GPS receiver being invalid, or 10 times within 9 min, the receiver is rebooted. If the receiver needs to be rebooted 5 times within

F98 – Command Response

MsgID	Chars.	Format	Description
F98	var		Command Response
	1	x	<STX>
	3	xxx	Message Id (=F98)
	4	xxxx	GPS week
	8	xxxxxx.x	GPS seconds of week [s] (at output)
	1	x	Dash ("-")
	1	x	Message type (I=Info, E=Error)
	1	x	Dash ("-")
	2	xx	Command ID
	1	x	Dash ("-")
	var		Text
	1	x	Dash ("-")
	2	xx	Checksum
	1	x	<ETX>

Fig. 10.37 GPS command response—from [63]. © IRS, University of Stuttgart

30 min (= 1800 s ↔ 18000 FDIR cycles ↔ max. 18000 times BOOT_DECREMENT), it is deactivated and marked as permanent faulty. With this mechanism, phases with high chances for radiation effects such as a passing of the South Atlantic Anomaly can be compensated.

$$5 \cdot BOOT\_INCREMENT - 18,000 \cdot BOOT\_DECREMENT! > BOOT\_LIMIT$$
$$5 \cdot 10000 - 18000 > 30000 \tag{10.5}$$

If all receivers show the same behavior, they are assumed to be damaged and are marked faulty after about 30 min and the *ACS_Assembly* object will report that it cannot keep the mode which results in *ACS_Subsystem* not being able to keep its mode. The spacecraft will automatically initiate a Safe Mode transition in such case.

For further investigation the GPS provides an info response as depicted in Fig. 10.37. This message can be requested by OBSW from the GPS receivers or by ground via PUS Service 2 and comprises details about the health state of the GPS receiver. It also reports errors that are marked with an 'E' in its message Char 18 and more specifying text in Chars 23 ff until end-Char minus 4. The content of this descriptive text message is not handled within the OBSW FDIR but must be processed on ground.

10.6.3 Star Tracker System Failure Detection

The star tracker system is used for fine attitude measurement. Two STR Camera Head Units (CHU) are arranged to point in different directions to avoid both CHUs being blinded by the same light source. The CHUs deliver individual measurements —illustrated in Table 10.12—that are combined in the *ACS_Controller*.

Table 10.12 Star tracker Attitude Telemetry Packet—from [61]

AscAttitudeTM						
Function Type: TM(5,1)						
Header	Report ID	Attitude	Info	COI Time	Checksum	
16 bytes	2 bytes	16 bytes	8 bytes	6 bytes	2 bytes	
Info						
Flags	Residual	Locks	Objects	Stars/Fail	ACG Floor	ACG Ceil.
8 bits	1 byte	1 byte	1 byte	1 byte	12 bits	12 bits
Flags						
Sequence	Correction	CameraID	Highrate	BBO	Time Ref.	Valid
1 bit	1 bit	2 bits	1 bit	1 bit	1 bit	1 bit

Table 10.13 Star tracker Event Report Telemetry—from [61]

AscEventReportTM				
Function Type: TM(5,1), TM(5,2), TM(5,3), TM(5,4)				
Header	Event_ID	Parameter 1	Parameter 2	Checksum
16 bytes	2 bytes	4 bytes	4 bytes	2 bytes

Table 10.14 Star tracker Housekeeping Telemetry—from [61]

AscHKTM									
Function Type: TM(3,25)									
Header	Structure ID	Temp CPU	Temp SMPS	HK CHU-A	HK CHU-B	HK CHU-C	HK CHU-D	Mode	Checksum
16 bytes	1 byte	1 byte	1 byte	2 bytes	2 bytes	2 bytes	2 bytes	1 byte	2 bytes

Inside the star tracker electronics diverse internal health and data plausibility tests are performed by the internal microcontroller which delivers a precise information for many failure cases. The protocol of the attitude TM delivers the data listed in Table 10.12. The important content is the "Valid" bit in the packet which provides information about the correctness of the delivered attitude as far as identified by the STR electronics. This packet is sent to the OBSW periodically at a rate of 5 Hz. If the attitude is considered invalid by the STR electronics, it will nevertheless be *submitted to the I/O-Board*, but the *datapool validFlag* for the attitude will be set to invalid by the OBSW.

If an abnormal Event in the STR system is detected by the microcontroller, an Event telemetry packet is generated and is sent to the OBSW. However, the report will be used by ground for investigations and not inside OBSW FDIR. A detailed "Status Report TM" can be requested from the STR system for further analysis—

see Table 10.13. This report contains information about failed communication and debug Events in the STR software. The request and the analysis needs to be performed by a ground operator.

Every second request to the STR (also 5 Hz) is reserved for a "HK TM packet" as shown in Table 10.14. This packet contains the measurements of the internal temperature sensors needed by the Thermal Control System. These are the temperature values of the STR CPU, of the power supply and of the CCD sensors of each CHU. It also provides the operational mode, which by default is "Application Mode" [61].

All telemetry packets contain a checksum, that is verified by the STR *deviceHandler* in every communication cycle. If the CRC check proves to be not successful, the STR is marked faulty as introduced in Sect. 10.4.

10.6.4 Sun Sensor Failure Detection

In the FLP Generation 1 analog sun sensors—simple solar cells—are used for determination of the sun direction. They provide an electric current depending on the intensity of the sun's radiation meeting the cells. Pairs of two cells are placed at 8 different locations on the spacecraft yielding a total of 16 sensors. An analog/digital converter (ADC) within the PCDU converts the analog signal of the cells into a digital signal that is provided within the PCDU HK data packet to the OBC/OBSW.

Figure 10.38 shows the principle of the sun sensor data transmission and failure detection:

- Two paired sensors SUSX1 and SUSX2 (8 pairs, X denotes the pair ID—see Fig. 7.4) provide their measurements to their common *deviceHandler* after conversion in the PCDU.

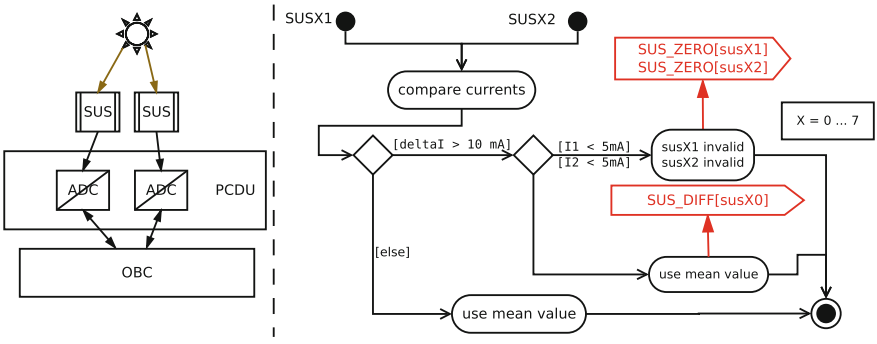


Fig. 10.38 Sun sensor monitoring in ACS. © IRS, University of Stuttgart

- A limit check is not deemed reasonable for two solar cells. Their output ranges between a minimum of 0 and, according to the specification, a maximum of 75 mA.
- A comparison between the measurements of both sensor of a pair is performed. It is assumed that deviation through degradation is small as the cells are of the same type. If the comparison shows a difference of more than 10 mA, the sensors are checked for their absolute value.
- If one sensor shows a significantly low output value (around 0) it is assumed to be invalid due to damage. A SUS_ZERO warning Event is generated and the sensor is marked as invalid for further processing.
- If only a difference of more than 10 mA can be detected, the faulty unit cannot be isolated. If the illumination intensity is high on that side of the S/C, the sensors are important for the correct determination of the sun direction. Therefore in such case it is deemed better to use a mean value of the two sensors than no value at all. SUS_DIFF Event signals the deviation between the two sensors for further analysis by ground.

Methods for the filtering of noise on the sensors, compensation of albedo, and compensation of loss of one sensor pair have been developed for the first FLP satellite "Flying Laptop" in [125] but these features are not yet integrated into the OBSW release used for launch. It is one of the optional features for later in-flight improvements.

10.6.5 Fiber Optic Gyro Failure Detection

Four Fiberoptic Gyroscopes (FOG) are arranged in a tetrahedron configuration (see Sect. 7.7) to allow for a 3-out-of-4 redundancy in case where one sensor is defective. The FOGs are monitored by OBSW in the same way as all other devices, observing power consumption and performing communication checks. Additionally, the redundancy allows for a measurement comparison between one FOG and the three others.

For a working FOG set the sensor failure detection can be performed using the following approach[2] as displayed in Fig. 10.39:

- The single measurements can be compared to hard limits which have been set from a maximum rotational rate in the operation, or by the manufacturer.
- The rotational vector of the spacecraft derived from the measurements of the FOG sensors can be obtained from four individual calculations. Three FOG sensors are required for the calculation of the vector, so there are four different triplets to calculate it, with four different results.
- The four different rotational vectors can be compared pairwise.

[2]For the detailed explanation of the method please refer to [136].

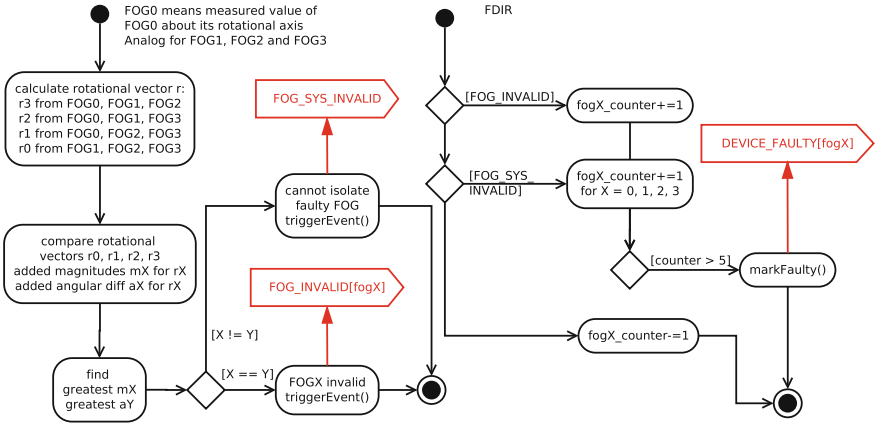


Fig. 10.39 Mechanism for FOG Monitoring. © IRS, University of Stuttgart

- The corrupted measurement disturbs three of the four rotational vectors and corrupts their direction and magnitude away from the actual result.
- So, the vector with the greatest deviation from all three others in magnitude and direction is assumed to be the correct one.
- The FOG that contributed to the calculation of all other three is the faulty one.

Furthermore the rotational rates can also be determined from the star tracker measurements. The rates computed from star tracker quaternion changes over time provide a comparable precision as the values from the FOG system.

In [136] it was identified that the FOGs of the first FLP spacecraft "Flying Laptop" show a high degree of degradation, probably due to the long period of storage.[3] This leads to an increased noise level for the sensors which in turn makes distinction between incorrect measurements and noise influence difficult. Figure 10.40 depicts the noise level of the different FOGs of the "Flying Laptop". FOG2 shows the worst result. Assuming a rotational rate in Idle Mode of maximum 1°/s which corresponds to 0.0174 rad/s, then the noise and actual rotational rate are of the same magnitude for FOG2. Even FOG3 (orange) and FOG0 (gray) statistically provide a noise level that is close to 0.01 rad/s.

For this reason in the particular case of the "Flying Laptop" mission the rotational rates are deduced from the STR system.

Note: Because of the low importance of the FOGs on the "Flying Laptop" the system in this mission will not transit to Safe Mode if the FOG system is not available anymore.

It could be considered to exclude FOG2 right from the start of the "Flying Laptop" mission and still have sufficient devices to determine the rotational rate of

[3]The FOGs were procured in a far too early phase of the satellite project and were stored several years until integration and environmental tests were completed.

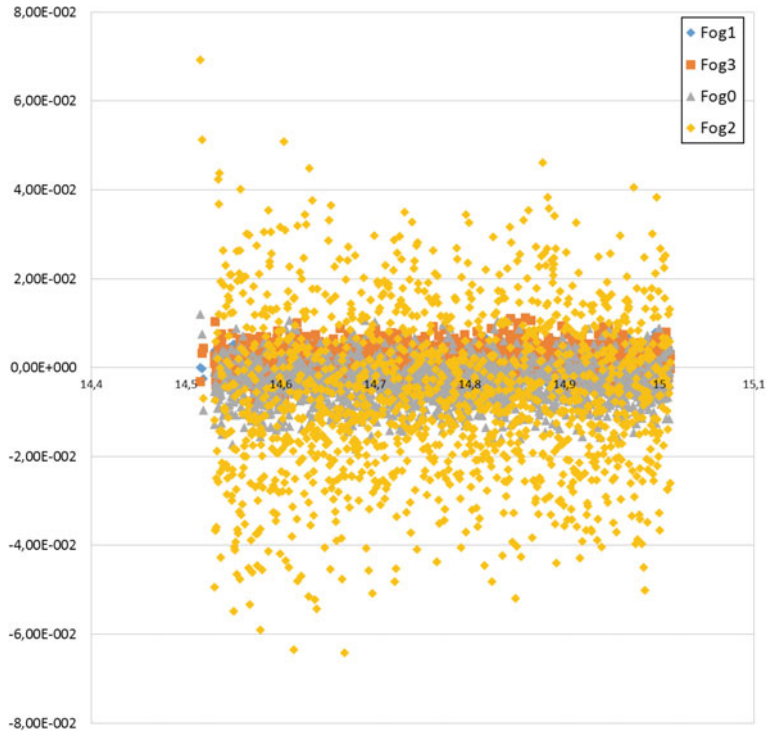


Fig. 10.40 FOG noise observation for the FLP-based "Flying Laptop". © IRS, University of Stuttgart

the satellite. However, a missing redundancy leads to FDIR not being able to detect when one of the remaining three FOGs provides a corrupted measurement. This is only applicable when the rotational rate cannot be determined properly anymore.

For the FOGs no additional failure detection mechanism is foreseen. Tests during the Commissioning Phase of the "Flying Laptop" mission comparing the rotational rates from STR and FOGs might provide better results and opportunities for failure checks.

10.6.6 Magnetotorquer Failure Detection

There are two cold redundant magnetotorquer systems on board an FLP spacecraft. Each torquer is arranged along one of the three major axes x, y and z of the satellite (see Fig. 7.13). One MGM/MGT cycle is 5, 0.5 s of which the MGM performs measurements. Hence, the MGT can be activated for max. 4.5 s in every cycle, depending on the request from ACS control. The MGT Electronics (Fig. 7.14) unit detects and provides information to the OBC/OBSW at 2 Hz about each coil's

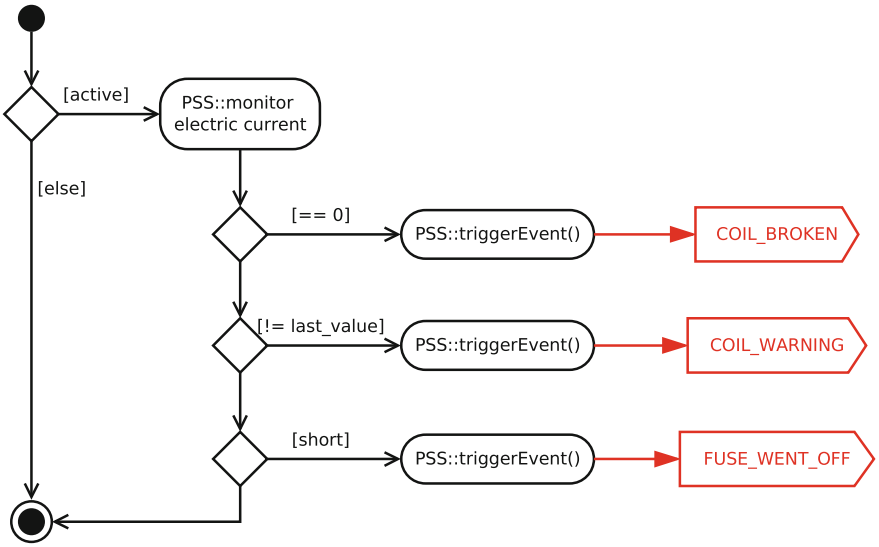


Fig. 10.41 MGT monitoring. © IRS, University of Stuttgart

status whether the coils are supplied with electric current (1 = current in positive direction, 0 = no current, −1 negative direction). For failure detection only the power consumption in relation to the torquer switch status can be monitored.

Figure 10.41 shows the monitoring logic for an MGT coil. The coils are accessed and used independently, so the mechanism is analogue for all three. Unlike for the ACS sensor devices, there is no complex failure detection in the *ACS_Controller* for the MGTs. The illustrated process is implemented in the *PSS_Controller* and is the following:

- If the MGT is commanded 'active' from the *ACS_Controller* the MGT Electronics applies electric current through the coil in positive or negative direction. There is no current control. The magnetic torque is controlled with the length of the activation pulse.
- If the *PSS_Controller* detects no electric current on the fuse, the Event COIL_BROKEN is signaled and shows that the coil might be broken. *FDIR* will mark the coil as invalid so that the redundant coil will be used from that point onwards.
- If electric current is detected, a comparison to the latest measured value is performed. If the current shows a significant difference, the Event COIL_WARNING is triggered. The Event indicates that something might be faulty with the coil but no actions are undertaken on board in this case. The Event just serves as warning for the ground operator.
- If a short circuit is produced in a coil, the fuse in the PCDU is automatically triggered as described in Sect. 10.2.8. The *PSS_Controller* reports the change of

the fuse state with the event FUSE_WENT_OFF. *FDIR* will mark the according MGT unit as unhealthy.

10.6.7 Reaction Wheel Failure Detection

The FLP design includes four Reaction Wheels (RW). In analogy to the FOGs the RWs are arranged in a tetrahedron configuration to allow for a 3-out-of-4 redundancy. The RWs are expected to be most critical devices w.r.t. failure due to wearout of motor or bearings. The friction in the ball-bearings is expected to increase over the RW lifetime. One way of identifying the degradation of the RWs is the observation of their power consumption at similar speed values over the spacecraft mission lifetime. This will be performed by ground on basis of standard housekeeping TM and is not implemented in OBSW FDIR.

A second mechanism to recognize the degradation of the wheels is to use the integrated accelerometer in the STR electronics to observe the vibrational load created by a wheel at a certain speed and how the load increases over the spacecraft mission lifetime. This is an approach that can also be realized by ground on basis of standard housekeeping TM and is not implemented in OBSW FDIR.

The RW built-in software provides the following internal safety features [68]:

- Built-In checksum calculation to avoid faulty operation due to bit-flips. (SEUs). The software is automatically reloaded inside the RW from its PROM to its controller if a watchdog is not reset for 2 s.
- A speed limiter checks every speed command. If the commanded speed exceeds the limit, this speed command is overwritten with the maximum value of 3000 rpm.
- In case where the RW control software is stuck, and the wheels are approaching the maximum speed value, this creates an electromagnetic field of approximately 20 V inside the device which the drive motor cannot overcome. This physically prevents the wheel from damaging itself by oversaturation.

Upon request the RW sends a TM package to the OBSW that contains a "Command Status" bit that can have the following values:

(0) Successful: A command has been executed successfully.
(1) Invalid Address: Each command contains a target address of either of the four RWs. If this is not correct, the command is ignored. An invalid address could be caused by an OBSW software bug and is not expected to happen during operations due to thorough preflight testing. If it occurs during flight, ground needs to investigate the problem. Since the wheels on FLP Generation 1 are not operated on a data-bus, all wheels have the same implemented address.
(2) Invalid Opcode: The command to a RW contains an "Opcode" for mode commands. If "2" is returned in the RW HK TM a wrong command was sent to the RW. This could either be causes by a bug in the OBSW—which is not

expected during operation since all commands are tested up-front—or a manual command from the ground operator contains the wrong Opcode.

(3) Bad Checksum: An SEU might cause corrupted data. The command will not be executed. If this happens during nominal operations it will not have any serious consequences. The next command is expected to be executed again. If this happens several times per minute ground investigation need to be performed.

(4) Message Format Error: This could be caused by an SE. It signals a software bug in the OBSW or in a command submitted by ground (in analogy to 2). The next command is expected to be executed. If this happens several times within one minute, ground investigation is required.

(5) Incomplete Message Timeout: This indicates that a command to the RW has not been transmitted completely. This could happen because of external disturbance or due to a fault in the Core DHS subsystem. External disturbances are not expected to cause major damage and will be ignored if it does not happen more than once per minute. If the Core DHS subsystem is faulty other mechanisms will take according recovery actions as described in Sect. 10.2.

(6) Buffer Overflow: A memory area in the RW cannot provide more space. This is not expected during nominal operation since RW HK is polled by the OBSW with moderate 5 Hz. If "Buffer Overflow" is reported ground investigation is required.

(7) Wheel Saturation: If one RW reaches its maximum permitted speed of 3000 rpm this status is reported. It is implemented into the *ACS_Controller* to desaturate the wheels at a total added speed rate of all wheels together being 6000 rpm [125]. If this mechanism fails RW_SATURATION_REACHED will be signaled by the reaction wheel *deviceHandler* (see Fig. 10.42). This Event is not expected to happen during operation since it would be caused by a

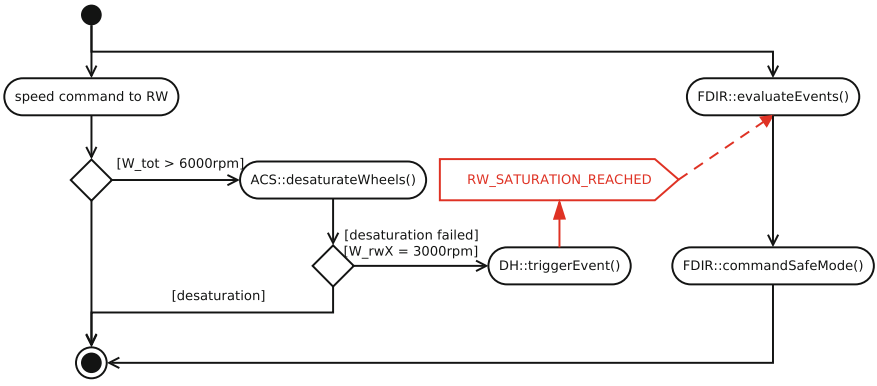


Fig. 10.42 RW saturation monitoring. © IRS, University of Stuttgart

malicious bug in the control system. If it occurs an immediate switch to Safe Mode will be initiated by *FDIR*. This leads to powering down the RWs and to perform spacecraft actuation with the MGTs. When the *RW_assembly* is commanded to power down the RWs it sends a command to the wheels to do a breaking maneuver and reduce the speed actively down to zero before triggering the switch-off in the PCDU.

10.6.8 Attitude Control System Failure Detection

In the previous paragraphs the FDIR for ACS equipment has been treated. In [125] basic failure detection mechanisms for the attitude control have been specified—in relation to the ACS system. The concepts are explained hereafter and the FLP implementation is presented. The failure detection mechanisms are based on monitoring of different ACS effects:

(1) Depending on the ACS mode, the system monitors the rotational rate of the satellite in every computation step—i.e. with 5 Hz. If the rotational rate of the satellite exceeds the mode-dependent monitoring limit (Pointing Mode = 1.2°/s respectively for Idle Mode = 3°/s), the event RATE_LIMIT_VIOLATION is triggered and FDIR will initiate a switch into the next lower mode, which is Idle Mode or Safe Mode (see Fig. 2.2). If commanded into Safe Mode, ACS will first re-perform detumbling due to the high rotational rate.

(2) The second step is somewhat specific for the FLP mission "Flying Laptop" but shall serve as representative case for optical Earth observation spacecraft based on the FLP:
In this step sun avoidance for the payload cameras is mandatory if the cameras are active. The control process is illustrated in Fig. 10.43. ACS examines the angle between the cameras—oriented in Nadir direction—and the sun direction. If the angle decreases below 60°, the Event SUN_PROXIMITY is triggered. This Event is handled as an info Event and does not trigger FDIR reactions. However, if the angle decreases below 45° ACS will trigger the Event SUN_AVOIDANCE. In that case *FDIR* switches off the cameras immediately. Since this is de facto a cancellation of the operational observation scenario *FDIR* will directly initiate a switch to Idle Mode where the payload cameras will automatically be directed away from the sun.

(3) The third step of the failure detection observes the accuracy of the pointing maneuver. The process is shown in Fig. 10.44. Therefore, the *target* is set with the target info. *target* is the attitude, the satellite is supposed to be in. If the

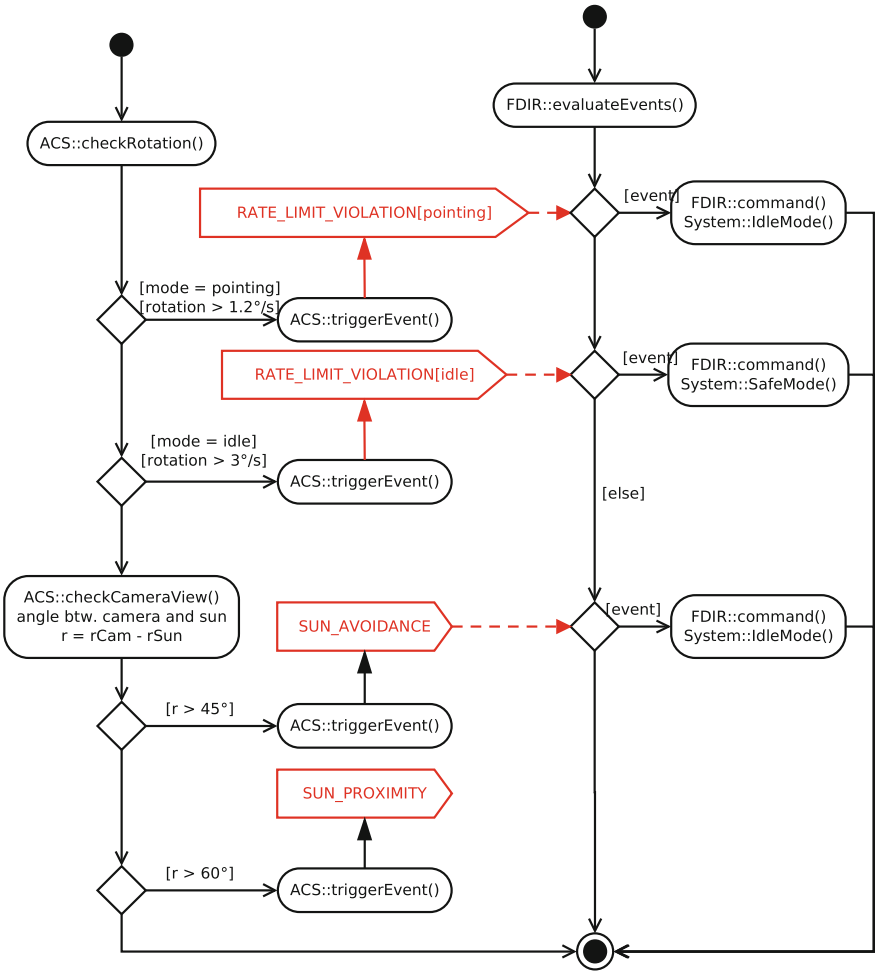


Fig. 10.43 ACS failure detection—Rotational rate and Sun avoidance. © IRS, University of Stuttgart

difference between *target* and the attitude of the satellite is greater than the maximum allowed deviation of 150 arcsec for more than 30 s, the target needs to be acquired again. As initiation, the ACS sets an *acquisitionFlag* to start the acquisition maneuver. If the target is not acquired after another 30 s, the target acquisition maneuver is assumed to be not accurately enough, or in other words—not successful, and the Event OFF_TARGET is generated by the *ACS_Controller*.

Failed acquisition is determined as when the attitude during the target acquisition maneuver exceeds the maximum allowed deviation of 150 arcsec (examined in detail in [125] Chap. 3) from the target for 150 times within the

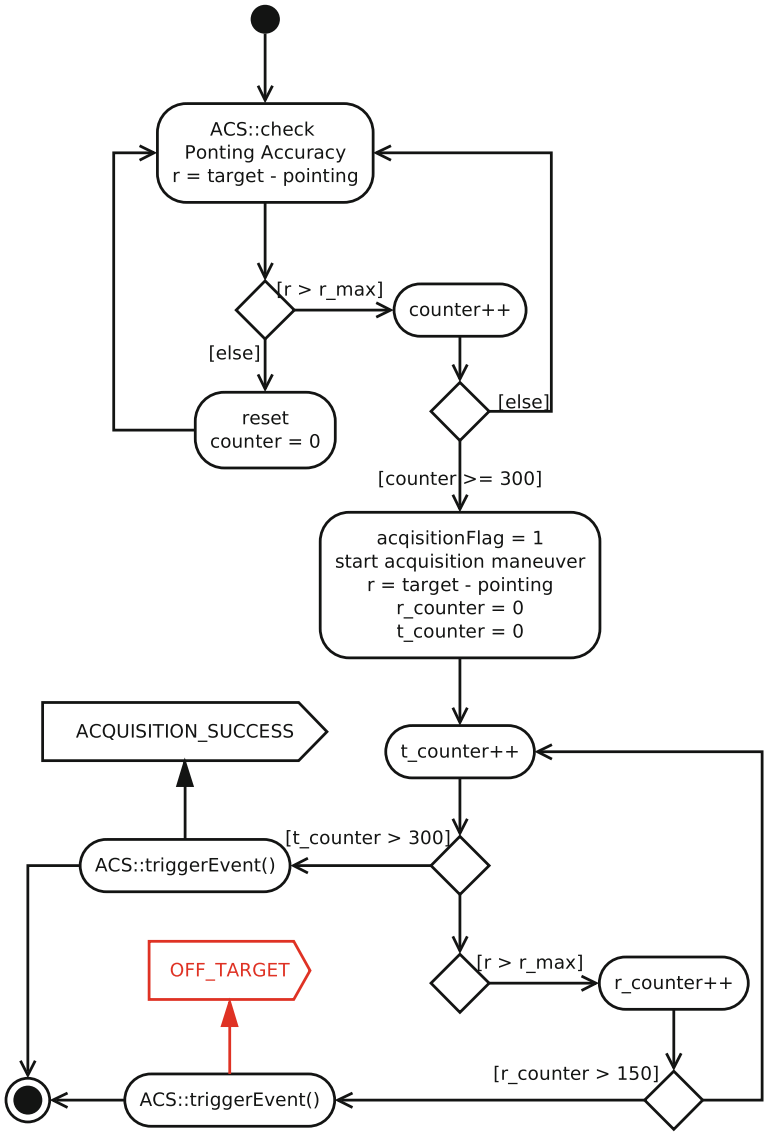


Fig. 10.44 ACS failure detection during target acquisition. © IRS, University of Stuttgart

30 s after beginning of the maneuver. In such case *FDIR* initiates the switch to Idle Mode. Ground interaction will be required to restore pointing functionality.

Component monitoring is executed at begin of each control loop as it is shown in Fig. 10.43 (in contrast to the older approach described in [125]).

10.6.9 ACS Autonomy

In [125] the ACS algorithms were developed to avoid switching into System Safe Mode upon device failure detection as long as possible—e.g. upon GPS failure. Depending on which unit fails these algorithms foresee a position propagation based on Kalman filters, magnetic field propagation etc. In the OBSW available for launch of the first FLP spacecraft these features will not yet be implemented. Their implementation into OBSW will be subject for subsequent implementation for FLP Generation 2.

In the theoretic ACS algorithms design of [125] the rotational rate data are provided by the FOG system and attitude determination is performed by the STR system delivering an attitude quaternion. Working STR and FOG systems constitute the ideal case and therefore guarantee the best quality regarding rate and attitude. For the loss of ACS sensors the method in [125] describes several different stages of quality for the determination of rotational rate and the attitude quaternion using existing sensors and the propagation of latest valid measurements.

In [125] it is demonstrated that in many cases and for a limited amount of time a spacecraft flight mode can be continued even if the quality of rotational rates determination or attitude determination is decreased.

- It was also proven in [125] that the satellite can be operated in degraded Idle Mode without STR and FOG systems.
- The same is possible for the Nadir Pointing Mode.
- The Inertial Pointing Mode cannot be operated without STR system but STR loss can be compensated for short durations.
- Target Pointing Mode operations do not provide sufficient accuracy for camera imaging if either rotational rate determination or attitude determination quality is reduced.
- However, the link to the ground station can be maintained if either rotational rate determination or attitude determination quality is reduced—with a reduced duration of the passover.

As mentioned before at least the first FLP Generation 1 missions will not yet feature these functions and will fall back to Safe Mode on loss of STR system, GPS system or MGM system.

10.6.10 ACS Events

This chapter provides an overview of the events involved in the ACS FDIR Failure Detection.

Event 10.40: LIMIT_VIOLATION

	Event Name	Event Severity	Event Parameter
	LIMIT_VIOLATION	LOW	Sensor object-ID, axis
Event description	This event will be generated if a measurement for one axis of an MGM exceeds the maximum expected limit		
Reaction	ACS sets sensor to invalid. FDIR counts errors and marks faulty when counter reaches threshold		

Event 10.41: DEVICE_FAULTY

	Event Name	Event Severity	Event Parameter
	DEVICE_FAULTY	LOW	Sensor object-ID
Event description	This event will be generated if a device is assumed to be faulty		
Reaction	FDIR counts failure messages for the device and shuts it down after counter reaches a threshold		

Event 10.42: MGM_MGT_WARNING

	Event Name	Event Severity	Event Parameter
	MGM_MGT_WARNING	LOW	–
Event description	This event will be generated if both MGMs show a limit violation on the same axis. It is assumed to be caused by an active MGT coil		
Reaction	–		

Event 10.43: MGM_DELTA

	Event Name	Event Severity	Event Parameter
	MGM_DELTA	LOW	–
Event description	This event will be generated if after comparison of the measurements of the MGMs, a significant delta is detected		
Reaction	Both MGMs are set invalid. FDIR counts the events. ACS uses magnetic field propagator		

Event 10.44: DEVICE_UNKNOWN_REPLY

	Event Name	Event Severity	Event Parameter
	DEVICE_UNKNOWN_REPLY	LOW	Device object-ID
Event description	This event will be generated if a deviceHandler cannot interpret a device response		
Reaction	FDIR counts failure messages for each device and shuts it down after counter reaches threshold		

Event 10.45: MISSED_REPLY

	Event Name	Event Severity	Event Parameter
	MISSED_REPLY	LOW	Device object-ID
Event description	This event will be generated if a deviceHandler Object does not receive a response from its device		
Reaction	FDIR counts the events and after a certain amount of events has occurred within a certain time period, the device will be rebootet. If the event keeps occurring, FDIR will shut the device down and mark it PERMANENTLY_FAULTY. Only ground can reactivate the device		

Event 10.46: FUSE_CURRENT_HIGH

	Event Name	Event Severity	Event Parameter
	FUSE_CURRENT_HIGH	LOW	Fuse object-ID
Event description	This event will be generated if the electric current on a fuse is significantly high		
Reaction	FDIR deactivates fuse if PCDU has not done already		

Event 10.47: ONE_GPS_VALID

	Event Name	Event Severity	Event Parameter
	ONE_GPS_VALID	LOW	Sensor object-ID
Event description	This event will be generated if after comparison of the measurements of the GPS receivers, only one GPS is deemed to be valid. Important Note: This event includes the working device, and not the faulty device		
Reaction	FDIR counts failure messages for each receiver and shuts them down after counter reaches 10. The counter will be increased for the two faulty sensors		

Event 10.48: GPS_INVALID

	Event Name	Event Severity	Event Parameter
	GPS_INVALID	LOW	Sensor object-ID
Event description	This event will be generated if after comparison of the measurements of the GPS receivers, one sensor is assumed to be faulty		
Reaction	FDIR counts failure messages for each receiver and shuts them down after counter reaches 10. The counter will be increased for this sensor		

Event 10.49: NO_VALID_GPS

	Event Name	Event Severity	Event Parameter
	NO_VALID_GPS	LOW	Sensor object-ID
Event description	This event will be generated if after comparison of the measurements of the GPS receivers, no sensor is assumed to be correct		
Reaction	FDIR counts failure messages for each receiver and shuts them down after counter reaches 10. All sensors will increase the counter		

Event 10.50: SUS_ZERO

	Event Name	Event Severity	Event Parameter
	SUS_ZERO	LOW	SUS object-ID
Event description	This event will be generated if there is a significant difference between two paired sensors and one is close to 0		
Reaction	FDIR marks the sensor as invalid		

Event 10.51: SUS_DIFF

	Event Name	Event Severity	Event Parameter
	SUS_DIFF	LOW	SUS object-ID
Event description	This event will be generated if there is a significant difference between two paired sensors, but isolation of a faulty one is not possible		
Reaction	–		

Event 10.52: FOG_SYS_INVALID

	Event Name	Event Severity	Event Parameter
	FOG_SYS_INVALID	LOW	Sensor object-ID
Event description	This event will be generated if the FOG system shows unexpected behavior but a faulty FOG cannot be isolated		
Reaction	Failure counter for all FOGs is increased until FDIR decides to mark FOGs faulty		

Event 10.53: FOG_INVALID

	Event Name	Event Severity	Event Parameter
	FOG_INVALID	LOW	Sensor object-ID
Event description	This event will be generated if one FOG shows corrupted measurements		
Reaction	FDIR counts the failures and marks the FOG faulty at some point		

Event 10.54: COIL_BROKEN

	Event Name	Event Severity	Event Parameter
	COIL_BROKEN	LOW	Coil object-ID
Event description	This event will be generated if no electric current can be detected for a coil of the MGT system		
Reaction	FDIR marks the coil as faulty		

Event 10.55: COIL_WARNING

	Event Name	Event Severity	Event Parameter
	COIL_WARNING	LOW	Coil object-ID
Event description	This event will be generated if the measured electric current of an MGT coil is not the same as the expected one since they should not differ		
Reaction	FDIR counts these events for each coil. If the counter reaches 5, the coil is marked faulty		

Event 10.56: FUSE_WENT_OFF

	Event Name	Event Severity	Event Parameter
	FUSE_WENT_OFF	LOW	Fuse object-ID
Event description	This event will be generated if a short is created due to a broken coil and a fuse in the PCDU is triggered in the PCDU		
Reaction	FDIR marks the coil as faulty		

Event 10.57: RW_SATURATION_REACHED

	Event Name	Event Severity	Event Parameter
	RW_SATURATION_REACHED	LOW	RW object-ID
Event description	This event will be generated if a RW reaches its saturation speed and desaturation has failed		
Reaction	FDIR powers off the RW		

Event 10.58: RATE_LIMIT_VIOLATION

	Event Name	Event Severity	Event Parameter
	RATE_LIMIT_VIOLATION	LOW	ACS Mode
Event description	This event will be generated if the rotational rate in an ACS mode exceeds the allowed limit for this mode		
Reaction	FDIR commands the system and ACS into a lower mode. Pointing → Idle Idle → Safe /Detumble		

Event 10.59: SUN_PROXIMITY

	Event Name	Event Severity	Event Parameter
	SUN_PROXIMITY	LOW	–
Event description	This event will be generated if a camera is active the angle between their direction and the sun direction is smaller than 60°		
Reaction	Will be used as an INFO event, no reactions are triggered.		

Event 10.60: SUN_AVOIDANCE

	Event Name	Event Severity	Event Parameter
	SUN_AVOIDANCE	LOW	–
Event description	This event will be generated if a camera is active the angle between their direction and the sun direction is smaller than 45°		
Reaction	FDIR deactivates the cameras to avoid for them looking directly into the sun if the path is followed further		

Event 10.61: ACQUISITION_SUCCESS

	Event Name	Event Severity	Event Parameter
	ACQUISITION_SUCCESS	INFO	–
Event description	This event will be generated if the target could not befixated but a new acquisition was successful		
Reaction	–		

Event 10.62: OFF_TARGET

	Event Name	Event Severity	Event Parameter
	OFF_TARGET	LOW	–
Event description	This event will be generated if the target could not befixated even after a new acquisition attempt		
Reaction	FDIR will command the satellite into Idle Mode		

10.7 TCS FDIR

10.7.1 TCS Failure Detection and Isolation

The thermal control subsystem (TCS) task is to maintain the correct temperature for all components on board the satellite within a specified range according to their operational state. In more detail this comprises:

- Storage of component temperature limits—operational and non-operational.
- Storage of control thresholds at which the control algorithm performs reactive actions. E.g. if a component is closing in on its lower operational temperature

limit—but has not yet reached it—a heater is activated to prevent the temperature from dropping further and prevent from having to shut the component down.
- Temperature monitoring of all components.
- Providing temperature TM data for all components at any time.
- Verifying correct function of temperature sensors.
- Managing redundancy for temperature sensors.
- Detection of heater malfunction.
- Managing redundancy for heaters.
- Reporting of heating actions as nominal events.
- Reporting of a detected malfunction as failure event.

As explained already in Chap. 8 there exist 66 temperature sensors on board the first FLP spacecraft—the "Flying Laptop"—of which 32 are individually placed at important locations or on/near components in the structure. As an exception, there are *two* sensors for each battery string since this is a crucial component. This allows for the detection of local temperature gradients in the battery strings. The other 34 are internal temperature sensors which are integrated within equipment units—providing a temperature in their respective housekeeping telemetry data packet. With these 66 sensors it is possible to determine a temperature for all components.

There exist six heaters—two on each structure module—for nominal thermal control. Radiator surfaces are used to transmit heat from the satellite to the environment. This setup might vary for other FLP missions depending on the payloads and layout of payload compartment.

Furthermore there are emergency heaters for the FLP OBC, TTC and the PCDU that are activated by the PCDU automatically in emergency situations. They are triggered when TCS control is not able to maintain the nominal state with the given conditions. The FDIR relevant monitoring functions are presented in Fig. 10.45.

This flow is repeated for every component's temperature computation:

- The starting point of the flow is in the top left. If the temperature T of a component is within the specified limit nothing has to be done and nominal operation continues as before.
- For temperature sensors one previous measurement value is stored for transient observation. The temperature gradient over time is computed, based on the recent sensor value. If this gradient is exceeding 10 K/s an Event T_GRAD_HIGH is issued. This leads to the sensor being marked as faulty.
- A temperature sensor might show the failure of delivering a plausible but out of range temperature such as below −50 °C or above +150 °C. Mission simulations have shown that these should not occur at any time. These values are also the measurement limits of the PCDU which should not be reached at any point. Such values indicate that there is a failure in the sensor hardware which leads to the *TCS_Controller* object issuing a failure Event for this sensor. The device will then be marked as faulty. In the subsequent TCS computation cycles this sensor will be ignored and the value of a redundant unit will be used instead.

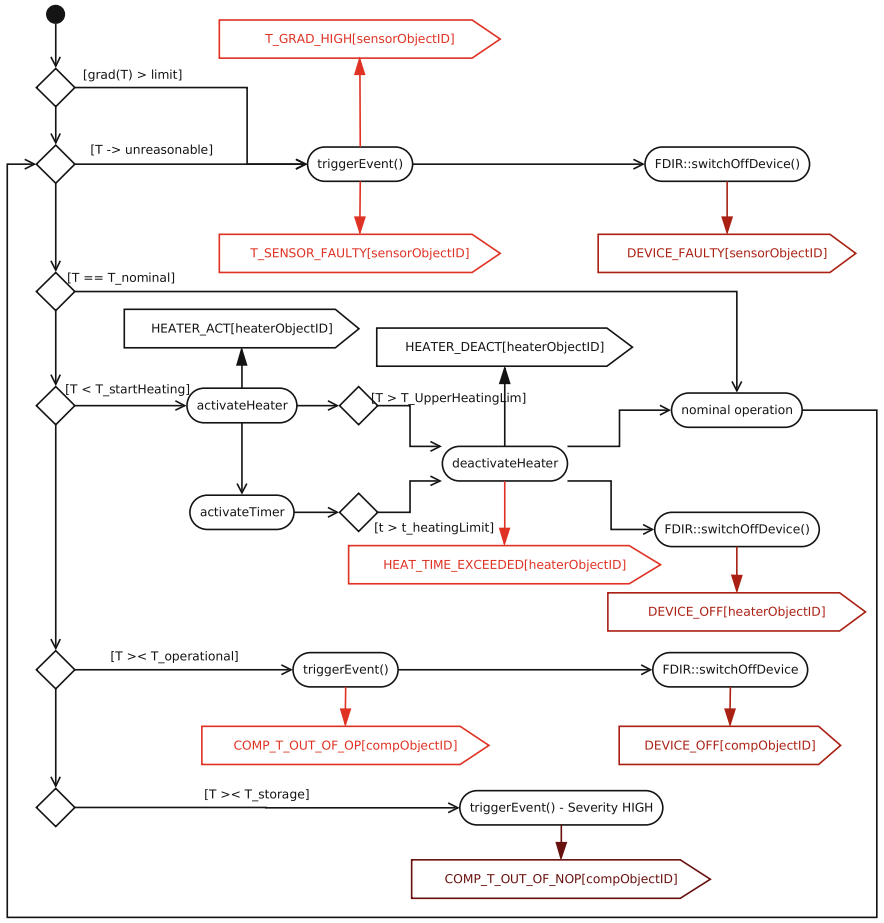


Fig. 10.45 TCS Failure Detection and Isolation. © IRS, University of Stuttgart

A temperature sensor cannot be deactivated or powered off. To simplify Event and command handling however the same Event/action approach will be taken, as if it were a standard avionics unit with a switch for deactivation. Inside the OBSW the sensor value is set to invalid and is ignored from this point onwards.

- The nominal TCS control case is also included in Fig. 10.45:
  If a component temperature drops below a control threshold the control algorithm will decide that a heater needs to be powered to prevent the component from cooling off. The activation of a component such as a heater will cause the generation of a nominal Event as shown in Fig. 10.45. The heater will heat up the component until an upper control limit is reached and further heating is not deemed necessary anymore (see Figs. 8.10 and 8.11 in Sect. 8.3.2.2). When the heater is deactivated, a second *INFO* Event will be generated to provide transparency of on-board processes for the ground operator.

- Heater malfunction—usually this would mean a broken wire or heater foil—is detected by the following mechanism: A timer is activated upon heater activation. If the heated component still drops below its specified temperature limit within the expiration of the timer, this will lead to identification of a malfunction in the heater. The monitoring period is defined to be 3600 s. Timer expiration will induce a HEAT_TIME_EXCEEDED Event that includes the object-ID of the affected heater. *FDIR* will then electrically deactivate the heater by means of the corresponding PCDU switch. An Event DEVICE_OFF will be generated by *FDIR* including the heater object-ID to allow observation of the processes on board by a ground operator.
  Observing this heater timeout might in fact have two reasons:
  - The heater circuit really is defective. It is not expected that a reactivation of the heater is harmful. The *PSS_Controller* will detect that the heater is activated but shows no power consumption. It will issue a WARNING_NO_CURRENT Event which will lead to *FDIR* marking the heater as faulty.
  - The monitoring period for a particular heater is to be fine adjusted during thermal-vacuum test of a spacecraft and might need fine adjustment again during in-flight commissioning. Therefore the algorithm needs close monitoring by the thermal control engineer on ground.
- If the nominal heater of an equipment or spacecraft Module is marked as faulty *TCS_Contoller* will automatically use the redundant heater in the next cycle.
- If the temperature of the equipment falls below the operational temperature limit an Event COMP_T_OUT_OF_OP with the component's object-ID will be generated. *FDIR* will deactivate the device with this object-ID and generates a DEVICE_OFF event including the component's object-ID.
- However if a critical situation causes the temperature to decrease even further and drops below the survival temperature limit for a component an Event COMP_T_OUT_OF_NOP with the component's object-ID is generated. Since FDIR is not able to perform any corrective actions in such case anymore the severity of this event is *HIGH* to alert ground.

10.7.2 TCS Failure Recovery

Temperature sensors that have been identified as faulty will be marked accordingly by the system FDIR. However since they cannot be powered off they will still deliver measurement values. In the implementation of the Thermal Control System as shown in Fig. 10.46, it is foreseen that after the calibration of the sensor values from raw values to degrees (°C) the *TCS_Controller* will check the health status of a sensor before evaluating a temperature value for a component.

If the health status of a sensor is non-healthy the value of this sensor will be ignored. A redundancy order is determined for all sensors, defining which sensor

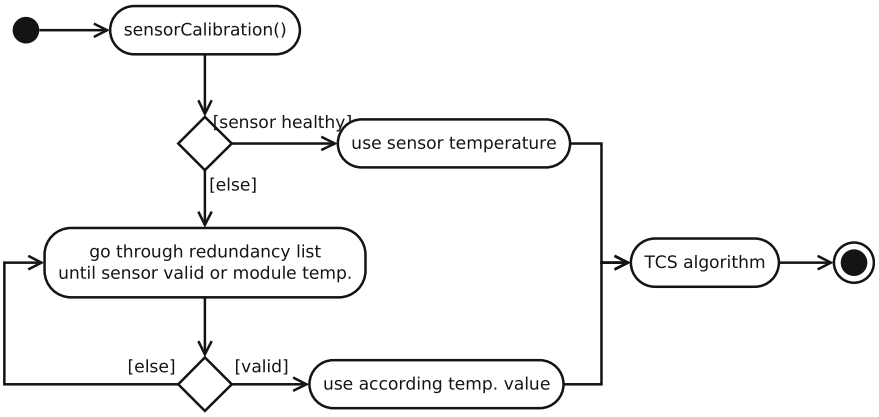


Fig. 10.46 Thermal Control System Sensor Redundancy. © IRS, University of Stuttgart

can be used as alternative in a failure case. This order depends on the location of the sensor. A sensor in close proximity is usually predefined—see Table 8.1. As last redundancy option for all sensors the temperature of the structure module is used. The module temperature is calculated according to Table 8.2. This might lead to a very inaccurate value for a component temperature and corrective actions should be undertaken by ground. This situation can only occur after more than a single failure.

The heaters for nominal control are implemented single redundant. If both heaters are healthy, they will be used in parallel for faster heating. If one heater is marked as non-healthy it will be ignored. The timeout period varies depending on the number of available heaters.

10.7.3 TCS Survival Mode

In case where the battery SoC drops significantly *FDIR* commands a system mode switch to the Safe Mode (see Sect. 10.3). If for any reasons the satellite will start to tumble randomly it cannot be guaranteed that the solar panels are oriented towards the Sun. This means that it cannot be guaranteed that there is enough power input from the solar panels into the system to perform nominal Safe Mode operation such as heating all components if necessary.

Therefore a TCS Survival Mode has been defined in which power consumption is decreased even further. The controlling mechanism in this mode is shown in Fig. 10.47. The TCS Survival Mode is commanded by *FDIR* as soon as the SoC of the battery reaches 35 % of the expected End-of-Life capacity. SoC limit violations are reported by the *PSS_Controller*.

In normal TCS control mode, the goal is to keep *all* components on board the satellite at least in their survival temperature range. In the TCS Survival Mode it is accepted that for example the temperature for payload equipment drops below their

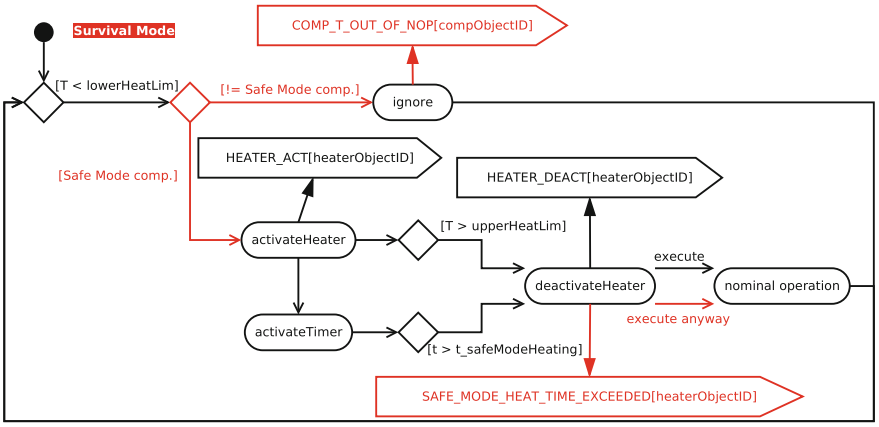


Fig. 10.47 TCS Survival Mode Diagram. © IRS, University of Stuttgart

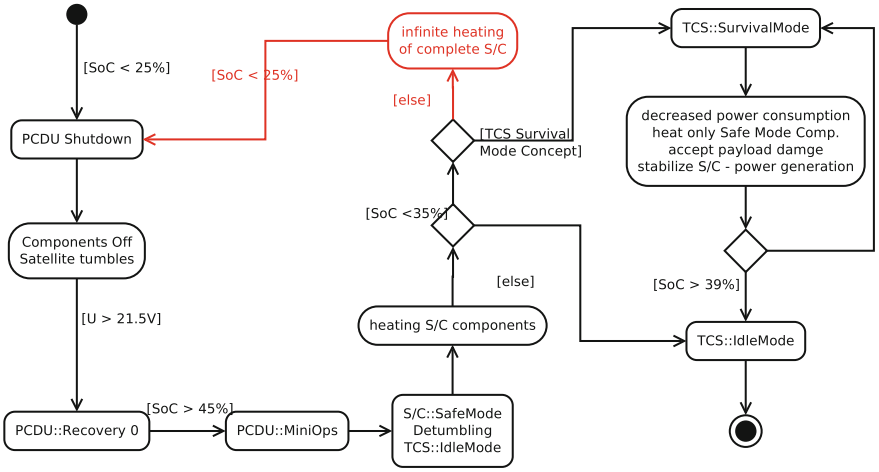


Fig. 10.48 TCS Survival Mode Concept. © IRS, University of Stuttgart

non-operational lower limit temperature to avoid activation of heaters due to priority of minimum power consumption. This means for the purpose of improving the chances for system survival in such a scenario it is accepted that payload equipment might be damaged when cooling off.

The only components that are heated in this TCS Survival Mode are the ones required to perform the detumbling maneuver—in Sect. 5.5.3 declared as Safe Mode components.

However, even these are only heated for 1800 s after which undertemperature damage is accepted in order to avoid an endless-loop of heating and low-power state of the battery—see Fig. 10.48. When the satellite is re-stabilized and back in

Safe Mode and approximately oriented towards the Sun the battery SoC is expected to increase again. Upon 39 % SoC, the TCS is switched back into its nominal mode to heat all S/C components and prevent further cooling off of components and their damaging.

10.7.4 TCS FDIR Events

Event 10.63: T_GRAD_HIGH

	Event Name	Event Severity	Event Parameter
	T_GRAD_HIGH	LOW	Sensor object-ID
Event description	This event will be generated if the gradient between two temperature sensor measurements is too high, indicating a failure in the sensor		
Reaction	FDIR marks the sensor as faulty		

Event 10.64: T_SENSOR_FAULTY

	Event Name	Event Severity	Event Parameter
	T_SENSOR_FAULTY	LOW	Sensor object-ID
Event description	This event will be generated if unreasonable sensor measurements are detected		
Reaction	FDIR marks the sensor as faulty		

Event 10.65: HEAT_TIME_EXCEEDED

	Event Name	Event Severity	Event Parameter
	HEAT_TIME_EXCEEDED	LOW	Heater object-ID
Event description	This event will be generated if a heating process takes too long for the component to reach a certain temperature. Indicates that the heater is broken or that the thermal control system design has flaws		
Reaction	FDIR switches the heater off. Ground may adapt the TCS		

Event 10.66: COMP_T_OUT_OF_OP

	Event Name	Event Severity	Event Parameter
	COMP_T_OUT_OF_OP	LOW	Device object-ID
Event description	This event will be generated if a component leaves its operational temperature range		
Reaction	FDIR shuts off the device		

Event 10.67: DEVICE_OFF

	Event Name	Event Severity	Event Parameter
	DEVICE_OFF	MEDIUM	Device object-ID
Event description	This event will be generated if FDIR decides to switch off a device		
Reaction	–		

Event 10.68: DEVICE_FAULTY

	Event Name	Event Severity	Event Parameter
	DEVICE_FAULTY	MEDIUM	Device object-ID
Event description	This event will be generated if FDIR decides to switch off a device that is deemed faulty		
Reaction	–		

Event 10.69: COMP_T_OUT_OF_NOP

	Event Name	Event Severity	Event Parameter
	COMP_T_OUT_OF_NOP	HIGH	Device object-ID
Event description	This event will be generated if a component leaves its storage temperature range. This is dangerous and might damage the component		
Reaction	Ground has to interfere		

Event 10.70: SAFE_MODE_HEAT_TIME_EXCEEDED

	Event Name	Event Severity	Event Parameter
	SAFE_MODE_HEAT_TIME_EXCEEDED	HIGH	Heater object-ID
Event description	This event will be generated if a heating timeout expires in Safe Mode		
Reaction	Ground has to interfere. The component will be activated anyway to attempt saving the mission		

10.8 Payload FDIR

This chapter describes the basic methods for payload FDIR that are implemented in the initial version of the FLP Generation 1 OBSW. FDIR for payload devices is attempted to be as simple as possible. More complex examinations of payload problems will be performed by ground. Rationale for this decision is:

- The FLP Generation 1 payload computer PLOC and its firmware are an academic experimental system. The included functions may need in-flight updates.
- Testing of payload FDIR functions in the OBSW (analysis of payload/PLOC/MMU/DDS failures) will have been performed with limited effort up to launch of the first FLP based "Flying Laptop" mission.
- Sophisticated software features cannot be implemented until launch of the first FLP based mission.

In summary substantial payload FDIR considering all details of payload control via OBC and potential payload data preprocessing on board cannot be realized before an overall FLP Generation 2 architecture as depicted in Fig. 9.14 is available.

10.8.1 General Failure Handling

The communication with payload devices is routed in FLP Generation 1 via the PLOC as shown by the interfaces and wiring in Fig. 1.6. Figure 10.49 illustrates a very simplified version of Fig. 1.6. The payload computer, all payloads and the OBC are powered via dedicated switches in the PCDU. Power monitoring—monitoring of power consumption, switch activation and deactivation—is controlled in the OBC for all payloads. The OBC commands the PCDU to trigger switches if payload devices are required to be available. Temperature sensors are routed via analog/digital converters in the PCDU to the OBC where thermal control is performed—again for all devices including all payloads

The OBC ↔ payload communication can be described very simplified as follows:

- A command to the PLOC or a payload is either stored in a mission timeline on board or is directly sent from ground to the payload. The communication is established via *deviceHandler* objects—as introduced in Sect. 3.2.2—that are used as interfaces to the payload devices. They encapsulate the corresponding packet for the payload devices.

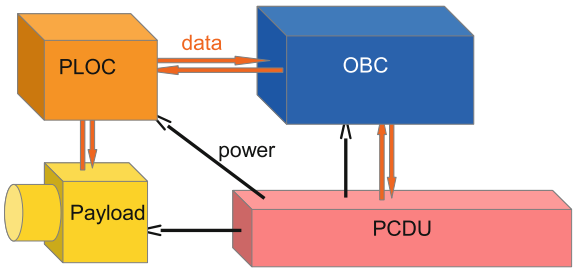


Fig. 10.49 Simplified electrical architecture between PLOC, OBC and PCDU. © IRS, University of Stuttgart

- The PLOC *deviceHandler* (PLOC-DH) forwards all commands to the PLOC hardware (PLOC and particular payload commands). Depending on the content the PLOC forwards the ground command to a payload or it reformats the information into a notation suitable for the payload device.
- The payload executes the command and sends back science data or HK TM with a status report.
- The packets with TM provided by a payload are encapsulated in a PLOC packet with CRC and are forwarded to the OBC/OBSW. The PLOC only examines science data packets and processes these—no payload housekeeping packets. It either stores the science packets in the MMU or forwards them to the Data Downlink System directly. As shown in Fig. 10.50 the PLOC *deviceHandler* unpacks all data and the payload specific packets are forwarded via the PLOC *deviceHandler* to the dedicated payload *deviceHandler* in the OBSW. The PLOC also reports whether a received packet from OBSW included a bad checksum.
- Device specific failure detection is mostly restricted to a failure in the communication between the devices. If failures are detected by a *deviceHandler* the same method is applied as for the bus components described in Sect. 10.4 and as shown in Fig. 10.51.
  This means that after a certain amount of failures within a certain time a counter value threshold is exceeded and the device is powered down and is marked faulty by the system FDIR. The mechanism is explained in Sect. 10.4.3.

Failure handling regarding thermal or power issues are already covered in Sects. 10.3 and 10.7. No further device specific failure handling is implemented for the current payload module design of the FLP Generation 1.

If the PLOC is identified as faulty by either of the mentioned subsystem failure detection functions it is deactivated and interaction by ground is required. *FDIR* does not perform automatic PLOC switch to the redundant unit. In such case the satellite system is commanded into Idle Mode which automatically leads to a deactivation of all payload devices.

If the PLOC is healthy but a payload equipment is identified as faulty by either of the mentioned subsystem failure detection functions, it is deactivated, but the current operational mode is kept as shown in Fig. 10.52.

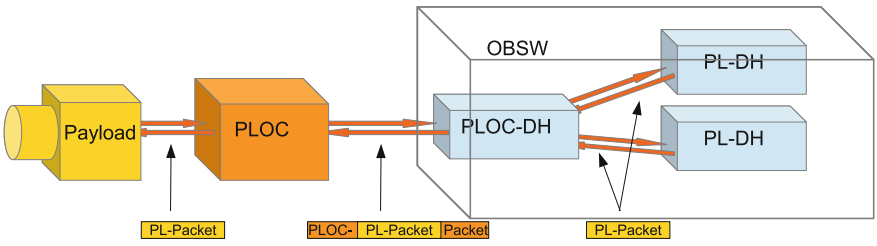


Fig. 10.50 Payload Data Flow. © IRS, University of Stuttgart

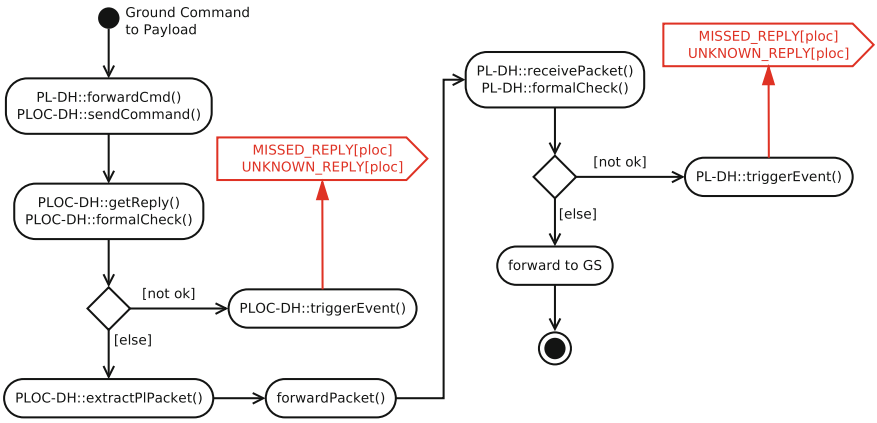


Fig. 10.51 Payload Device Failure Detection

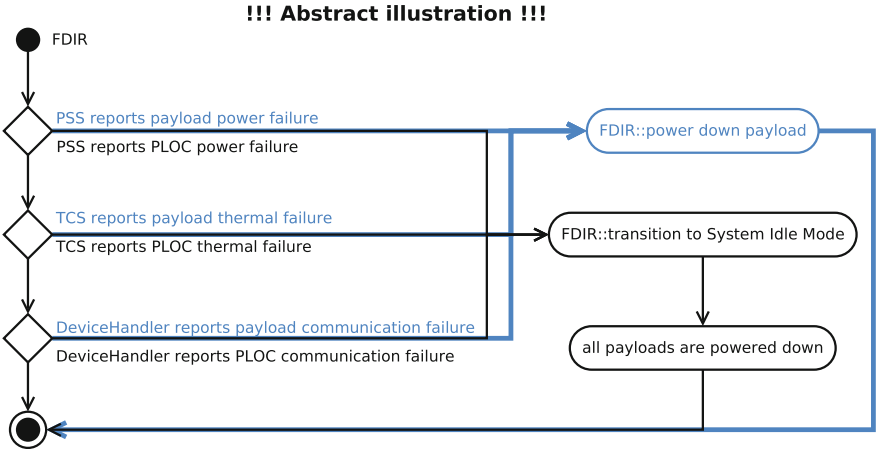


Fig. 10.52 Deactivation of Payloads in a failure case. © IRS, University of Stuttgart

This approach allows for the payload to show several communication failures before being shutdown—in analogy to the satellite bus devices (see Sect. 10.4).

Note: The payloads of the first FLP based satellite—the "Flying Laptop"—are not completely based on radiation-hardened electronics and have no flight heritage from earlier missions. Also, they have not undergone radiation robustness tests. Thus it is expected that effects like SELs or sporadic SEUs can occur. Therefore it is important especially in the payload Commissioning Phase that a payload is not shutdown immediately after a single failure symptom. Further mechanisms for payload management and failure recovery are described in [135].

10.8.2 Payload FDIR Events

There are no dedicated payload Events defined for the FLP Generation 1 design based on the PLOC as payload controller. The OBSW will be upgraded for the FLP Generation 2 common OBC for platform and payload management as outlined in Sect. 9.3.2.

10.9 Failure Propagation

The mechanisms introduced in the previous chapters follow the paradigm of keeping the satellite operational or in Idle Mode as long as possible—with reasonable complexity in failure detection and recovery. For detailed analysis by ground *FDIR* generates Events that support diagnosis. But extensive analysis has to be performed by a ground operations team.

The *FDIR* task runs with the highest frequency and Event evaluation is performed in every FDIR cycle to guarantee quick response to failures and hazardous situations. Some Events can have different consequences depending on the situation. Therefore this FDIR concept foresees to collect all Events from one cycle before determining recovery measures. This prevents that the same actions are processed several times and that all Events are treated separately although they might be interrelated.

A detected failure is handled as local as possible. With the assembly concept a switch to a redundant device can be realized on low level, when FDIR marks a nominal device as being faulty. If more than one unit of an assembly is identified as faulty and the assembly cannot keep up the required operational mode, it reports accordingly by Event to the subsystem. The subsystem will propagate the report upwards until *System* decides to perform a mode switch into the next fall-back mode, which are

- Idle Mode for pointing Modes, and
- Safe Mode for Idle Mode.

In summary the overall system state is switched down to a safe state to avoid critical failure propagation. Device failures are not deemed critical and need confirmation before the devices are powered down.

- Due to the different steps of the SoC surveillance energy supply of the system is as good as possible. A quick drop of SoC or voltage level leads to immediate deactivation of payloads and to re-pointing of the satellite towards the sun in Idle Mode. The remaining energy is analyzed to be sufficient, even if the satellite enters eclipse at the activation of the Idle Mode switch.
- The satellite will remain in Idle Mode as long as possible. If the SoC drops further, severe damage must have occurred and the switch to Safe Mode and ground interaction is mandatory.

- Device failure induced discharging of the battery is avoided as good as possible with four different steps of components deactivation.
- Failures in the power subsystem or a significant drop of the State-of-Charge of the battery leads to immediate component switch-off or to transition into Safe Mode to avoid damaging further components in the system or the battery.
- In the worst case the satellite deactivates itself until the battery SoC has recovered to a reasonable level. The PCDU detect this and boot the system.

10.10 Satellite Safe Mode Implementation

The satellite modes and mode transitions are already depicted in Fig. 2.1. The Detumble Mode and the Safe Mode here only differ by spacecraft rotational rates but the activated ACS equipment and the control laws are the same for both modes. For the Safe Mode some specific characteristics of the FLP Generation 1 have to be listed however.

- When a Safe Mode transition is initiated by whatever Event the satellite transits to ACS Safe Mode and limits attitude acquisition to the Safe Mode sensors and actuators.
- ACS control will orient the spacecraft to sun-pointing direction and will initiate the spin-up for orientation stabilization in eclipse.
- The OBSW will shut down all payloads.
- And the OBSW will shut down electrically all non Safe Mode sensors and actuators in case where the Safe Mode was triggered by a power limitation event.
- The Combined-Controller in the CDPI PCDU will reconfigure a failed OBC unit to the redundant side in case where the Safe Mode was triggered by such an OBC failure.

In contrast to a classic Safe Mode implementation the OBSW of the FLP Generation 1 will not automatically trigger a full switchover to a redundant equipment set (OBC + PCDU + ACS + data lines/SpaceWire).

The full spacecraft switchover to redundant can be commanded by ground via Flight Procedure for the FLP Generation 1. For the FLP Generation 2 a full B-side Safe Mode configuration is foreseen—based on the SpaceWire network implementation outlined in Sect. 9.3.2

Safe Mode initiation by command:

- The Safe Mode can be initiated by OBSW command from ground.
- The Safe Mode can be initiated by OBC Processor-Board re-powering or reconfiguration via HPC-1 command from ground.

Recovery from Safe Mode is achieved via simple commanding of the spacecraft to a higher mode—such as Idle Mode—after having analyzed and eliminated/isolated the root cause of a potential fallback by ground intervention.

10.11 Ground Based FDIR

On-board FDIR cannot always provide the best or complete identification and recovery for all failure types. The approach for OBSW *FDIR* is to realize a save solution that decreases the risk for the spacecraft to a minimum. Furthermore it is the goal to provide sufficient observability to ground for failure identification and sufficient commandability for failure isolation and recovery [140].

10.11.1 Observability of the Space Segment

The goal of *FDIR* and the software framework in this context of ground interaction is to allow for ground to have full observability of system, subsystem and device characteristics and of processes and Events on board the satellite. All Events generated during operational scenarios are stored in a separate TM memory area on the I/O-Board (see Sect. 4.3.2) and can be dumped as a first transaction upon established contact to the ground station. The usage of PUS-Services and TM Virtual Channel allocation allows for configurable, priority controlled HK and Event TM downlink.

All events reported in the OBSW—independent of the source object—are visualized in the mission control system. Failure events are marked in different colors. This is illustrated in Fig. 10.53 which shows an exemplary event history in SCOS-2000.

In the black frame at the top of the image the heading shows “EvID” which is the *eventID* in decimal number, “Severity” and a message text for further explanation.

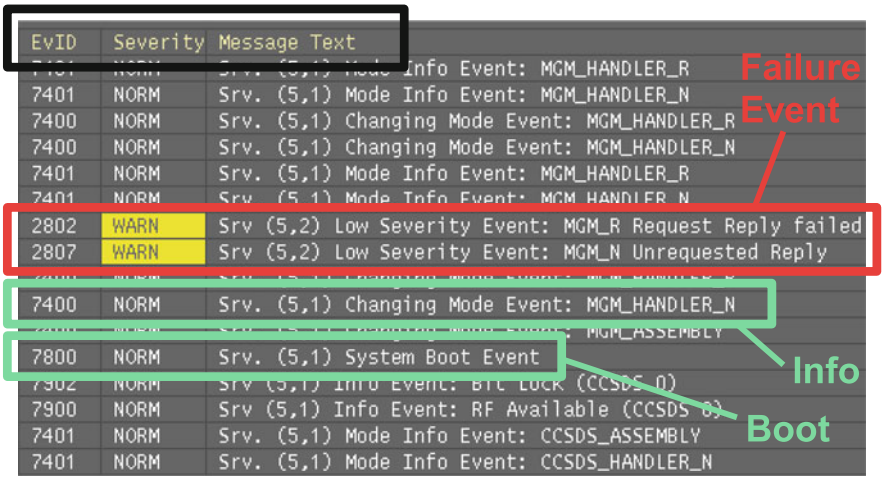


Fig. 10.53 Event History in SCOS-2000. © IRS, University of Stuttgart

This example shows the boot up of the OBC marked with a green frame, which is a normal information event with ID 7800 which indicates that the boot was successful and the according objects have been instantiated. All tasks have been started and the software is up and running.

A second information in the example is a Mode Change Event for the MGM *deviceHandler* with *objectID* 'MGM_HANDLER_N' and the *eventID* '7400'. In the red frame two non-nominal Events with severity LOW are shown, which indicate a warning that failures have been detected. The severity field of this Event is marked yellow to indicate a failure event. They occurred due to the mode change command in this example and are not expected to be severe. In that case no reactions would have been initiated.

As explained in Sect. 3.3.3 the class of information Events indicates a nominal change or transition that happened on-board. These Events can be used in command schedules to trigger follow-up commands. Additionally there are three different severity levels for Events:

- LOW: indicates that a failure detection method has detected a failure in the system and triggered a failure Event. This does not necessarily lead to recovery actions.
- MEDIUM: Indicates an event from OBSW *FDIR*. MEDIUM events are triggered by *FDIR* if responsive measures are taken to LOW-severity events.
- HIGH: indicates that severe Events have happened on-board. These can trigger either Safe Mode or Survival Mode Switch due to low available energy or due to a problem somewhere in the OBDH system. An Event with severity class HIGH could also indicate that it cannot be handled by OBSW *FDIR* e.g. if a component exceeds its upper non-operational temperature limit.

The PUS Services also allow for very detailed reports of housekeeping data, *datapool* dumps and reports to analyze on-board processes. With the customized services (201, 1) and (202, 1) all mode tables and health status of devices can be requested and examined on ground. These measures and features of the FLP platform in sum provide a 100 % complete S/C observability.

10.11.2 Commandability of the Space Segment

Complementing the observability of the spacecraft, the platform design needs to offer sufficient commandability functions for failure recovery by ground [140]. The FLP offers the following features:

- PUS-Service 1 provides acknowledgment of all commands submitted to the space segment.
- Customized Subservices (2,128)–(2,133) of the Device Commanding Service can be used to directly access and command devices without needing to address

the complete system. This can be used for detailed analysis of a certain device if it has been identified not to perform correctly.

- The On-Board Monitoring Service (12) can be adjusted such that specific variables are examined in greater detail than during nominal operations. Monitors can be assigned via TC to any of the available *datapool* variables.
- The Event-Action Service (19) configuration can be extended to define specific actions/commands that are to be executed upon detection specific Events. This allows for a quick extension of the on-board FDIR functionality without needing to update the complete software if event-actions are identified during flight which are not yet covered by the presented OBSW *FDIR*. These actions can be activated and deactivated during operations with this service. However the actions consequentially will need to be uploaded with a software patch.
- If required, devices can be switched into a dedicated mode for ground analysis or for a continuing operation using the Mode Control Service (200). Since all objects in the FLP OBSW have a dedicated object-ID (compare Sect. 3.2.8) they can be accessed via this service as well.
- The Payload Control Service (203) is used to configure the payload computer and command the payload devices.

In emergency situations, the already cited High Priority Commands can be utilized to activate and deactivate devices. HPCs are routed directly from TTC to PCDU without being addressed by OBSW so that their execution is independent of OBC Processor-Board or OBSW. They are decoded by the PCDU controller and can be used to change switches. For every switch there is a dedicated HPC as can be taken from Table 10.4.

With the HPCs all devices can be activated and deactivated except for the PCDU controller itself which is implemented as a single-failure-tolerant inherently self-monitoring system. All OBSW objects can be accessed via PUS-Services. This offers 100 % commandability coverage for an FLP based spacecraft.

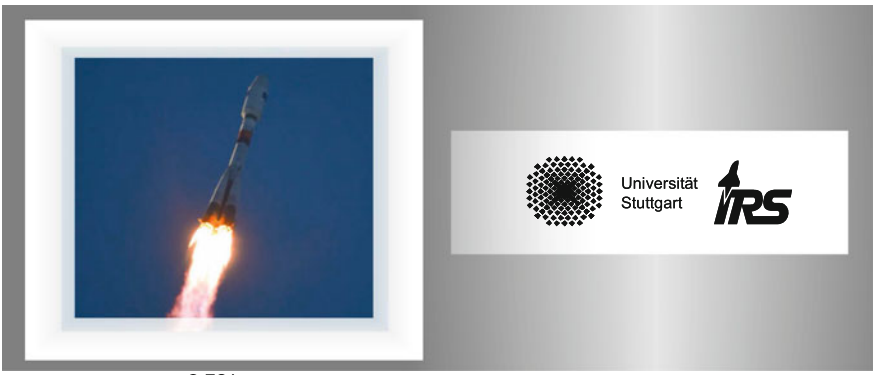
10.11.3 Ground Analyses

The commandability and observability of the spacecraft provide the required possibilities to examine processes and statuses on board the satellite. In case of a malfunction they are mandatory for failure recovery. Especially during the initial flight phases—LEOP and Commissioning Phase—and in particular for the first FLP based satellite—the "Flying Laptop"—it will be a challenge to distinguish between software bugs or flaws in the control algorithms and actual S/C hardware errors. This has to be examined in detail whenever unsolved issues appear on-board.[4]

[4]It has to be noted that despite all careful design, implementation and testing, the OBSW is an academic product that is implemented with limited resources and experience.

Chapter 11
Satellite Mission Phases and Planning

Jens Eickhoff, Michael Lengowski and Kai-Sören Klemich



Abstract The chapter covers the flow of the work steps over the diverse mission phases, from mounting to a launcher, the launch, spacecraft power up, the first contact initiation until end of life deorbiting.

Keywords Launcher interfaces · Activities before launcher integration · Pre-launch activities · Launch and early orbit phase (LEOP) · Spacecraft power-up · Link establishment · Commissioning phase · Nominal operations · End of life phase

J. Eickhoff (✉)
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

M. Lengowski · K.-S. Klemich
Institute of Space System, University of Stuttgart, Stuttgart, Germany
e-mail: lengowski@irs.uni-stuttgart.de

K.-S. Klemich
e-mail: klemich@irs.uni-stuttgart.de

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_11

11.1 Overview

The FLP platform is designed for lightweight satellites of the 150 kg class which typically will be launched as piggy-back payload (also called "secondary payload" or "auxiliary payload") together with a primary launch passenger. In most cases such secondary passenger payloads of the launcher are not allowed to emit any RF signal/noise during launch, which only can be assured by deactivation of the satellite during launch. Therefore the FLP also was designed for being launched in Off Mode.

The "Flying Laptop" as first FLP-based satellite is designed as secondary passenger compatible with two launchers in order to offer flexibility in the launcher choice.

- The first launch system is a Soyuz 2-1b rocket with a Fregat-M upper stage launched from Baikonur, Kazakhstan. The launch site is located at approximately 45.6°N and 63.4°E. Its local time is Universal Time Coordinated (UTC) +6 h, i.e. the time difference to Germany will be +5 h in winter and +4 h during daylight savings time.
- The second launch system is the Polar Satellite Launch Vehicle, (PSLV) of the Indian Space Research Organization (ISRO), launched form from Sriharikota, (SHAR) located 80 km north of the city of Chennai (13.7°N and 80.2°E).

For the mission of the "Flying Laptop" the initial satellite orbit is planned to be a "Sun-Synchronous Orbit" (SSO) with an altitude below 650 km. The altitude requirement results from the pursued compliance with the European Code of Conduct for Space Debris Mitigation [1] which implies to deorbit the satellite after end of mission within 25 years. At an altitude of 600 km and at a "Local Time of Descending Node" (LTDN) of approximately 09:00 h the inclination of this orbit is ca. 97.79°, its orbital period is ca. 5800 s, 96.7 min or 1.61 h. In the initial orbit, the sunlight periods will last for ca. 4020 s, 67 min or 69 % of the orbital period and the eclipse periods will be roughly 1745 s, 29 min or 30 % of the orbital period. The remaining percent of the orbit the satellite will be in penumbra. For more details please also refer to the orbit analysis in annex Sect. 17.10.

The selected launch vehicles will first release the main payload and the secondary passengers thereafter one by one. While the secondary passengers of the PSLV will be injected into the same orbit, the Soyuz Fregat upper stage can release the auxiliary passengers at different altitudes.

Target orbit data will be provided before launch and the actual estimated orbit data shortly (some minutes) after separation. The applied data format is a time, a position vector and a velocity vector. The data will be used for planning the IRS antenna station passovers (see Chap. 12) and will be shared with DLR GSOC for planning the passovers of the antenna station in Weilheim (see Chap. 13).

This general process is backed up by some information available on the German Agency DLR[1] satellite TET-1. TET-1 was launched in a SSO with a LTAN 11:27 h [43] at 12:41 h local time in Baikonur or 8:41 h MESZ [45]. It was separated from the launcher at 9:33 h MESZ and the first contact in Germany was at 11:54 h MESZ [44] by the DLR "German Space Operations Center" (GSOC). Due to the LTAN being in the morning and not the LTDN as for the "Flying Laptop", TET-1 was launched in northern direction, but the remainder of the procedure can be assumed to be similar between the two.

11.2 Launcher Mechanical Interface

The mechanical interfaces of the selected launchers are rather similar to each other. The maximum envelope given by the fairing are

- 1000 mm × 800 mm × 800 mm for Soyuz Fregat and
- 600 mm × 700 mm × 850 mm (radial × transverse × height) for the PSLV.

Figure 11.1 shows the payload configuration of both rockets. It can be seen that the separation adapter axes are different in both cases:

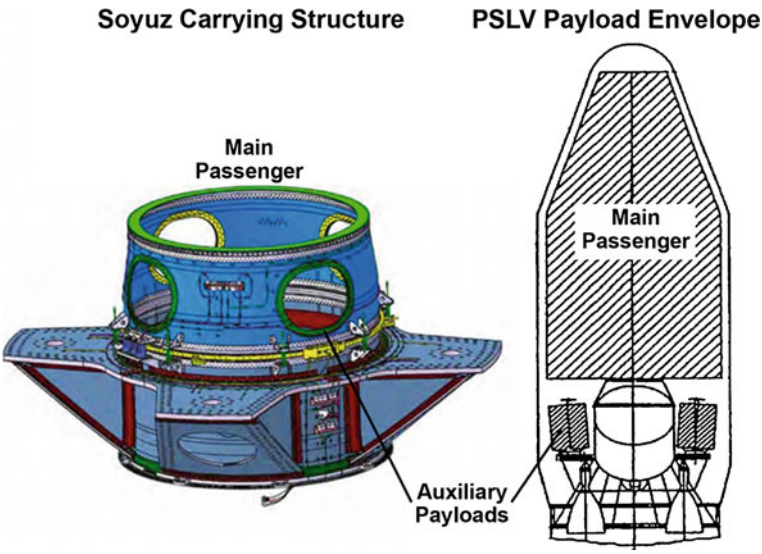


Fig. 11.1 Auxiliary satellite configuration of Soyuz Fregat [138] and PSLV [137]

[1]Deutsches Zentrum für Luft- und Raumfahrt.

- In case of the Russian launcher the auxiliary satellite is mounted on the Soyuz Carrying Structure ASAP-S.
- Auxiliary payloads of PSLV will be placed on the Vehicle Equipment Bay, (VEB) tilted at an angle of 4°–5° to the longitudinal axis of the vehicle in order to provide safe clearance from the vehicle during separation. Both auxiliary payloads are placed below the main passenger.

The Indian Space Research Organization (ISRO) provides qualified separation systems for piggyback satellites in a mass range of 50–150 kg (excluding the separation adapter). For the "Flying Laptop" the separation adapter for satellites with a mass of up to 120 kg can be used. Figure 11.2 shows the technical draft of this adapter and Table 11.1 lists its technical characteristics.

This launch adapter can be applied as separation system for both launch options, PSLV and Soyuz.

The separation system consists of an upper and a lower ring adapter locked together by means of a "Ball Lock" mechanism. The mechanism is opened by pyrotechnic bolts rotating a retainer ring between the upper and lower rings. The spacecraft is ejected then by helical compression springs positioned in between these rings.

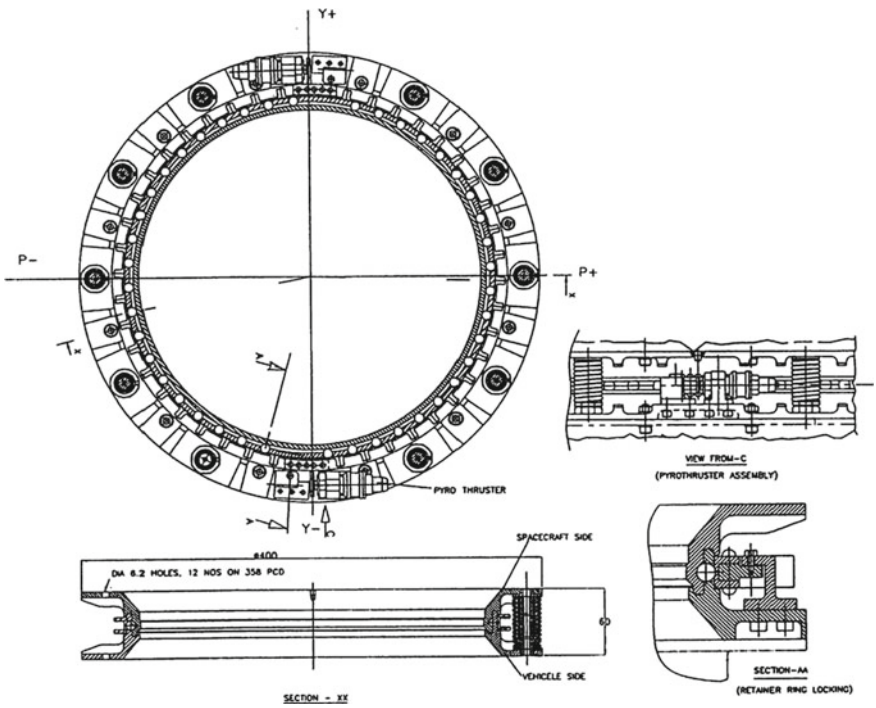


Fig. 11.2 Separation system configuration of PSLV [137]

Table 11.1 Characteristics of the launch adapter for 120 kg satellites

Class of Satellite	Designation	Mounting Pitch Circuit Diameter [mm]	No. & Size of bolts	Nominal Separation Velocity [m/s]	Mass [kg]	Mass retaining on the Satellite [kg]
120 kg	IBL-298	298	M6x12	1	4.5	1.1

Mounting the satellite physically to the launch adapter is performed already at spacecraft manufacturers premises—in case of the FLP mission "Flying Laptop" at the University of Stuttgart. This includes mechanical mounting to the adapter—Fig. 11.2—for the launcher secondary passengers—see Fig. 11.1.

11.3 Launcher Electrical Interface

The electrical interface between the auxiliary satellites and launcher is applied via one umbilical link for both launchers but in different configurations:

The PSLV umbilical connector (DBAS 74 12 0 SN 059 on Spacecraft side) consists of an interface for battery trickle charging (pin 1 and 2), a separation detection for launcher (pin 3 and 4) and for the satellite, as well as a minimum set of satellite checkout pins. Pins 3 and 4 are to be shorted on the satellite side with a short-loop. This connection on the vehicle side will be used to monitor satellite separation through vehicle telemetry in flight. The scheme is depicted in Fig. 11.3.

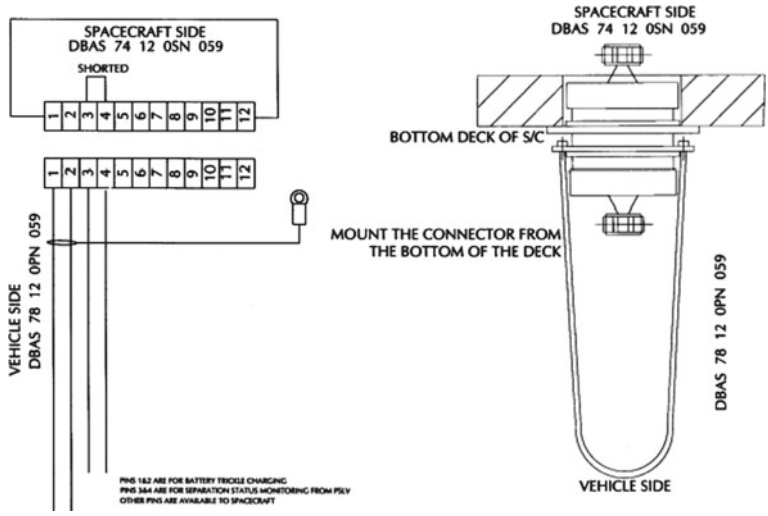


Fig. 11.3 PSLV electrical connector interface [137]

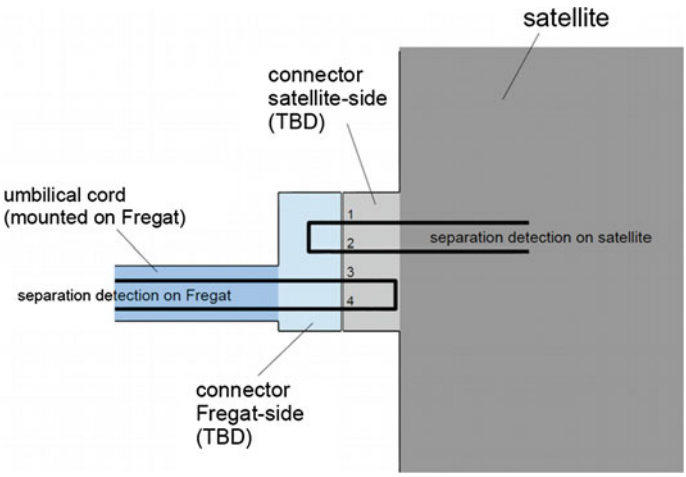


Fig. 11.4 Electrical connector interface for Soyuz Fregat (pins and connectors in multiple instances for redundancy). © IRS, University of Stuttgart

For the 120 kg system the connector location is fixed at 196 mm from the center of the separation system.

The umbilical connector for Soyuz Fregat offers only the separation detection for satellite and launcher. In case of a launch with Soyuz the satellite battery will be charged by means of the external power interface of the skin connector during launch preparation. The disadvantage of this configuration is the lack of possibility for trickle charging during pre-launch and early countdown phase. The connector configuration principle is depicted in Fig. 11.4.

Proper grounding between the various parts of the spacecraft as well as between vehicle and spacecraft shall be ensured to avoid undesirable build up of static charge. The contact resistance between spacecraft and vehicle must be less than 10 mΩ.

11.4 Activities Before Launcher Integration

In this chapter only a short overview on the activities shall be provided, that have to take place after arrival at the launch site. The list does not claim to be entirely complete and will vary depending on the individual FLP mission and satellite instrumentation. Also differences have to be considered between FLP Generation 1 and 2 spacecraft.

- The first activity at the launch site is to unpack the spacecraft and the GSE equipment and to connect the EGSE setup to the spacecraft in the launch site cleanroom.

- Then the spacecraft will be started up in the None Mode where PCDU and OBC are running and spacecraft housekeeping TM can be received and TCs can be submitted. This includes the qualitative test of the S-band transmission system.
- The next step will be rerunning the Abbreviated Function Test (AFT) [38]. This test is a recheck of all electrical signal and power lines with respect to pure electrical characteristics. The AFT is split into a platform AFT and a payload AFT.
  - For the platform AFT the IRS has developed automated MOIS/SCOS test sequences which are reusable independent of a dedicated mission, however presuming the payloads being controlled by a PLOC.
  - IRS also has an automated payload AFT sequence available for the dedicated checkout of the "Flying Laptop" payload module.
  - Furthermore the Data Downlink System (DDS) for the scientific instrument data must be retested.
  - For FLP Generation 1 this is the amateur radio S-band DDS system still included in the payload module. Therefore it is part of the payload module retests.
  - For FLP Generation 2 it will be an X-Band transmission system—Airbus is currently analyzing compatibility of both the SSTL 150 X-Band TTC (CAN-Bus) and of the Myriade X-Band TTC (LVDS).
- Depending on the mission specific payload being mounted on the FLP (cameras, telescope, scanners, multi-spectral imagers or other) it might be necessary to perform alignment rechecks of the payload after transport to the launch site. This is not required for the payload setup of the first FLP based mission "Flying Laptop" but might be for others.
- Similarly for FLP missions with specific optical instruments it might be necessary to perform a final sensor calibration re-check or a final pre-launch sensor re-calibration at launch site.
- Then a final run-in of the RWs at nominal speed over 30 min is performed to homogenize lubricant distribution in the bearings and to minimize stiction effects at start-up and friction differences between the RWLs at speed-up.
- Then a final Battery SoC calibration and maximum battery string capacity reset is performed.
- The following OBSW variable settings are to be checked/set via the JTAG interface:
  - The LEOP-Completion Flag has to be set to 0 (zero) in the safeguard memory of both I/O boards.
  - The Separation-Completion Flag has to be set to 0 (zero) in the safeguard memory of both I/O boards.
  - The SA-Deployed Flag has to be set to 0 (zero) in the safeguard memory of both I/O boards.

- For FLP Generation 2 missions with propulsion subsystem (Xenon resistojet or even hydrazine mono-propellant) the corresponding preparatory procedures before fueling will be performed:
  - Pressurization with pure pressurant gas
  - Valve checks
  - Thruster heater checks (resistojet heater respectively catalytic-bed heaters for the hydrazine thrusters)
  - Purging.

11.5 Pre-launch Activities and Checks

As first step of the pre-launch activity the separation adapter together with the secondary launch passengers will be mounted onto the launcher. Proper electric grounding of the spacecraft to the launcher reference on the adapter has to be assured also during this step. After mounting the following activities will be conducted:

- For the platform each of the four retaining mechanism of the solar panel deployment system are equipped with an activation connector (see annex section on green-tagged items 17.13) closing both electric heater circuits on one mechanism for heating the melting braid (see Fig. 5.5). Missing connectors prevent activation during the integration, test and transport phase.
  As soon as these connectors now are mated, the PCDU is armed and will initiate the solar panel deployment after its next boot-up. Therefore the spacecraft needs to remain switched off from this moment onwards until separation in orbit.
  This implies that no further OBSW diagnosis activities can be performed any more from this moment onwards as the PCDU is in Off mode and so must be the OBC.
- Therefore also the spacecraft separation circuit switch must be closed now.
- The remaining green-tagged items—the safety caps for the spacecraft skin connector Skin Connector Panel—will be applied, except for the trickle charge power connector for charging the battery. This connector will be disconnected as last step of the pre-launch activity. The skin connector panel layout, the connectors and their pin allocation are described in the annex Sect. 17.11.
- For FLP Generation 2 missions with propulsion subsystem (Xenon resistojet or even hydrazine mono-propellant) the final fueling will follow in accordance with the relevant safety measures (especially for hydrazine propulsion system fueling).
- All red-tagged items like payload instrument covers etc. need to be removed as next step.
- For the FLP Generation 1 the external power connector will be unplugged on ground as last activity before the fairing of the launcher is closed and the

connector cover will be mounted (green-tagged item). From this point onwards no further trickle charging nor diagnosis activities can be performed any more.
- The fairing of the launcher is mounted and closed.

11.6 Launch and Early Orbit Phase

Then the launch itself is performed and the launch sequence will follow approximately a scheme as depicted in Fig. 11.5. The estimated launcher separation time and the target orbit data for the launcher's secondary passenger satellites will be provided by the launch supplier some days to hours before lift-off. A second set of more precise orbit data will be provided after separation from the launcher.

The Soyuz launch agent indicated a time span of 2 h between lift-off and separation. After separation it will take approximately two launcher orbits and as result approximately 3–4 h after lift-off until the "Flying Laptop" satellite would be visible for ground stations over Germany, in the morning of local German time. Scenarios with the PSLV look similar.

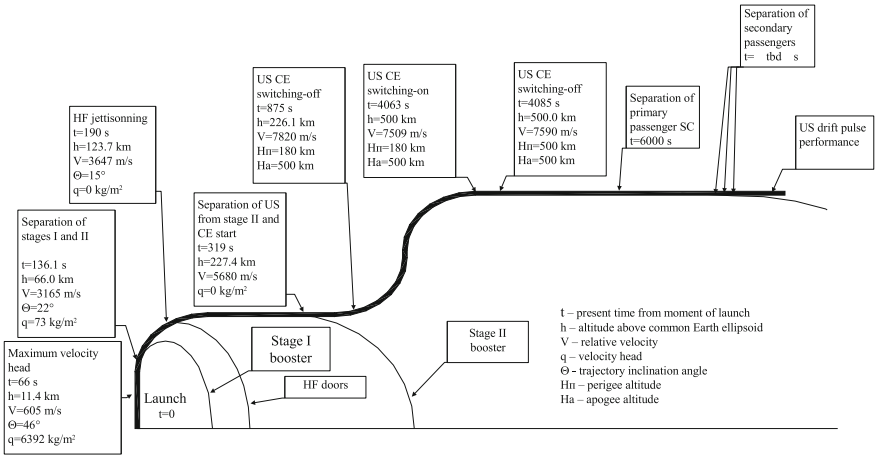


Fig. 11.5 Fictive launch profile—adapted from [140] for secondary passenger example

11.6.1 Spacecraft Power-up to Safe Mode

The PCDU is in Launch/Shutdown mode while the spacecraft is mounted on the launcher and it remains in this mode during launch until before separation of the satellite from the launcher (see Fig. 5.12). In this mode, the PCDU is only in

standby—no satellite devices are powered. The BCRs are active to balance the battery strings. It is possible for the system to re-enter this mode when the battery SoC drops severely.

As explained in Sect. 5.5.3 the initial steps of the spacecraft boot are performed by the PCDU when identifying the separation strap circuit being opened (see also the "micro switch" circuit in Fig. 5.9). After separation detection the PCDU boots into its MiniOps Mode. It activates the power circuits of the OBC and as a result the OBC nominal side with Processor-Board N and I/O-Board N and both hot redundant CCSDS-Boards is booted. After the OBSW is up and running it will command the PCDU into its Idle Mode.

The OBSW then starts S/C operation by setting the satellite automatically into Detumble Mode. This implies also setting the Attitude Control System (ACS) into Detumble Mode which then starts to reduce rotational rates of the satellite resulting from the launcher separation by means of Magnetotorquers, measuring the achieved attitude by means of Sun Sensors and Magnetometers.

The initial rotational rates after separation are induced by the separation mechanism and their maximum values result from the mechanism design. Currently rates of

- 4.5°/s around each satellite main axis are assumed,
- corresponding to a total of 8°/s accumulated over all axes.

Start of detumbling is marked by an Event TM packet for information on ground.

Furthermore, other vital parts of the OBSW such as the controllers for power supply subsystem (PSS), the telemetry and telecommand subsystem (TTC) and the thermal control subsystem (TCS) are set to their standard modes.

During these initial operational steps of the OBSW it generates according boot Event TM (being written to the TM packet store on the I/O-Board) for later downlink (for more details on the boot process see Sect. 4.3). The OBSW furthermore collects all relevant TM of initial rotational rates, the deployment status data from the PCDU and the number of deployment attempts and allocates the data in according system TM packets for downlink.

During this phase the OBSW will also request the status of the SA-Deployed Flag from the PCDU and will update the values accordingly in the OBSW safeguard memory.

Depending on the orbit injection, the position of the first visible ground station and the initial tumbling rates of the spacecraft after launcher separation, it might occur that the Safe Mode is reached before first ground contact or vice versa.

Simulations in [128] showed that detumbling to less than 0.4°/s (condition for the next S/C mode transition) takes approximately 75 min. After detumbling, the system transits to its Safe Mode, in which the solar-array is oriented coarse pointing towards the sun (−Z-axis) and the satellite is spun up around its principal axis to a

- rotational rate of 2°/s
- or one evolution every 3 min.
- Coarse sun pointing assumes a maximum pointing error of 20°.

The begin of Safe Mode entry is logged by the OBSW by a mode transition system Event TM packet. The final acquisition of the Safe Mode with the achieved rotation around the Z-axis and sun pointing, is tracked with a Safe Mode-reached system Event TM packet.

11.6.2 Spacecraft Link Establishment

For the LEOP Operations of the first FLP satellite—the "Flying Laptop"—IRS, Uni-Stuttgart is granted to use the ground Station of the German Space Agency "Deutsches Zentrum für Luft- und Raumfahrt" (DLR) at Weilheim near Munich. For more details on the infrastructure and procedure please refer to Chap. 13. The antenna station in Weilheim is equipped with an automatic scan mechanism to "search" for the carrier signal from the satellite.

With either of the two cited alternative launchers and launch sites the first passover in separated mode is expected several hours after launcher liftoff. The estimate is T_0 +4 h. Therefore for this mission the satellite is expected to be in Safe Mode at ground contact.

- As the OBSW after boot is still in the LEOP time range (see Sect. 6.3.4) it automatically will activate not only the TTC receivers, but also the nominal TTC transmitter.
  Note: *Later during the mission when the LEOP Completion Flag was set to 1, the transmitter activation is done by the TTC_Controller as result of time tagged commands in the timeline or as result to receiver lock detection—no longer automatically after OBC boot.*
- From ground an unmodulated carrier wave (Continuous Wave, CW-Mode) is transmitted towards the expected S/C position shortly before the satellite is expected to appear at this location and carrier frequency sweeping is induced to account for Doppler shift due to the satellite motion towards the ground station. During sweep the point will be reached where the receiver on board will lock its PLL to the identified carrier. The on-board transmitter will then be activated by the OBSW and will transmit its carrier with modulated live TM respectively idle frames back to the ground station.
- Once this signal is received at level of the RF-SCOE in the ground station and the receiver is locked onto the signal, the CW-Mode is disabled in the ground equipment and idle packets are uplinked to the satellite (BD-Mode—see Table 12.1).
- For final bidirectional link verification a ping TC is sent to the satellite and ping TM should be received by the GS.
- The full lock of the satellite receiver can be determined on ground by interpreting the Command Link Control Words (CLCW) being present in every downlinked frame as soon as the communication chain has been commanded to AD-Mode

(see Table 12.1). This mode also permits verification that no TC frames are lost during transmission.
- If this mode is established, a TC burst test with a number of ping TCs should be performed to assure TC link stability and no loss of TC frames.
- During the entire satellite passover, the antenna pointing is continuously adjusted according to the predicted satellite position. No satellite tracking by means of ranging of the signal is performed.
- The DLR antenna station (see Chap. 13) performs pointing by measurements and by this means also determines more accurate orbit data than initially provided by the launch supplier.
- The IRS ground segment (see Chap. 12) relies fully on the orbit data provided by
  - initially the launch supplier
  - then DLR orbit measurements from Weilheim
  - and later by tracking data provided online by NORAD
  - and even later during the Commissioning Phase from the GPS receiver (see Sect. 11.7).

11.6.3 Initial System Checkout

After initial ground contact establishment, the first verification activity is to check for the battery voltage status to estimate energy availability on board.

Second verification is to check the solar array deployment flags indicating proper deployment. For the "Flying Laptop" mission these are the two flags for the left and right deployable panel—for FLP missions with other deployment geometries this might be slightly different. If the satellite is in sunlight and the panels are already sun pointing, i.e. the Safe Mode is already reached, deployment of the panels can also be verified by comparing the voltages and currents of the two wing panels.

If all seems properly deployed the next step is to set the LEOP-Completion Flags to 1 (one) on both I/O-Boards, so that whenever the OBSW is rebooted on whatever OBC board configuration, it will not perform any LEOP specific activities anymore.

From that point onwards the normal scheme of Transmitter operation will be followed, meaning

- Transmitter activation by time-tagged command from the executed on-board timeline or by received ground command.
- Transmitter deactivation by command or at the latest by automatic reaction of the communication controller when the spacecraft did not receive any TC for 20 min.[2]

[2]The time delay is a configurable OBSW variable.

- Automatic transmitter activation when a receiver bit log is detected by the communication controller, i.e. when a ground station transmits a signal on the correct frequency trying to establish radio contact to the spacecraft. This feature is implemented to avoid a blind acquisition when the time-tagged transmitter activation commands fail, e.g. when the on-board command queue is deactivated.

Furthermore it is checked whether the satellite is still in Detumble Mode or whether it already has reached Safe Mode.

11.7 Platform Equipment Commissioning in Orbit

The following phase is the spacecraft Commissioning Phase. Here subsequently all the higher flight mode equipment of the platform and later also of the payloads are taken into operation and are characterized/calibrated in orbit if necessary.

This platform Flight Operations Manual focuses exclusively on the commissioning of the FLP platform equipment. The commissioning steps are the following:

- After the initial ground contact establishment first the TTC transceiver signal characteristics are analyzed. Signal strength is analyzed in relation to the distance and visibility angles to the station and so are signal spectrum and noise.
- As described in Sect. 11.6.1 the spacecraft transits through a Detumble Mode to a stable Safe Mode where it is Sun oriented and spinning. During commissioning here the avionics equipment TM is analyzed whether
  - the requested Mode has been achieved (Safe Mode),
  - the spin rate is approximately 2°/s
  - for all magnetotorquers both coils are operating properly
  - magnetometers and Sun sensors are qualitatively providing plausible data
  - the system power bus voltage is in a tolerable range and the PCDU indicates battery charging during Sun phase.

These checks can be applied directly based on flight TM from the first ground station passover(s). Thereafter some activities follow which are partly based on pure TM but which also imply some commanding. These concern the power subsystem —namely the battery.

- When the satellite is flying in Safe Mode the battery charge status and the power-bus voltage should increase from orbit to orbit until the battery is fully charged. This has to be monitored by the relevant TM parameters.
- Once the battery is fully charged, the SoC estimation algorithm calibrates itself to 100 % SoC (see Sect. 5.4.1.2). As the initial SoC value after launch can only be guessed, the battery must be fully charged before the SoC estimation can be considered to be reliable. Event TM from the power controller indicate the calibration of the SoC algorithm for all three battery strings.

- From this point on, the SoC calculates the SoC reliably based on the configuration parameters set before launch, which are derived from ground test data of the battery FM.

The equipment power-up is foreseen to be performed in the following sequence:

- As soon as possible the GPS receiver shall be taken into operation as it takes quite some time to acquire the first navigation solution at initial boot-up (approx. 20 min). See Sect. 7.11.2.4 for details.
- Furthermore the GPS receiver data provide a better estimation of the actual orbit characteristics than the initial launch supplier injection data and the subsequent analysis data gained by ground measurements from antenna stations.
- The GPS receiver HK telemetry has to be analyzed for consistency and plausibility and for showing no failure events.
- The star trackers and the fiberoptic gyros are to be powered up next.
- For both devices health relevant TM and again Event TM has to be checked.
- For the STRs the plausibility check of the reported quaternions against the spacecraft orientation taken from the Sun sensors together with GPS-based spacecraft position has to be performed.
- For the FOGs a qualitative check of the rotational rate and axis reported by the FOGs versus Sun-oriented spacecraft spin axis has to be made. The actual spin rate can be deduced from receiver signal strength TM during ground contact, as the TTC antenna gain varies with the spacecraft aspect angle.

The next major step before the transition to any higher mode is to take the reaction wheels into operation.

- As it is very delicate to verify the RW polarity on ground during AIT a dedicated satellite mode is foreseen for in-orbit test activities that require a maximum use of the ground station contact time during passover. This is the Coarse Nadir Pointing Mode. This mode—which is an RW-free nadir pointing mode purely controlled via Magnetotorquers—is to be activated and it will take 2–3 orbits until the satellite has properly switched from Safe Mode Sun-pointing spin to stable nadir pointing.
- The platform is designed that although the solar arrays in this mode are only illuminated optimally during a small part of the orbit, the battery is not discharged in total. The energy budget over an orbit and over multiple orbits is positive.
- When Coarse Nadir Pointing Mode is reached and ground contact for an optimum passover is achieved, the RWs are commanded ON by one command, all at the same time and all with a limited, moderate rotational rate of 300 rpm.
- Due to all RWs starting up at the same time—under the assumption of correct polarity matrix in the OBSW—the stiction effects at startup of the overall RW assembly should almost level themselves out. The spacecraft attitude should only show a slight “shake” effect.
- Due to all RWs starting up with very limited rotational rate and rotational torque, in case of a polarity matrix error in the OBSW the spacecraft

counter-rotation would be visible as constant increasing pointing deviation. However the induced tumbling rate would be limited and as the maneuver is monitored by ground operators in real-time, it can be emergency stopped simply by immediate electrical power-down of the 4 RWs. In such a case the error in the RW polarity has to be carefully analyzed by ground based on the acquired telemetry.

- If all RWs have started up without problems to the desired 300 rpm, the RWL assembly can now be commanded to fully speed up the RWs to nominal speed and to take them into operation.

From this point onwards the spacecraft can be commanded to Idle Mode and to the higher ACS modes to check out control algorithm performance in orbit under life conditions.

Any occurring negative commissioning results require corresponding ground based failure analysis and potential further TM downlink—optionally by activation of further TM packets of the affected subsystem/equipment.

11.8 Nominal Operation Phase

This phase represents the operational phase in which the satellite actually performs scientific experiments and technology demonstrations which need special arrangements and planning are conducted in this phase.

The phase comprises all nominal operational modes of the satellite. It will be in Idle Mode if no instrument observation activity or avionics based experiment is running. Instrument based observations of ground targets, near Earth objects, the Moon or other celestial bodies will be performed in higher operational modes than Idle Mode, together with the corresponding ACS mode for pointing.

Observations planning will be performed by use of a mission planning infrastructure. Chapter 14 outlines a mission planning infrastructure example as installed at the IRS, University of Stuttgart and the usage of its infrastructure.

Instrument and platform operations TCs are integrated into the generated timelines and are uplinked to the spacecraft for execution over multiple orbits. Science data recorded by the mission instruments are downlinked to the science data antenna station. The science data reception and storage chain applied by the IRS, University of Stuttgart is outlined in Sect. 12.4.

11.9 End of Life Phase

In order to comply to the European Code of Conduct on Space Debris Mitigation [1] already the FLP Generation 1 is equipped with a De-Orbiting-Mechanism (DOM), which is developed and built by the Tohoku University, Japan. The DOM

will deploy a 1.5 × 1.5 m^2 Kapton sail in order to decelerate the satellite by increasing the satellites aerodrag. The deployment of the sail is achieved with the help of a leaf spring mechanism which is deployed by a melting braid device.

With this additional aerodrag the satellite will de-orbit in 25 years or less if the initial orbit altitude does not exceed 650 km.

FLP Generation 2 based missions may also use higher orbit altitudes as they provide a propulsion subsystem. For deorbiting at end of life phase sufficient remaining fuel must be provided for at mission design time. FLP Generation 2 missions also can use both, the propulsion and the DOM for orbit altitude decrease.

End of life phase is reached in three cases:

- In the first case the satellite has reached or exceeded its nominal mission lifetime and equipment is subsequently degrading such as battery or non-radiation hard electronics, payload instrument or normal avionics units with limited lifetime like the FOGs.
  - In such a case ground defines the end of the mission and initiates the deployment of the De-Orbiting-Mechanism (DOM) which will be released by the PCDU.
  - For control of the DOM release the PCDU firmware contains an internal DOM deployment counter which is starting to count down one tick per second as soon as the PCDU is booted to Mini-Ops mode or higher. The start value of the counter is set to 4096 h which corresponds to approximately 170 days or 5.5 months.
  - During nominal operations, the DOM timer is reset to the initial value once a day by command from ground to avoid triggering deployment of the DOM. Note that there is also a HPC defined to reset the DOM timer, which can be used in case of an OBSW failure, which does not constitute the end of the mission.
  - If ground decides that the mission has ended, deployment of the DOM can be initiated by activating the respective switches in the PCDU. Note that the DOM timer cannot be set to a value below 2000 h for safety reasons. Such values can only be reached by countdown.
- Mission end can also be reached in case where a fatal error happened on board. In this case the satellite system is disposed and no further operations can be performed.
- Another fatal error definitely will have occurred in case where no contact to the satellite can be established from ground over several months, regardless of the actual root cause of the problem. In this case the PCDU internal counter will not be reset by ground and will reach zero approximately 5.5 months after the last reset. The DOM will be deployed automatically.

Chapter 12
Stuttgart Mission Control Infrastructure

Jens Eickhoff, Nico Bucher, Maximilian Böttcher, Charles Thibaut, Dougie Johnman and Bryan Tatman

Abstract The chapter provides a detailed insight on the mission control infrastructure installed at the University of Stuttgart, Germany, including the antenna system and the science data signal chain.

J. Eickhoff (✉)
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

J. Eickhoff · N. Bucher · M. Böttcher · C. Thibaut
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: bucher@irs.uni-stuttgart.de

M. Böttcher
e-mail: boettcher@irs.uni-stuttgart.de

C. Thibaut
e-mail: thibaut@irs.uni-stuttgart.de

D. Johnman · B. Tatman
Satellite Services B.V., Noordwijk, The Netherlands
e-mail: d.johnman@ssbv.com

B. Tatman
e-mail: b.tatman@ssbv.com

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_12

Keywords Mission control infrastructure · SCOS-2000 · RF signal chain · Antenna system · Flight dynamics infrastructure · Science data signal chain

The operation of IRS FLP missions will be conducted primarily through the main ground station located at the University in Stuttgart at 48° 44′ 58.19″N and 9° 06′ 13.84″E, approximately 500 m above sea level.

The facility is fully equipped with a satellite control room, serving as the interface for the operators, and an antenna station and is described in the frame of this chapter.

In addition to the Stuttgart ground station the IRS team has access to the antenna station of the German Space Agency "Deutsches Zentrum für Luft- und Raumfahrt" (DLR)—operated by the DLR German Space Operations Center (GSOC). This antenna station is located in Weilheim near Munich, Germany. The DLR/GSOC antenna station will be used in the LEOP, Commissioning Phase and during potential contingency situations within normal spacecraft operations phases. In both cases—with the Weilheim antenna infrastructure or the IRS own antenna station—the operations platform in the Ground Control Center (GCC) at IRS is the same. Just the routing destination changes. More details on the technical interface and the use process of the DLR/GSOC infrastructure is provided Chap. 13.

12.1 Platform Control Infrastructure

The ground station on the Stuttgart university campus was specifically implemented for the operation of IRS and partner university satellites. Satellites can be monitored and commanded by means of the ground station either via a link established by its own antenna or by an external antenna station. The ground station has facilities for satellite control and science data downlink. The platform ground control infrastructure comprises the satellite control system infrastructure, the control room environment, the network link to the antenna system, the transmission of high-frequency signals and the antenna itself on the roof of an institute building on the university campus.

The institute's Ground Control Center is equipped with the Satellite Control and Operation System 2000 (SCOS-2000), the standard Mission Control System (MCS) used and maintained by ESA/ESOC. The primary SCOS-2000 server is operated by the spacecraft controller. It is connected to a TM/TC routing system which is extendable to interface a range of external ground stations (see Chap. 13).

The SCOS software provides the operator with the interface to the spacecraft, enabling commanding, telemetry reception and monitoring of telemery. Telecommands and telemetry are exchanged with the Radio Frequency Special Check-Out Equipment (RF-SCOE) which converts the data to the transmittable Command Link Transmission Units (CLTU). SCOS-2000 runs on a dedicated IRS server to enable multiple operators to log in via clients and to perform different telemetry monitoring and telecommanding activities simultaneously-see Fig. 12.1.

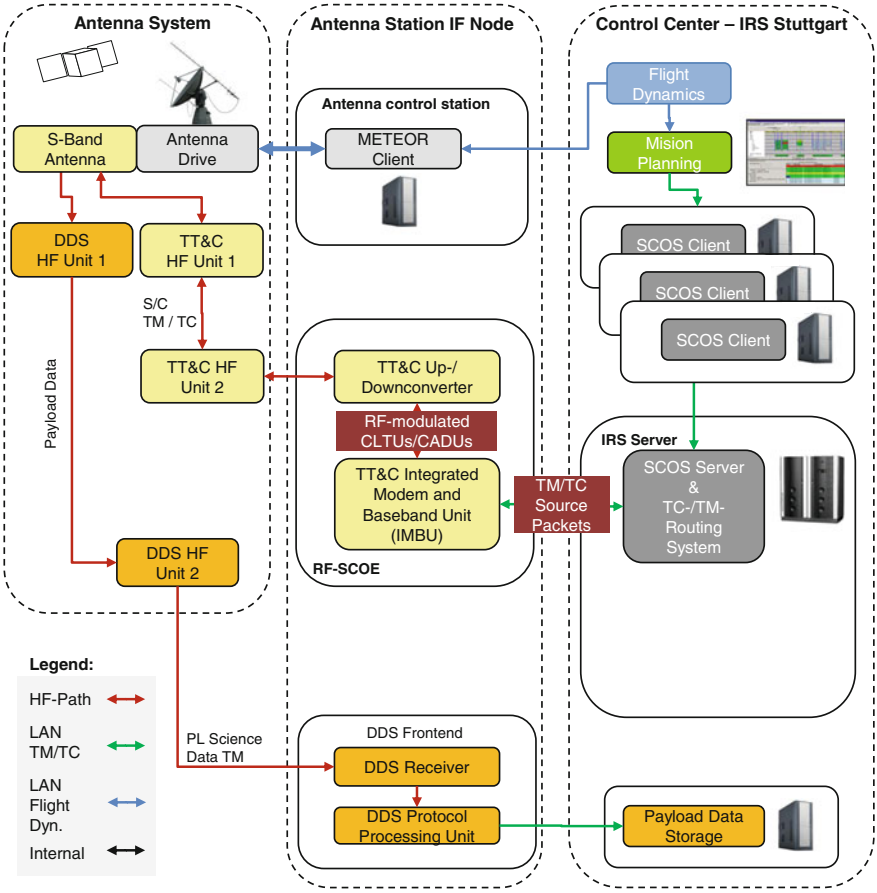


Fig. 12.1 IRS ground control infrastructure. © IRS, University of Stuttgart

While SCOS-2000 is suitable for the compilation of fixed sequences of TCs an additional software system, the Mission Operations Information System (MOIS), is used to control entire mission Flight Procedures—see also Chap. 15. MOIS is a high-level tool which interfaces with SCOS to provide enhanced capabilities for the automation of mission sequences. It also provides a full suite of tools for mission planning, Flight Procedure definition and debugging These infrastructure aspects are covered later in more detail in Chap. 14.

The SCOS system interfaces to the RF infrastructure via equipment units which have already been used during system tests of the first FLP satellite, the so-called RF Special Checkout Equipment (RF-SCOE). The RF-SCOE handles all the CCSDS protocol layers and converts the packets up to the Command Link Transmission Units (CLTU) and in the downlink path from the Channel Acquisition Data Units (CADU) down to packet level. The RF-SCOE together with

the attached Up-Converter and Down-Converter also provides the functions of TC signal modulation and TM signal demodulation to/from the carrier frequencies. More details are provided further below in Sect. 12.1.2.

Finally the antenna system comprises the antenna itself, the separation of spacecraft TC/TM signal from the payload data downlink signal and all filters, amplifiers etc.

Furthermore the antenna station interface node is also used for the accommodation of hardware, required for the control of the antenna drive mechanism. Optical fiber cables provide the data links between the antenna base and the antenna station interface node. Antenna pointing is controlled by means of the software METEOR, an IRS in-house development. The antenna system comprises a parabolic antenna with a diameter of 2.5 m, an antenna rotor and an antenna base.

12.1.1 Digital TM/TC Processing Components

12.1.1.1 Ground Control Software SCOS

As already mentioned, the IRS uses a SCOS-2000 system as mission control system. The IRS was supplied with a system license from ESA/ESOC in 2004 when it started its satellite simulation activities. SCOS is ESA's generic mission control system software and is also used by other S/C operators and agencies such as by DLR/GSOC. The version applied by IRS is a V3.1, which is already somehow

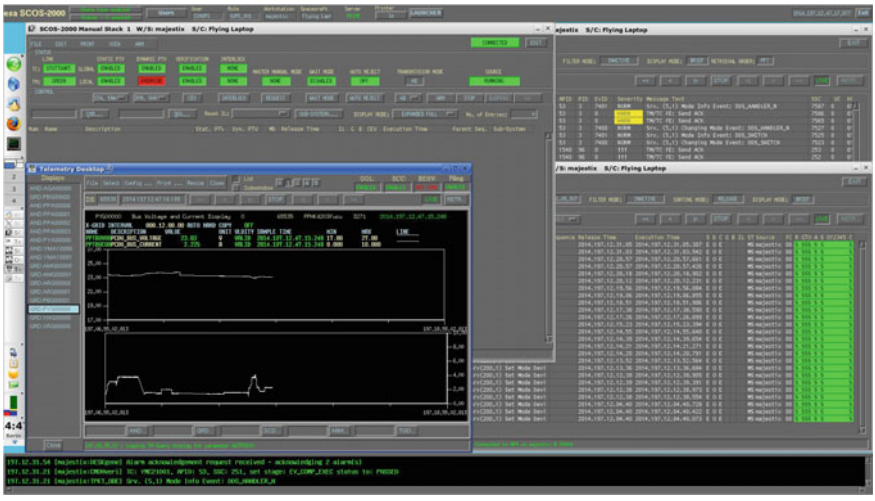


Fig. 12.2 SCOS command log, HK TM log and parameter plot window example. © IRS, University of Stuttgart

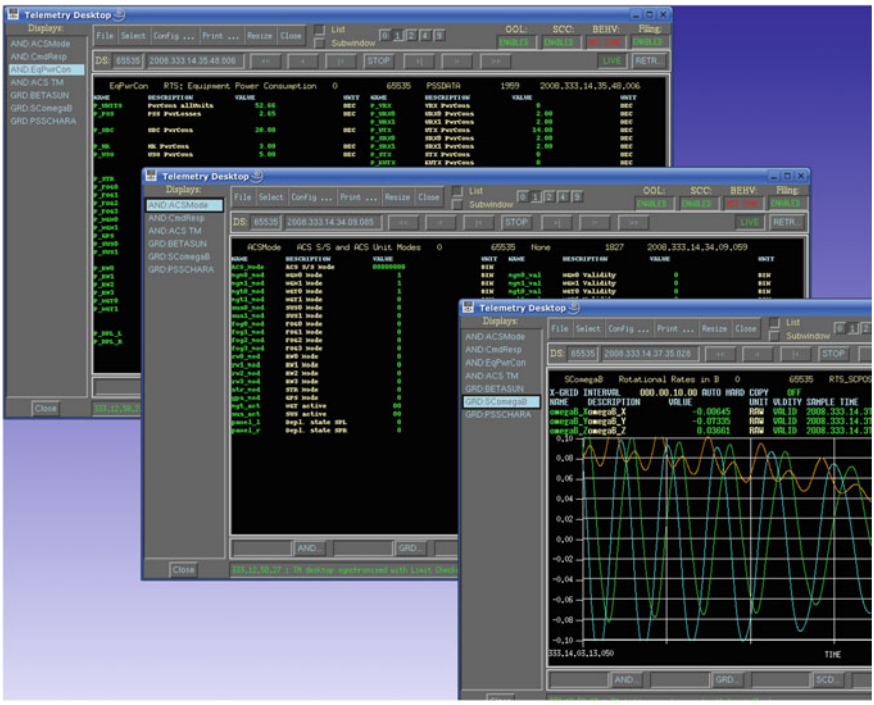


Fig. 12.3 Diverse SCOS windows (from AOCS test on simulator). © IRS, University of Stuttgart

outdated but is kept currently for the first FLP based satellite and will be upgraded for subsequent missions. The recent SCOS versions are the V5 series.

SCOS is based on a database which contains the definition of all satellite telecommands, all telemetry and all TC and TM variables from the OBSW *datapool* which are included in TC packets or TM packets. This database is called the Mission Information Base (MIB) and the FLP MIB is treated in Chap. 16. For raw variables the MIB also includes the calibration information to convert raw data to engineering units for visualization on the ground control consoles.

Example visual impressions from the SCOS Man-Machine Interface are provided in Figs. 12.2 and 12.3. They show both TC/TM log windows as well as graphic parameter displays—so-called synoptic displays.

The SCOS TC chain is based on the CCSDS standard. SCOS can submit TC packets which are framed externally or can assemble directly CCSDS TC frames, TC segments or even CLTUs. Command stacks can be loaded either manually or automatically, and are validated against predefined constraints prior to release.

The SCOS TM chain for telemetry reception expects its input on TM frame level or TM packet level. Please refer to Figs. 12.1 and 13.2 which explain the CCSDS layer levels used for the implementation in the IRS ground station or in case where an external partner infrastructure is used for S/C control.

In addition the system provides a TC/TM packet and parameters archive functionality to track entire operational sessions—e.g. ground station flyovers of the satellite or satellite test sessions during AIT phase.

To ease the command and control of the FLP platform for the ground staff two layers of abstraction are introduced:

- OBSW functions are available which can be triggered/activated from ground via the already cited PUS Service 8 (Function management service).
  An example can be a function for activation of a payload from ground where the OBSW executes the detailed steps from power supply switch via payload controller boot control, initial PL onboard data-bus TM verification, power consumption control etc.
  The FLP platform Gen. 1 does not provide any On-board Control Procedures (OBCP) as defined in [13]. This functionality is replaced by PUS Service 202 (see Sect. 2.4.17).
- Flight Procedures (FP) are another means for increasing the level of commanding. Flight Procedures are somewhat the complement to OBCPs. While an OBCP is a sort of "command script" executed on board, a Flight Procedure is a "command script" implemented in the ground control system.
  An example is a Flight Procedure which submits the function commands for the AOCS to switch from idle mode to fine pointing, for the data handling subsystem (MMU etc.) to prepare for science data recording and for the payload to switch on—all for preparation of a payload instrument measurement on board.
  Flight Procedures can comprise both low level commands to S/C units, higher level commands to S/C subsystems and system level commands and they can trigger onboard functions and OBCPs.
  Any command defined in the MIB (and thus implemented in the OBSW) can be included in an FP. Both individual commands and entire FPs can be commanded from a spacecraft ground control console.
  In general FPs allow the definition of

– absolute and relative time tags for the individual commands,
– command flow IF/Then branching according to "return values" received via TM back from the S/C during procedure execution—provided ground contact exists and
– Do/While loop constructs—as far as supported by the procedure execution engine in the ground control system.

The execution, i.e. the subsequent submission of such Flight Procedure command sequences and the according branching in IF/Then cases requires more than just a simple play-list of commands. It requires the commands being embedded into a script language and the execution of such scripts by the ground control console. Thus an according procedure execution engine has to be coupled to (or must be integrated into) the ground control system. For SCOS several script languages and

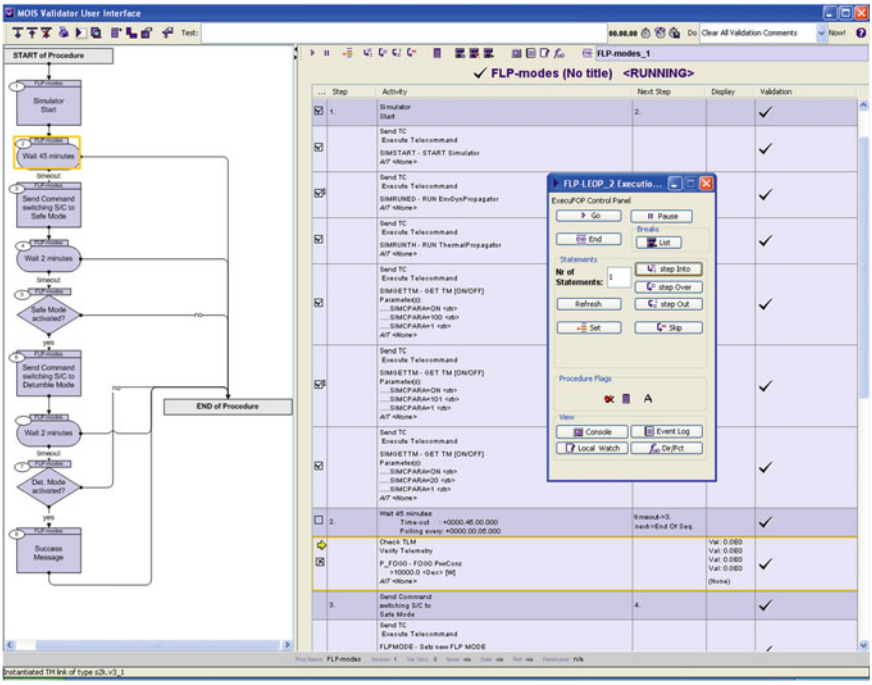


Fig. 12.4 Execution of flight procedures with the MOIS executor/validator. © RHEA system S.A. and IRS, University of Stuttgart

execution engines are available, one for the older TCL language (cf. [106]), and one for the newer PLUTO language which is standardized by the ECSS (cf. [107]) and is applied in modern ESA missions.

To avoid writing such procedures via a text editor and inducing errors via typos etc. FPs are today defined by means of flow chart editors. They provide different views on the task flow and offer the operator to select commands and according parameters as they are defined in the MIB. The IRS is equipped with an entire tool suite from RHEA System S.A. in Belgium, which is called The Manufacturing and Operations Information System (MOIS). The entire MOIS suite is presented in more detail in Sect. 14.1. Here only the two main elements for the Flight Procedure context shall be noted.

- One is the procedure editor—the "MOIS Writer".
- The other one is the procedure test tool—called "MOIS Executor/Validator"—as depicted in Fig. 12.4. In the IRS infrastructure the MOIS Executor replaces the cited SCOS script language engine for TCL or PLUTO.

All Flight Procedures as cited in Sect. 17.4 are defined by means of the MOIS Writer. During their execution the MOIS Executor stepwise submits S/C commands defined per procedure step to the SCOS TC chain and evaluates TM variables if required to branch If/Then or Do/While procedure statements accordingly.

12.1.1.2 NCTRS Router

The Network Control and Telemetry Routing System (NCTRS) originally is a software tool from ESA/ESOC to interface from their control center SCOS systems with remotely connected satellites—either via data link (in most cases X25) to the antenna stations—or via data link to a satellite still under test in the assembly hall of the manufacturer. The system also optionally performs protocol level conversions from e.g. SCOS TC packets to CLTUs and similarly for TM.

The "NCTRS Router" at IRS performs similar routing tasks, but without automatic telecommand or telemetry layer conversions. And it routes only inside the IRS ground segment LAN. Nevertheless the router—embedded in the block "SCOS Server and TC/TM Routing System" of Fig. 12.1—is a key element of the infrastructure as it allows dynamic allocation of control stations and targets.

For understanding the architecture it first must be explained, that the SCOS server and the clients all run on virtual machines in the IRS environment. A SCOS client—running in the Ground Control Center on whatever lab PC then can connect—via the SCOS server—to a target which can be

- the TM/TC Frontend of the simulator testbench to command a test with the simulated spacecraft in the laboratory,
- the TM/TC Frontend of a FlatSat setup of spacecraft equipment in the IRS clean room during spacecraft EM or FM assembly,
- the TM/TC Frontend or RF-SCOE of a complete satellite under test in the IRS clean room,
- the RF chain equipment routing TC and TM to/from the IRS antenna,
- or the connection to an SLE unit for connection to an external agency antenna station (see later in Chap. 13).

All these variants are simply configurable by router software commands. In case of commanding the real satellite from SCOS by means of the IRS antenna station, the router connects the SCOS to the RF chain via the so-called RF-SCOE—see Fig. 12.1. This RF-SCOE was already mentioned in Sect. 12.1 and is explained in more detail in the following Sect. 12.1.2.

Figure 13.2 depicts the route from SCOS to the DLR/GSOC ground station implemented by route connection from SCOS to the DLR provided SLE equipment.

12.1.2 RF Signal Processing

The RF-Signal processing is performed by an infrastructure, which was used as RF-SCOE during satellite test phase in the cleanroom and which is now used for conversion of the digital CCSDS data from SCOS to CLTUs modulated onto a carrier and vice versa to demodulate CADUs from the downlink carrier and to decode CCSDS input for the SCOS system. This infrastructure is provided by a

close cooperation partner of the IRS, namely Satellite Services BV in Noordwijk aan Zee, Netherlands.

The TTC-IMBU system—as it is called by SSBV—uses a "Hybrid Modem" approach which optimizes the combination of Modem and TM/TC-Frontend functionality within one system and allows the system to be used both during AIT and operationally. The Integrated Modem and Baseband Unit (IMBU) provides interfaces both at 70 MHz Intermediate Frequency (IF) and via Bypass Interfaces (RS422/TTL) with integrated Baseband functions. These functions include baseband uplink coding, modulation together with downlink demodulation, decoding/correction.

The TTC-IMBU is based on the latest single-board IMBU product from SSBV that allows all these functions to be integrated into a single 3U/19″ rack mountable unit. This is further enhanced for the "Hybrid Modem" approach by augmenting the TTC-IMBU with a 1U server that hosts the TTC Modem Control and Monitoring software with the TM/TC-Frontend software. The software layers provide data processing up to the packet level and the routing of the TM and TC data to third party systems using network interfaces.

The TTC-IMBU can be controlled and monitored locally or remotely via LAN and for testing purposes also supports different modes of operation supporting TM simulation/coding/modulation as well as TC internal baseband loopback decoding for specific test purposes.

The TTC-IMBU interfaces at an intermediate frequency (IF) of 70 MHz at one end and via LAN to control/monitoring and data processing systems on the other end. The TTC-IMBU also allows TC to be sent and TM to be acquired via bypass interfaces that can be used during spacecraft assembly, integration and testing (AIT).

In summary, the main functions/features of the TTC-IMBU are:

- RF interfacing at 70 MHz intermediate frequency (IF) namely:
  - Two (identical) TC outputs used
  - Two separate TM inputs (LH/RH)
- Bypass interfacing via RS422 (outputs can also be used single-ended)
- Selection of either Left Hand Circular Polarized (LHCP) signals or Right Hand Circular Polarized (RHCP) signals.
- Baseband processing of TM (data decoding, frame synchronization, Reed-Solomon and Convolutional decoding/correction, time- and status tagging)
- Remote Interfacing via LAN for control, status and FTP control/configuration and file transfer
- TM/TC interfacing via LAN to external equipment
- Maintaining local station time in hardware and software through NTP for software and external
- 10 MHz reference, IRIG-B & PPS for hardware

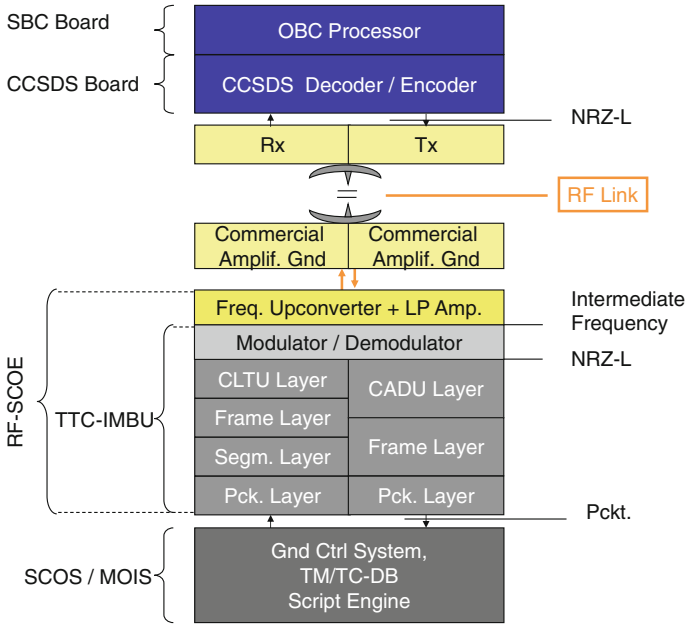


Fig. 12.5 TTC-IMBU in the ground—space TC/TM chain. © IRS, University of Stuttgart

- Providing Graphical User Interface (GUI) for local operations based on the existing SSBV CMS software
- Supporting TM and TC test/simulation functions (as a specific mode of operation) for equipment and (sub)system testing
- Supporting long life-time, high reliability and maintainability

As already explained the equipment first was used as RF-SCOE during satellite tests and now is used as the core of the satellite "ground station"—please also refer to Fig. 12.5.

To fulfill the various scenario needs, a set of EGSE elements have been defined based around the SSBV TTC-IMBU. The TTC-IMBU combines the functionality of a TM/TC-Frontend with that of a modem. The TM/TC baseband section is responsible for the processing of telecommand data down to coded bit stream level and likewise the acquisition of telemetry serial bit streams and the recovery of telemetry data from this.

The modem section is responsible for the modulation of the telecommand signals and the demodulation of the telemetry. Telemetry and telecommand processing is available to/from different layers. The TMTC-IMBU processes the data to packet level (see Fig. 12.6).

The TTC-IMBU interfaces to the spacecraft either via baseband (Bypass) interfaces or via 70 MHz IF. The Bypass interfaces are NRZ-L Data+Clock serial interfaces (the TC Link is augmented by an Enable signal) at either RS422 or LVDS (configured at time of manufacture). The system is shown diagrammatically in Fig. 12.7:

Fig. 12.6 TTC-IMBU and hybrid modem workstation © Satellite Services BV

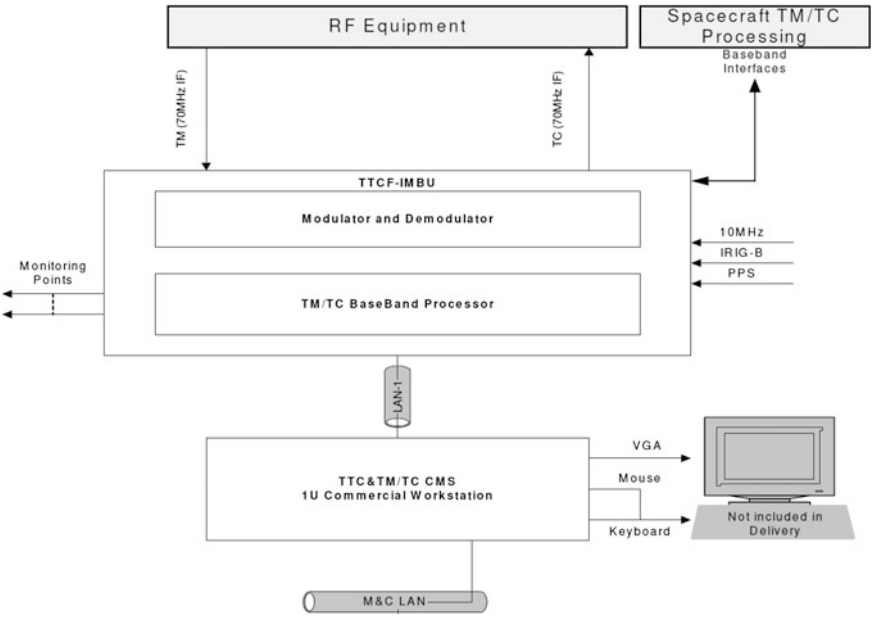


Fig. 12.7 TTC IMBU primary interfaces. © Satellite Services BV

Within the overall environment, several additional elements could be used in conjunction with the TTC-IMBU—see Fig. 12.8.

The following operational elements can be identified:

- TTC-IMBU and 1U commercial workstation
- Time Reference (typically a GPS based timeserver with IRIG-B, PPS, 10 MHz outputs and NTP server)

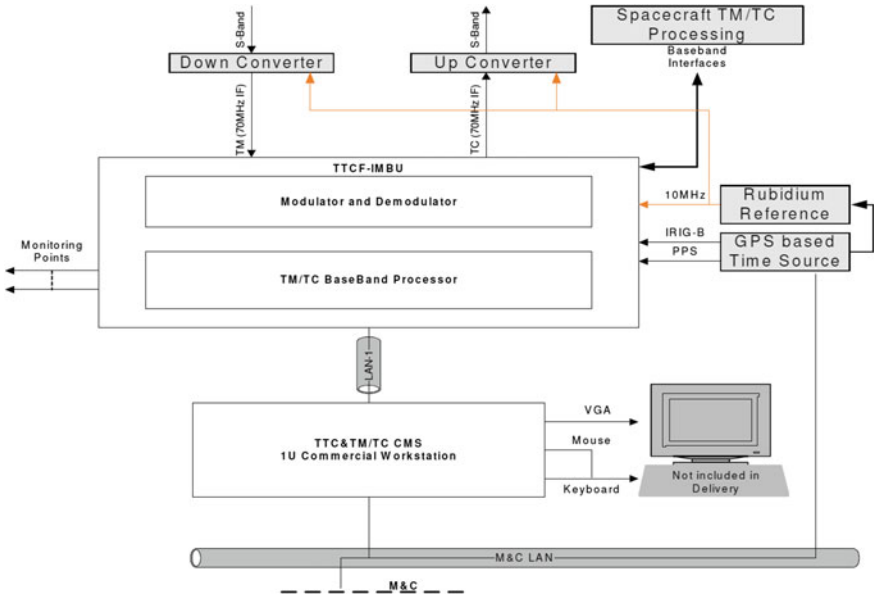


Fig. 12.8 RF SCOE EGSE elements. © Satellite Services BV

- Clock Reference (typically a low phase noise, high stability reference slaved to GPS time server
- Up-Converter (70 MHz IF to S-band) slaved to 10 MHz reference
- Down-Converter (S-band to 70 MHz IF) slaved to 10 MHz reference

It should be noted that the workstation can be connected to an external keyboard, monitor and mouse if required. Alternatively the GUI may be accessed removed using VNC or similar. The TTC-IMBU is monitored and controlled via LAN. In addition TM and TC data is passed via this LAN. The remote interface uses the C&C LAN protocol. The TTC-IMBU is implemented as a compact, single-unit system with external workstation, and as such contains all interfaces and functions that are supported by the system.

12.1.2.1 TTC-IMBU Lower-Level Block Diagram

On the basis of the high-level block presented in Fig. 12.8, a more detailed internal block diagram of the unit that identifies the major module level elements has been composed. This diagram is depicted in Fig. 12.9.

The following module level blocks can be distinguished:

- TTC-IMBU module (SE-1544):
  The SE-1544 module is the basis for the complete implementation of the modem and baseband functions of the TTC-IMBU.

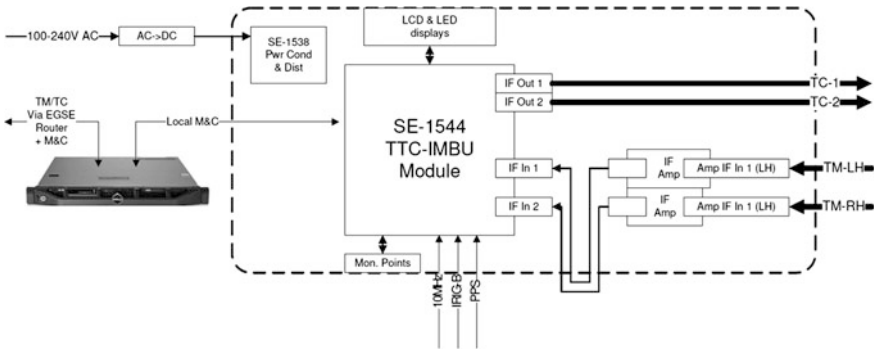


Fig. 12.9 TTC-IMBU lower level block diagram. © Satellite Services BV

The SE-1544 module features a flexible and expandable hardware/firmware architecture to accommodate future in-the-field changes/expansions if these would be required.

As part of the on-board interfaces offered, two switchable 70 MHz IF outputs (carrying the same IF signal) and two separate 70 MHz IF inputs (TM LH/RH) are present. These are complimented with a number of video and baseband test points/connectors to allow easy system level testing and diagnostics.

The SE-1544 module hardware directly drives the LED and LCD display that are present on the front-panel of the TTC-IMBU and also supports an on-board LAN interface for internal communications and in-field (or remote) support activities.

To minimize noise and other EMC disturbances and to improve long term support, the SE-1544 module is equipped with a DC power connector that provides the module with the required range of voltages to operate the interface logic, FPGAs and other electronic components.

The different DC voltages are provided from the SE-1538 Power Conditioning and Distribution Module (PCDM), which in turn takes its power from an external AC –> DC power source.

- IF Input Amplifiers:
  For some applications, it is desirable to lower the input level for the two IF inputs. The TTC-IMBU supports this through the use of two fixed level IF amplifiers (e.g. 30 dB) that can be incorporated within the IMBU housing. These allow the IF signal from external equipment to be amplified (e.g. by 30 dB) prior to insertion into the SE1544 LHCP and RHCP IF inputs. The amplifier inputs and outputs are available on the back panel of the IMBU. If these are included, loop-back cables are pre-installed on these to link the amplifier outputs to the SE1544 IF Inputs.
- User Workstation:
  The 1U User Workstation's primary role is to implement the LAN based interfaces to the M&C and SLE source/sink and to offer a local Graphical User Interface (GUI).

The outside world interfaces that are supported by the workstation include: 2*Ethernet Gigabit LAN, video and USB. One of the Gigabit LAN connections is used for local M&C link to the SE-1544 module.

- Power Supply and SE-1538 PCDM:
  As mentioned above, for performance and RAMS/maintenance reasons, the TTC-IMBU makes use of an external AC <> DC PSU concept.
  For this reason, an industrial grade external AC/DC converter is used to perform the mains AC to DC conversion. The resulting DC power (24 V–100 W max.) is provided to the power connector on the back of the TTC-IMBU.
  The AC/DC adapter provides basic power protection as well as over current and short-circuit protection. Inside the TTC-IMBU, a Power Conditioning and Distribution Module (SE-1538) is used to convert the 24 V DC to the different voltages required for the SE-1544.
- LCD & LEDs:
  On the front of the unit, two LED displays (one for TM-RX, one for TC-TX) and an illuminated button controlled LCD display are present. These modules are controlled directly from the SE-1544 hardware and provide real-time feedback as to the status, operation and configuration of the overall unit.
- Monitoring Points:
  To aid in the monitoring of selected internal and TM/TC baseband signals, front-panel digital monitoring points are implemented. The buffered monitoring points are provided through BNC connectors and are based on a standard 50 Ohm output impedance circuit to allow easy connection of standard T&M equipment. These signals provide TTL level signals when not loaded.
  The selection of monitoring points can be done locally using the control buttons on the front- panel (with feedback through the LCD). The monitoring point selection is flexible and implemented by the FPGAs on the SE-1544 module.

12.1.2.2 TTC-IMBU Operation

The TTC-IMBU is primarily intended for the reception of telemetry and the transmission of telecommands. As such, the GUI includes a status indication for the Downlink Demodulator and Uplink Modulator. In addition however, the system includes test modes for both Telemetry and Telecommand links.

The TTC-IMBU not only allows connection to the spacecraft via the RF links but also via the “Bypass Links”. The Bypass links provide direct serial RS422 connections to the spacecraft during AIT and during launch preparations.

The PM mode uses a single modulator that can be reconfigured to use either TC or TM. The PM Demodulator consists of a single demodulator however this is only available for TM demodulation.

A simplified block diagram of the key functions is shown in Fig. 12.10.

The system implementation uses routing blocks to allow the various signals to be routed between the TMA, TMS, TCG, TCA, Modulator, Demodulator and the

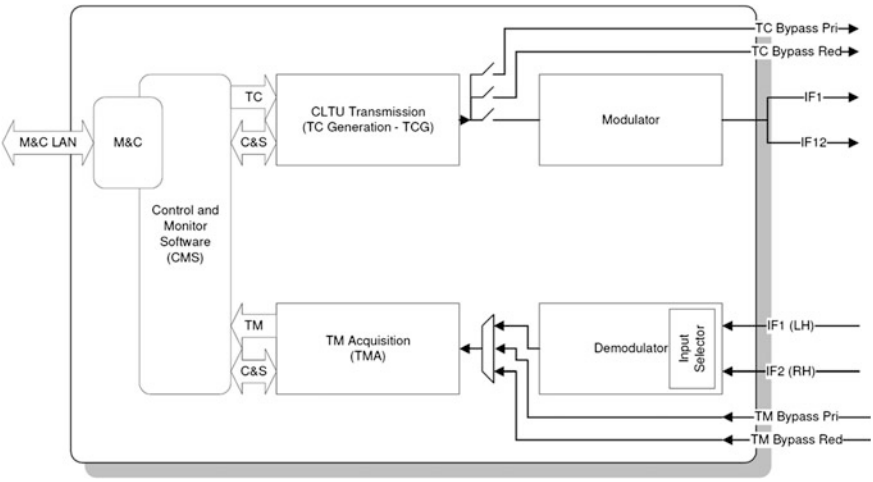


Fig. 12.10 Simplified block diagram in normal operation. © Satellite Services BV

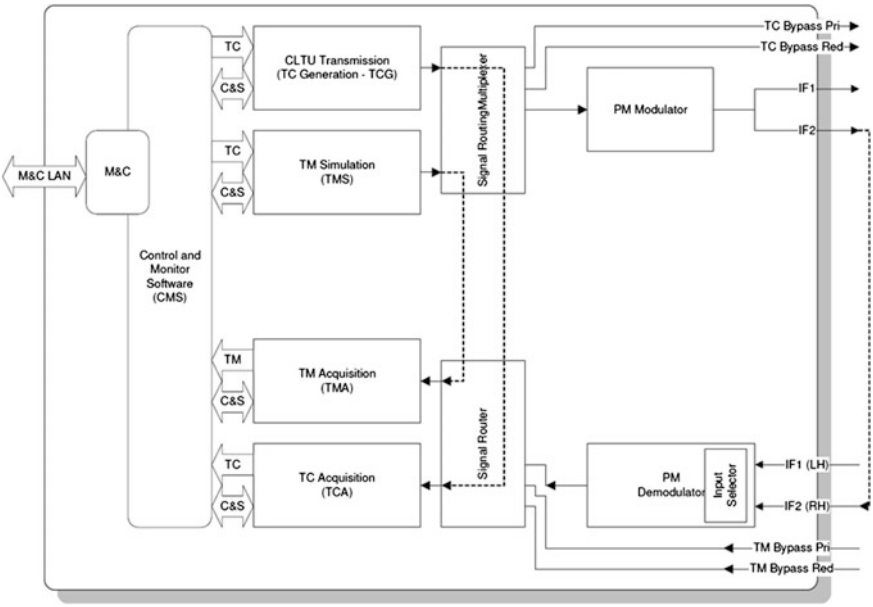


Fig. 12.11 Simplified block diagram in simulation operation. © Satellite Services BV

bypass interfaces. This is shown diagrammatically in Fig. 12.11 (dotted lines depict commonly used loop-back paths):

In addition to the nominal and loop-back test functions of the TTC-IMBU, the system includes standard Bit Error Rate (BER) Testers. The BER function operates

by using a transmitter block and a separated receiver block to send and receive Pseudo Random Bit Sequences (PRBS). Both the TCG and TMS blocks contain BER transmission functions while the TMA and TCA contain paths to BER PRBS receiver/checkers.

The Hybrid TTC modem has a number of key processing functions relating to the Telemetry and Telecommand Processing (Table 12.1). These can be primarily separated into:

Table 12.1 TTC modem functions overview

Function	Description
I/O Routing	The I/O Routing is responsible for selecting the source and destination for the other processing blocks. For inputs, the selection is one from many whereas outputs can normally be selected to go to none, one or multiple destinations simultaneously
TMA—TM Acquisition	The TMA receives serial data from a selection of one of the following: Primary Bypass, Redundant Bypass, Demodulator or TM Simulator
	The TMA receives, detects and processes the CADUs. Including Viterbi decoding/error correction (K=7, Rate½ for this project), derandomization, RS(223, 255 IL5) error detection and correction, acquisition time stamping. The acquired frames are then checked (MCFC, VCFC consistency etc.) and passed to the Source Packet Reconstruction process from which packets are recovered. The TMA also includes the extraction of CLCWs for the TCG Service A handling
	The TMA includes an intelligent frame synchronization process that is designed to reliably capture TM in the presence of noise but with low latency and extremely fast acquisition
	Acquired frames are time stamped within the hardware before data is passed through to the software layers. The software processing includes the identification and notification of spacecraft time correlation events based on the hardware timestamp of the appropriate received CADUs and subsequent time packet
	During the overall TM processing, data at CADU and Packet level is available for monitoring in the local GUI and can be archived
	During the TMA process, statistical information is collected regarding Viterbi error counters, Frame errors (MCFC, VCFC, FECW, SCID etc.), RS errors and packet errors
	The SSBV TMA has been proven over many years
TCG—TC Generation	The TC Generation process outputs serialized CLTUs to none, one or multiple destinations (Bypass Primary, Bypass Redundant, Modulator, TC Acquisition)
	The TC Generator can serialize pre-prepared CLTUs and includes a local GUI editor to create and save CLTUs. This can be very useful for low level debugging but not operationally
	In some projects, the CLTU interface is used to forward data from an NDIU or from an encryption process

(continued)

Table 12.1 (continued)

Function	Description
	The nominal path however is that the TC Generator receives TC Packets from the CCS and must convert these into segments, frames and CLTUs. The TC Generator can send TCs to selectable MAP-IDs on selectable VC-IDs. The TCG also supports both BD-service (send-and-forget) and AD-service (expedited service). The AD-service uses feedback from the Telemetry chain (CLCWs) to try to ensure transmission of TCs to the spacecraft
	The symbol rate of the TC generation is driven by the subcarrier modulator. The specified TC rate is based on a 16000 Hz subcarrier with a symbol rate division ratio of 4. The link to the subcarrier is mandatory in order to ensure the zero crossing modulation transitions. If the modulator is not available or not configured for TC generation, a local clock generator within the TCG can be used
	The TCG includes control over the means of the CLTU generation (related to the PLOP modes) where data can be sent in RF Auto mode, Idle Continuous, RF Always or Idle Timeout
TMS—TM Simulation	The TM Simulation function is a special test mechanism that allows simple closed loop testing. The system generates a continuous stream of CADUs with the appropriate error correction (RS + Viterbi) and randomization. The CMS includes TM packet handling such that they can be correctly inserted into the CADU stream. The system includes a TM Packet editor tool that can be used to define TM Packets at hexadecimal level. Prepared packets can be stored with descriptive text and grouped as one or more packet per file. A second tool allows the prepared packets to be grouped in TM Packet Lists that can be simply recalled for specific tests
	In addition to the Frame (CADU) based generation, the system also supports raw data replay with or without further coding
	The symbol rate of the TMS generation is driven either by a local internal clock generator or by the subcarrier modulator. The specfied TM rate is 128205 ksps. The link to the subcarrier is mandatory if the TM is to be modulated in order to ensure the zero crossing modulation transitions. If the modulator is not available or not configured for TM generation, the local clock generator within the TMS can be used
	The TMS output can be fed either into the modulator or looped directly back into the TMA such that it can be further processed as though it was Telemetry coming from the spacecraft
	The TMS does not support Service A feedback but does fill in a basic CLCW
TCA—TC Acquisition	The TC Acquisition function receives serial data, detects and captures the CLTUs (including BCH correction/detection). The TCA is typically used as an Echo decoder to "listen" to the output of the TCG. The recovered CLTUs can be displayed and stored and TC Segments recovered from these. When aggregation is not used, the TC Segments are the same as the TC packets

(continued)

Table 12.1 (continued)

Function	Description
Modulation	The modulator is responsible for the generation of the symbol rate clock for the TCG (or TMS) and the subsequent modulation of the TCG (or TMS) data onto the subcarrier that is then PM on the carrier. The Modulator also includes a Doppler Simulation function that adjusts the 70 MHz IF, subcarrier and symbol rates accordingly relative to the RF frequency. The Doppler Simulator supports linear and triangular profiles with adjustable range, rate and number of repetitions
	The modulator supports 2 identical (but individually selectable on/off) 70 MHz IF outputs
Demodulation	The demodulator supports a selection of 1 of 2 70 MHz IF inputs. The output of the carrier and subcarrier demodulation is made available for routing to the TMA or TCA as appropriate
Control and Monitor Software	The CMS provides the overall control and monitoring of the system. The CMS can be considered in two halves, the TM/TC Front End and the TTC Modem. The CMS provides the user with a suite of preparation tools for the creation and editing of TM and TC data (TM/TC Packets as well as TC CLTUs). Operationally it provides a remote and local mode. When in remote mode, the CCS controls the system and the local GUI can only be used to monitor data and status. In local mode, the viewing and monitoring is augmented with the capability to control the generation of data
Remote Interfaces	The SSBV CMS is designed to support different remote interfaces depending on the needs of the customer. For this project the C&C interface shall be supported
Other functions	The system includes a number of additional features: • Hardware based time stamping • Optional ITU compatible Bit Error Rate Tester (source and sink)

12.1.2.3 The TTC-IMBU Man-Machine Interface

The MMI is divided into several key sections:

• Menu	• TTC-CMS Status Viewer
• System Status	• Serious Error Display
• Master Log	• Status Bar
• System Control	• Data Editors, Generators and Monitors

Some example screenshots of the MMI are depicted in Figs. 12.12 and 12.13.

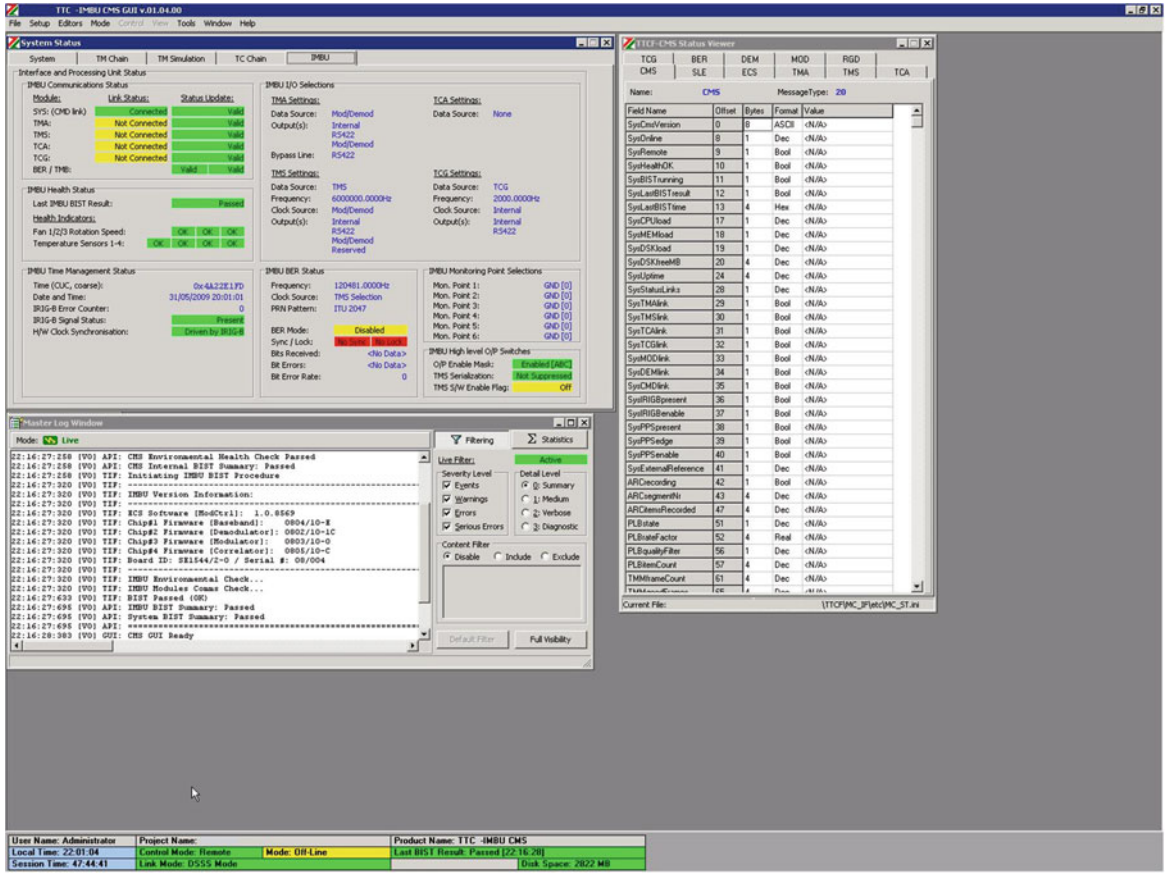


Fig. 12.12 Main TTC-CMS MMI overview. © Satellite Services BV and IRS, University of Stuttgart

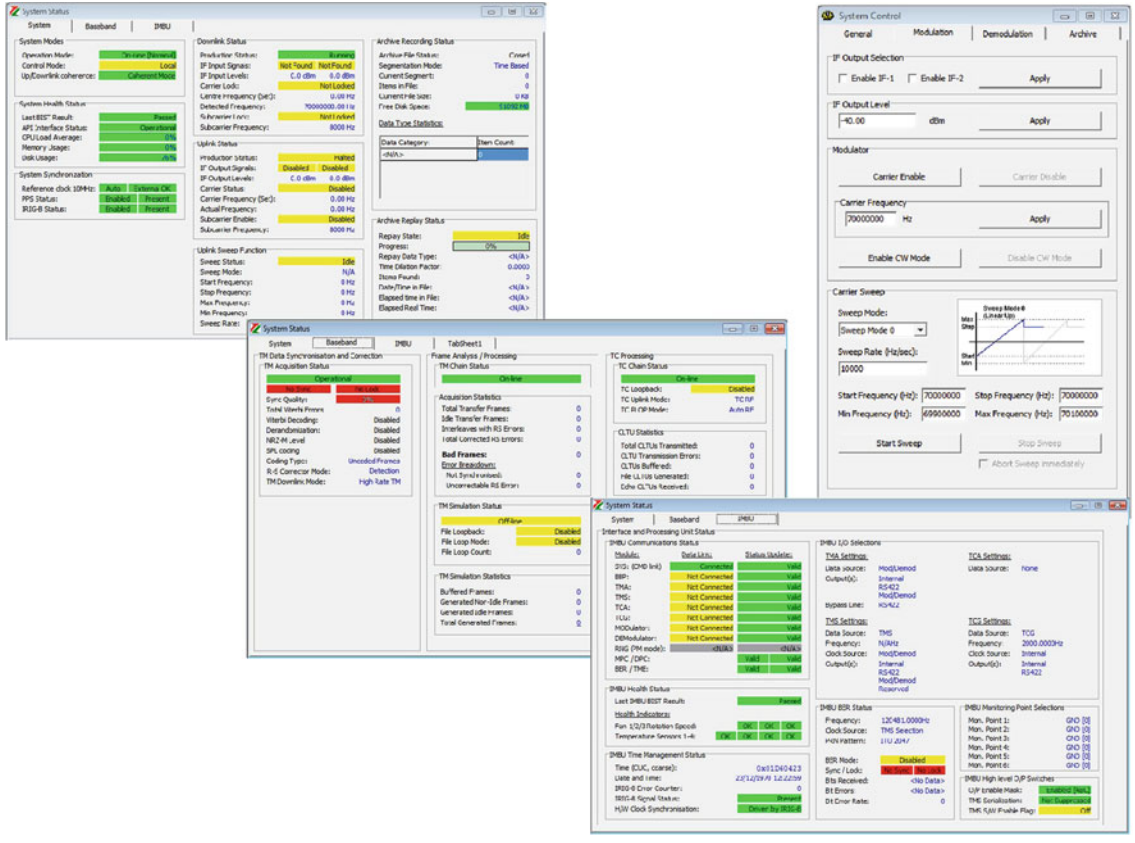


Fig. 12.13 System status, baseband unit status, IMBU status and sweep control. © Satellite Services BV and IRS, University of Stuttgart

Fig. 12.14 Frequency up and down-converter. © IRS, University of Stuttgart

12.1.2.4 RF Up/Down-Converters

The TTC-IMBU converts between the CCSDS digital signal level from/to the SCOS mission operations system and CLTUs/CADUs modulated onto the intermediate frequency (IF) of 70 MHz.

For the up-conversion to the ground/space transmit frequency of 2.0835 GHz respectively for the down-conversion of the downlink frequency of 2.2635 GHz to the intermediate frequency the infrastructure is enhanced by two commercial off the shelf converters from WORK-MICROWAVE, Munich, Germany (Fig. 12.14).

The dedicated types are an SCU-S-70-50 for up-conversion respectively an SCD-S-70-50 for the down-conversion. For more details please refer to the company's web-page [105].

12.1.3 The High Frequency Chain

On the track towards the antenna itself—please also refer back to Fig. 12.1—the next unit is the TTC HF Unit 2 which includes filters and the High Power Amplifier (HPA). The unit was developed in the frame of a PhD thesis at the IRS, Stuttgart (Fig. 12.15).

Fig. 12.15 TTC HF unit 2 with filters and HPA. © IRS, University of Stuttgart

12.1.4 The Antenna System

The antenna system comprises the antenna itself, the antenna drive motors and control electrics/electronics and the antenna control system. The parabolic antenna (Fig. 12.16) is suitable for broadband operation and was specifically designed for the FLP TC uplink frequency of 2.0835 GHz, the FLP TM downlink frequency of 2.2635 GHz and the frequency of 2.4 GHz for the science TM Data Downlink System (DDS).

Right hand circular polarization (RHCP) is used for satellite platform control TT&C signals. Left hand circular polarization (LHCP) is applied for the downlink of payload data via the DDS. The antenna reflector is mounted on a stand providing an elevation and azimuth drive. It transmits with an EIRP of 32.3 dBW.

The antenna feed already separates two chains of telemetry signals—the RHCP signals of FLP platform housekeeping TM in the commercial S-band—and the LHCP signals being science telemetry from the "Flying Laptop" specific Data Downlink System.

- Telecommands for uplink are coming from the previously presented TTC HF unit 2.
- Both the spacecraft TC and TM are filtered by the TTC HF unit 1 which for the TM downlink comprises an LNA and a filter and at the antenna end side it comprises the diplexer.
- For the LHCP science TM in amateur radio S-band in the case of the "Flying Laptop", the corresponding DDS HF unit 1 comprises an LNA and a filter.

Fig. 12.16 IRS antenna. © IRS, University of Stuttgart

Both TTC and DDS HF units 1 are integrated into the antenna stand and are depicted in Fig. 12.17.

The antenna drives with their power electrics, drive motors, position sensors and control electronics are shown in the Figs. 12.18 and 12.19 only for completeness to provide an impression of the overall system.

The antenna drive control is performed by the already mentioned software system METEOR. Its Man-Machine Interface is depicted in Fig. 12.20. METEOR receives the required antenna azimuth and elevation angle data over time from the flight dynamics infrastructure toolset which is described in more detail in Sect. 12.2.

Concerning satellite contact using the IRS antenna station constraints must be taken into account due to effects induced by some nearby buildings. Furthermore, ground contact to IRS will be lost when the satellite passes near zenith and when it passes in the direction of the sun. The zenith loss is due to the limits of the antenna drive angular velocity near zenith. The second limit is due to high noise levels caused by the sunlight.

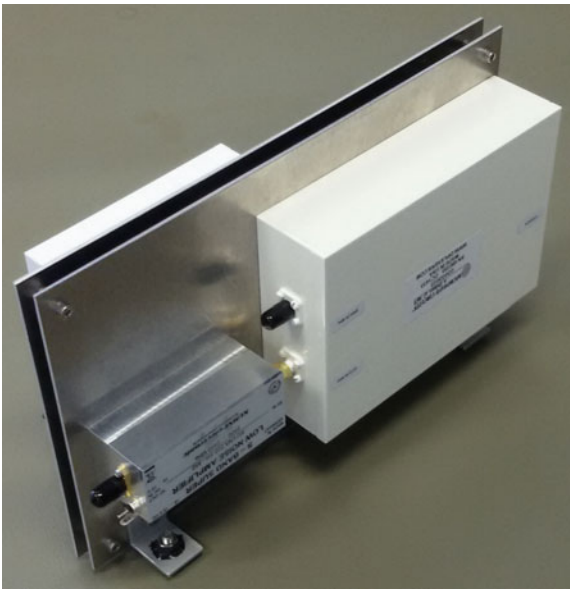


Fig. 12.17 TTC HF unit 1 (*backside*) and DDS HF unit 1 (*frontside*). © IRS, University of Stuttgart

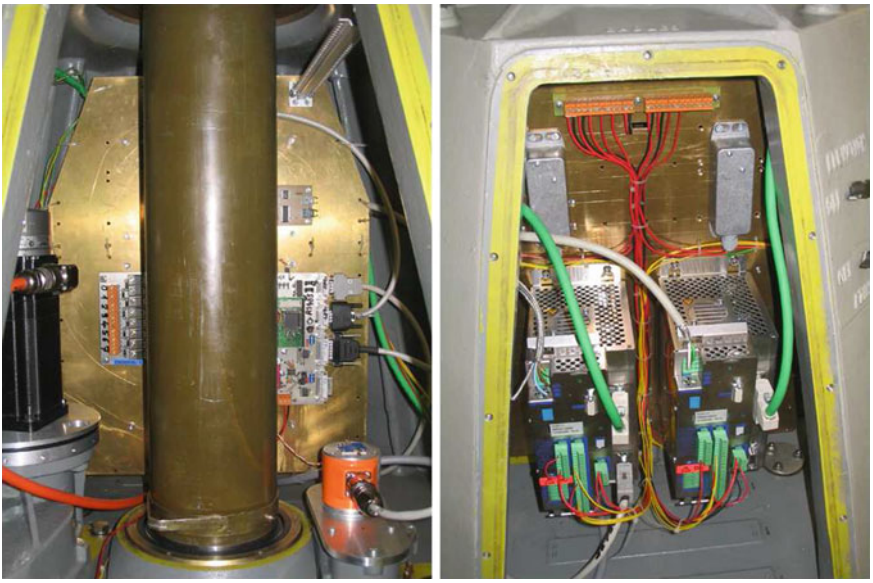


Fig. 12.18 Antenna azimuth drive, position sensor, electronics and elevation/azimuth power electrics. © IRS, University of Stuttgart

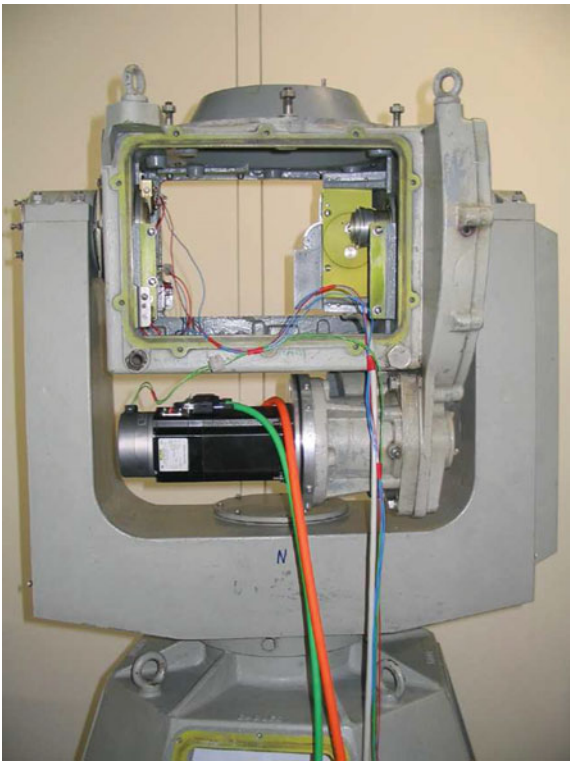


Fig. 12.19 Antenna elevation drive, position sensor and electronics. © IRS, University of Stuttgart

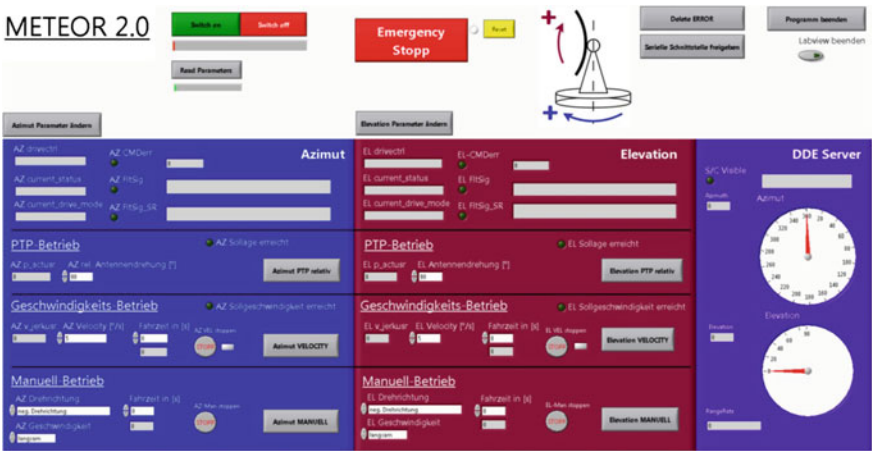


Fig. 12.20 Control software for the IRS antenna. © IRS, University of Stuttgart

12.2 Flight Dynamics Infrastructure

The flight dynamics infrastructure (Fig. 12.21) serves for

- to analyze the available orbit and position data of the satellite,
- to therefrom generate position propagations for the upcoming orbits and
- based on these orbit predictions to identify the data for Weilheim and Stuttgart ground station passovers.

Furthermore the orbit propagations serve as input for the mission planning infrastructure for planning ground contact activities as well as for planning payload operations activities such as target imaging etc. This mission planning aspect is further detailed in Chap. 14.

The core of the orbit analysis and propagation is based on the commercial tool Systems Tool Kit (STK) [109]. The overall infrastructure manages diverse data import formats and conversions, the control of STK functions for orbit propagation and output generation in the required file formats.

As stated above the core task of flight dynamics toolkit is to compute the actual orbit data of the satellite. In the very early flight phases, IRS is supported by the DLR/GSOC flight dynamics team. Based on data from the launch supplier DLR will perform the satellite contact establishment with the DLR antenna station and by functions of its antenna infrastructure it will track and deduce the satellite's position and trajectory data at initial passover. These data will be subsequently improved

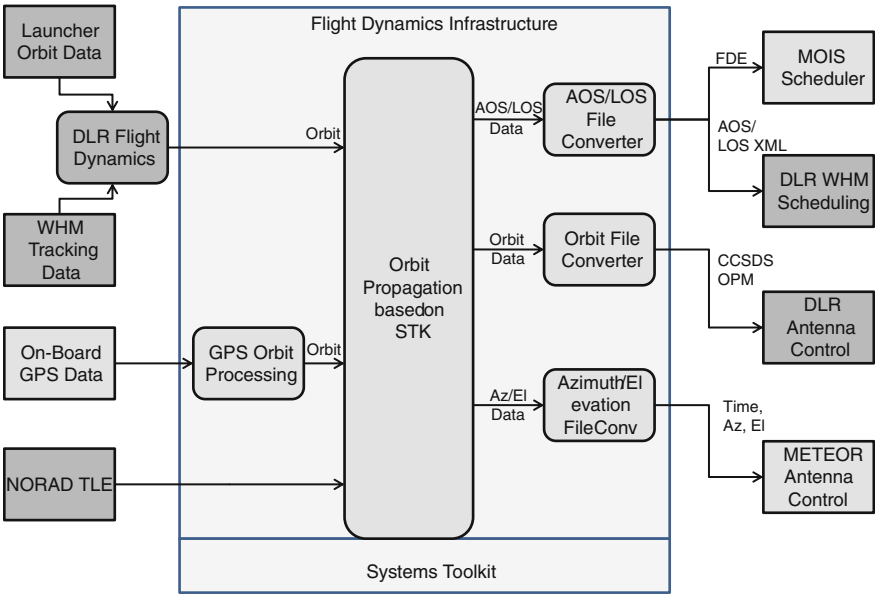


Fig. 12.21 Flight dynamics infrastructure. © IRS, University of Stuttgart

during the following passovers and orbit data will be handed over to the IRS as input to its own flight dynamics infrastructure to model the orbit in STK.

Another source of orbit information is becoming available in the timeframe of days after launch. These are object tracking data from the North American Aerospace Defense Command (NORAD). The NORAD orbit information is provided in the form of so-called Two Line Elements (TLE). For a very brief introduction to the TLE format see the annex Sect. 17.14 taken from [110] to [112]. The flight dynamics infrastructure provides an interface to import NORAD TLE data of the satellite and these can be used as input for orbit computation in STK.

The final source of information are GPS position data in the satellite HK telemetry after the onboard GPS of the satellite has been activated during Commissioning Phase. These will be available both live during passover, as well as playback position data over the recent orbits. These GPS data are the most precise information and are used for the orbit computations during nominal satellite operations.

The output of the flight dynamics computation can be split into two types of data:

- The first type are parameters for control of an antenna and
- the second type are scheduling data for planning antenna station requests or mission target visibilities for science payload operations.

The mission planning data comprise the Acquisition of Signal (AOS) time and Loss of Signal (LOS) times for each station passover.

In case of mission planning during the nominal spacecraft operations after the Commissioning Phase—including payload operations—the AOS/LOS data are handed over to the MOIS mission planning infrastructure at IRS. More details on the mission planning and the applied MOIS infrastructure are provided in Chap. 14.

In case of LEOP and Commissioning Phase as well as in emergency support cases the AOS and LOS parameters will be handed over to the antenna station team in Weilheim in one of the DLR prescribed data formats—as far as they are not already computed by DLR e.g. during LEOP phase. DLR uses these data for generating the overall allocation schedules of its diverse antennas. More information about the work process, antenna allocation scheduling etc. is provided in Sect. 13.4.

In case of usage of the DLR station in Weilheim the antenna control data will be converted to a CCSDS standard format—the so-called Orbit Parameter Message (OPM)—see [113] and is transmitted to DLR/GSOC.

In case of use of the IRS antenna station, the orbit information is directly converted to antenna azimuth and elevation angle parameters and is handed over to the METEOR antenna control software (Fig. 12.20).

12.3 Spacecraft Link Establishment Process

In order to establish a TM/TC radio link to the satellite from the IRS GS, the GS antenna is automatically directed to the expected satellite location at 5° elevation in case of North-South passover and at 10° elevation in case of South-North passovers. The latter is due to avoid campus wireless LANs interfering or disturbing satellite carrier acquisition.

An unmodulated carrier wave (Continuous Wave Mode or CW-Mode) is transmitted shortly before the satellite is expected to appear at this location and carrier frequency sweeping will be induced to account for Doppler shift due to the satellite motion towards the. During sweep the point will be reached where the receiver on board will lock its PLL to the identified carrier. The onboard transmitter then will be activated by the *TTC_Controller* in the OBSW and will submit its carrier and idle frame TM signal back to the GS.

Once this signal is received at level of the RF-SCOE in the ground station and the receiver is locked onto the signal, the CW-Mode is disabled in the RF-SCOE and idle frames are uplinked to the satellite. The full lock of the satellite receiver can be determined on ground by interpreting the uplinked Command Link Control Words (CLCW) being present in every downlinked frame.

In such case for final bidirectional link verification a, a ping TC (PUS Service 17) is sent to the satellite and ping TM should be received by the GS.

During the entire satellite passover, the antenna pointing is continuously adjusted according to the predicted satellite position. No satellite tracking by means of ranging of the signal is performed. The IRS GCC relies on orbit data provided by the launch supplier and DLR/GSOC for initial orbit injection and later by tracking data provided online by NORAD and on satellite GPS HK-TM.

12.4 Science Data Signal Chain

Onboard the spacecraft the science data are transferred from the payload control computer to the downlink system. In case of the first FLP Generation 1 based "Flying Laptop" the payload controller PLOC was already introduced in Sect. 1.9 and Chap. 9. It transfers the payload science TM to the so-called Data Downlink System (DDS).

The DDS system input from PLOC is a pushed synchronous data stream with a maximum raw data rate of 10 Mbps. The data structure is defined by the PLOC according to the type of payload. According to data type and rate the PLOC optionally can reduce the data rate to DDS by multiples of factor 2. Such a reduction improves the link budget.

Inside the DDS unit on board the raw onboard data stream from the payload computer is packed into CCSDS frames which are converted to CADUs being modulated onto the carrier.

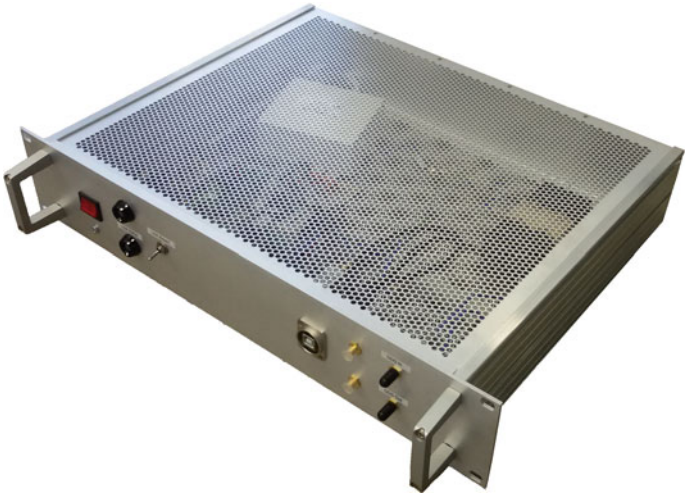


Fig. 12.22 DDS HF unit 2 with filter, amplifier and signal quality monitoring © IRS, University of Stuttgart

The science data downlink with the current DDS subsystem of the FLP Generation 1 uses the amateur radio frequency band.

- The center frequency is 2.408 GHz and
- the bandwidth is <8 MHz
- The waveform is 10 Mbps QPSK modulated and SRRC filtered ($\alpha = 0.4$)
- Transmitter Output power: max 1 W, typical 0.5 W
- The polarization is left handed.

As already mentioned in the IRS antenna the signal split between downlinked S/C TM data and the science data postprocessing chain is performed by means of a polarization splitter. The right hand polarized signal is fed into the satellite TM chain and the left hand circular polarized signal is routed into the science data downlink chain—see Fig. 12.1.

The science data HF signal then passes through a filter and is amplified by means of a low noise amplifier in the DDS HF unit 1 (see Fig. 12.17). Thereafter it passes through the DDS HF unit 2 (see Fig. 12.22) where it is filtered again, is amplified and the signal gain is measured/monitored.

Finally the signal it is down-converted in the DDS Frontend unit, is demodulated and is forwarded into the science packet processing in the "DDS Protocol Unit" (Fig. 12.23)—again please also refer to Fig. 12.1. In the DDS Protocol Unit follows the reconstruction of CADUs and the re-assembly of the CCSDS science data frames according to the following technical specifications:

Fig. 12.23 DDS protocol unit © IRS, University of Stuttgart

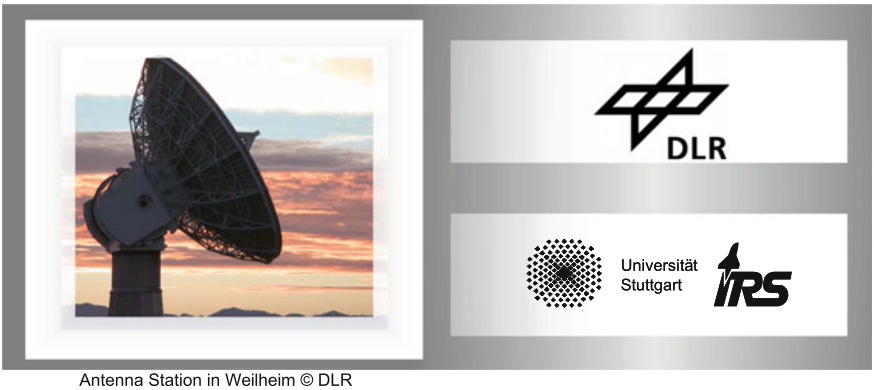
- Layer 1: CCSDS TM synchronization and channel coding:
  - No differential coding: Phase is disambiguation over ASM
  - CADUs length: 1279 octets including ASM for non-turbo-coded data
  - ASM = 0x1ACFFC1D
  - Reed-Solomon encoding: yes (255, 223 CCSDS standard with Berlekamp dual basis conversion)
  - Interleave factor: 5
  - Pseudo randomization: yes (CCSDS standard)
- Layer 2: CCSDS TM data link:
  - Frame length: 1115 octets including header
  - Secondary header: no
  - Trailer: no
  - Number of virtual channels: 1

As output of the DDS Protocol Unit the reconstructed payload data are directly transferred via Ethernet TCP to the Payload Data Storage server in the ground segment. The DDS unit comprising down-converter and Protocol Unit is a development of IRS.

Since the type of payload data are largely depending on the mission payloads, they are not treated further in this platform FOM.

Chapter 13
Stuttgart/DLR Infrastructure for LEOP

Peter Willburger, Klaus Wiedemann, Rolf Kozlowski, Marcin Gnat, Ciprian Furtuna, Jens Eickhoff and Nico Bucher



Antenna Station in Weilheim © DLR

P. Willburger (✉) · K. Wiedemann · R. Kozlowski · M. Gnat · C. Furtuna
Deutsches Raumfahrt Kontrollzentrum (GSOC), Oberpfaffenhofen, Germany
e-mail: peter.willburger@dlr.de

K. Wiedemann
e-mail: klaus.wiedemann@dlr.de

R. Kozlowski
e-mail: rolf.kozlowski@dlr.de

M. Gnat
e-mail: marcin.gnat@dlr.de

C. Furtuna
e-mail: ciprian.furtuna@dlr.de

N. Bucher
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: bucher@irs.uni-stuttgart.de

J. Eickhoff
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_13

Abstract The chapter explains the link between the Stuttgart Mission Control Center and the DLR/GSOC antenna station in Weilheim, Germany, which the IRS is permitted to use for the Launch Phase and the Commissioning Phase of the first FLP-based spacecraft. The chapter is complementary to Chap. 12 and describes the technical details of the TC/TM routing from Stuttgart to Weilheim and the TM routing backwards as well as the RF infrastructure and antenna in Weilheim. It also covers the operational workflow between Stuttgart and Weilheim during these early flight phases of the satellite.

Keywords Mission control infrastructure · IRS-GSOC network bridge · Space link extension system · RF signal chain · DLR antenna station weilheim · Passover scheduling · LEOP phase · Commissioning Phase

13.1 Link Between IRS and the DLR Antenna Station

For commanding the "Flying Laptop" satellite from Stuttgart during LEOP and Commissioning Phase the IRS is supported by the German Space Agency "Deutsches Zentrum für Luft- und Raumfahrt" (DLR). The IRS in Stuttgart is granted to use one of the 15 m S-band antennas of the DLR array in Weilheim, Germany for operating the satellite, with the geodetic coordinates as listed in Table 13.1.

Therefore an interface between the IRS mission control facilities in Stuttgart and the DLR "German Space Operations Center" (GSOC) multi-mission node in Oberpfaffenhofen, Germany, has been established to transmit real-time and offline data. Figure 13.1 shows the real-time data (TM/TC) interface using a Space Link Extension (SLE) connection via a VPN tunnel, allowing IRS a secure connection to

Table 13.1 Weilheim 15 m antenna coordinates

Station	Latitude	Longitude	Altitude
S67	47° 52‘ 48,21452“ N	11° 05“ 07,15603“ E	663,942 m
S69	47° 52‘ 52,29358“ N	11° 05“ 01,09200“ E	663,937 m

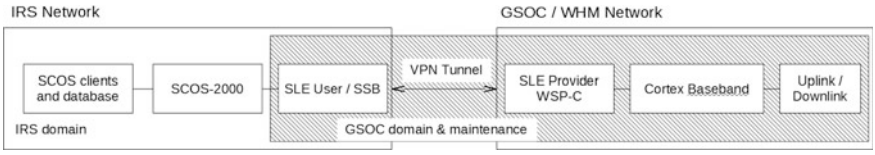


Fig. 13.1 IRS—GSOC network bridge. © DLR/GSOC, © IRS, University of Stuttgart

GSOC in Oberpfaffenhofen, and from there to the Weilheim antenna station (details see Sect. 13.3.1).

As for the commanding of the satellite via the IRS own antenna station in Stuttgart, the satellite command/control will be performed via the SCOS system. Some technical details however are different w.r.t. the command setup in Stuttgart:

As explained in Chap. 12 the SCOS input/output towards the RF-SCOE are TC/TM packets in the Stuttgart architecture. Transmission details on lower protocol layers—such as Virtual Channel (VC) selection—are performed in Stuttgart via the RF-SCOE equipment.

To allow full command and telemetry control from the SCOS system (without the RF-SCOE), in the setup with DLR the SCOS TC output is arranged on Command Link Transmission Unit (CLTU) level. This allows to access also Frame and Segment settings by SCOS-2000—such as VC parameters (please also refer to Figs. 2.6 and 2.7). The TM route from the antenna station back to SCOS is implemented on Frame level.

13.2 DLR Antenna Station Infrastructure

This section shall provide a brief overview on the key features of the DLR antenna station infrastructure as far as relevant for this mission. Not all elements of Fig. 13.2 will be discussed in detail—in particular not the antenna drive and electronics.

In the control room (see Fig. 13.3) all activities for the ground contact of the assigned satellites are performed. This includes:

- Preparation of the antennas according to an overall acquisition schedule presented further below,
- health monitoring of the antennas,
- monitoring of proper RF signal acquisition with the assigned satellites at time of acquisition,
- monitoring of loss of signal (LOS) not earlier as predicted,
- analysis of signal quality.

The satellite command/control is performed by the operators. For DLR owned or DLR operated satellites they are located for example in the GSOC control rooms in Oberpfaffenhofen, Germany, or—in case of the “Flying Laptop” satellite—they are located in the IRS control room in Stuttgart. TC/TM signals which are exchanged between Stuttgart and Weilheim via the IRS—GSOC network bridge (depicted in Fig. 13.1) are RF-encoded/decoded in Weilheim via Cortex Command, Ranging and Telemetry Processors from Zodiac Data Systems [117] (see Fig. 13.4). The Cortex processor performs the following functions (only those relevant for the FLP missions being listed):

- CCSDS TC processing up to COP-1 transfer layer
- Digital signal modulation to 70 MHz intermediate frequency

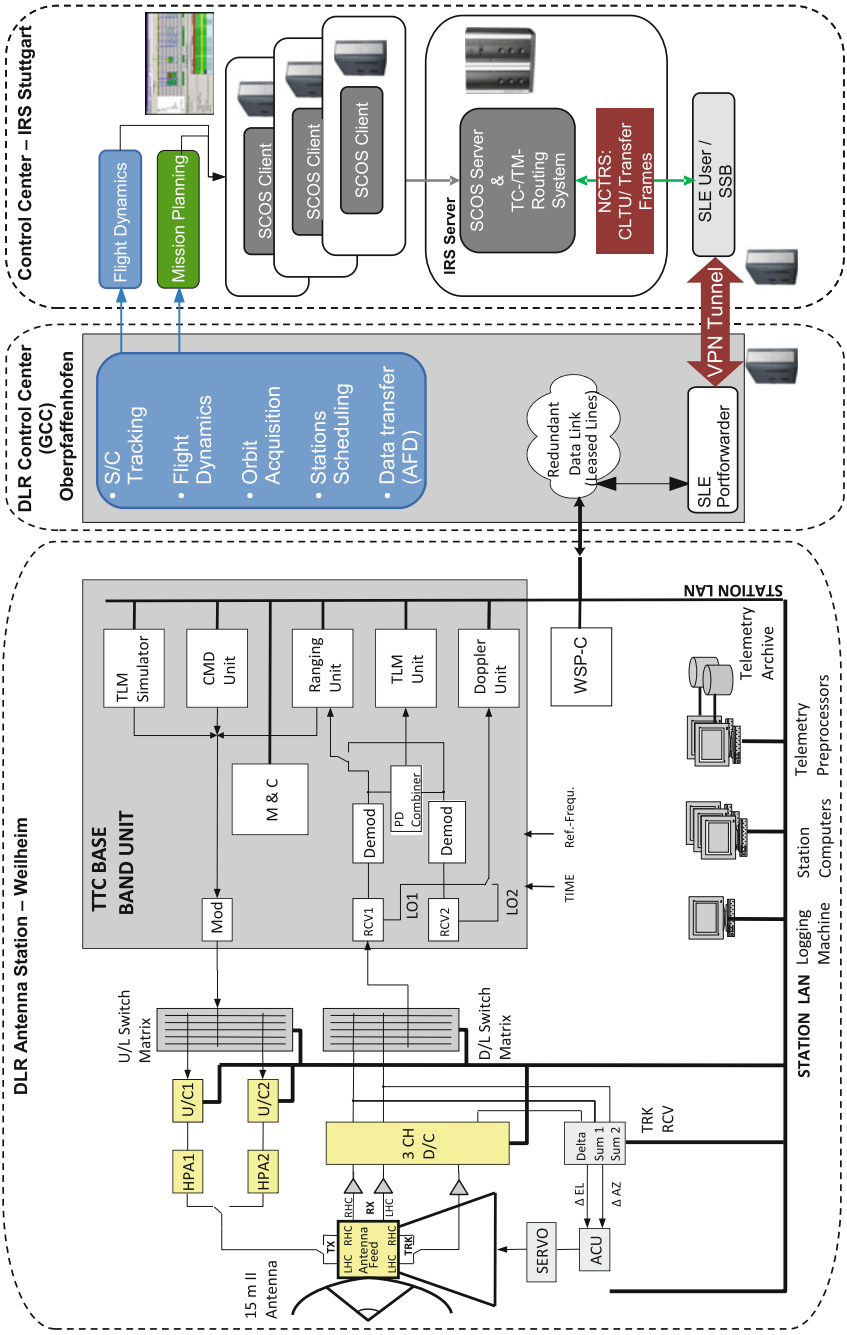


Fig. 13.2 Link to DLR /GSOC antenna station from IRS Control Center. © DLR/GSOC, © IRS, University of Stuttgart

Fig. 13.3 DLR antenna control room in Weilhei. © DLR/GSOC, © IRS, University of Stuttgart

Fig. 13.4 Cortex—CCSDS based digital/RF–RF/digital conversion processor. © Zodiac Data Systems (from [117])

Fig. 13.5 Cortex room in Weilheim. © DLR/GSOC and IRS, University of Stuttgart

Fig. 13.6 One of the 15 m antennas of the DLR array in Weilheim. © DLR/GSOC and IRS, University of Stuttgart

- Telemetry processing for direct PCM and PCM with sub-carrier scenarios
- Demodulation and decoding
- Ranging—which is not applied for FLP
- Doppler estimation
- Spread spectrum processing for TC, TM (and ranging)
- Signal quality monitoring

The Cortex processors are cross-allocatable to the diverse antennas in Weilheim and are placed in a dedicated server room in Weilheim (Fig. 13.5).

The TC RF upconversion from intermediate frequency and the high power amplification both are performed by equipment directly mounted within the antenna foundation (Fig. 13.6) to be as close as possible to the antenna dish itself. The same applies for the RF receiver and the TM downconversion to intermediate frequency.

13.3 GSOC Systems Supporting the FLP Mission

This section gives a brief overview on the main systems provided by GSOC to support the FLP-based "Flying Laptop" mission. Besides the Network Integration System (NIS) to interface network wise IRS with GSOC a short description of the Automated File Distribution (AFD) and Space Link Extension (SLE) system can be found in the following subsections.

13.3.1 Space Link Extension (SLE) System

The data link to support the IRS mission is realized through the CCSDS Space Link Extension (SLE) protocol. An SLE connection is used to transfer Telemetry and Telecommands (TM/TC) between Stuttgart and DLR/Weilheim as shown in Fig. 13.1. The SCOS server in Stuttgart is connected to the SLE user gateway—the SLE Switchboard (SSB)—which is a GSOC maintained LINUX PC with the SSB software installed. Besides the SLE user gateway functionality, the SSB computer at the IRS features several applications by which the entire TM/TC chain can be tested locally on the computer without any network connection.

On the other side of the link the SLE provider gateway "Weilheim SLE Provider to Cortex" (WSP-C) is installed.

The SLE protocol is a standard defined by CCSDS. Relevant recommended standard documents (so called blue books) are

- the SLE Cross Support Reference Model [98],
- the TM Return All Frames (RAF) service specification [99],
- the TC forward CLTU service specification [100],
- the Internet Protocol for Transfer Services [101].

13.3.1.1 SLE Switchboard (SSB) System

The SLE Switchboard (SSB) is a front-end computer installed in the mission control center to support the interface with various types of ground station equipment. The SSB supports acquisition of telemetry and sending of telecommands.

The SSB offers interfaces to support various MCS. For the FLP mission the following interfaces are used:

- SLE protocol for TM/TC to SLE Provider (WSP-C)
- Network Control and Telemetry Routing System protocol (NCTRS) for TM/TC to SCOS

The primary functions of the SLE Switchboard (SSB) software are (cf. also Fig. 13.7):

- Reception of telemetry streams from multiple service instances (i.e. data can originate from different SLE providers)
- Transmission of telecommand streams to FCLTU service instances (one command stream per service instance)
- Multiplexing and delivery of telemetry streams to multiple SCOS instances
- Reception of different telecommand streams from different SCOS instances

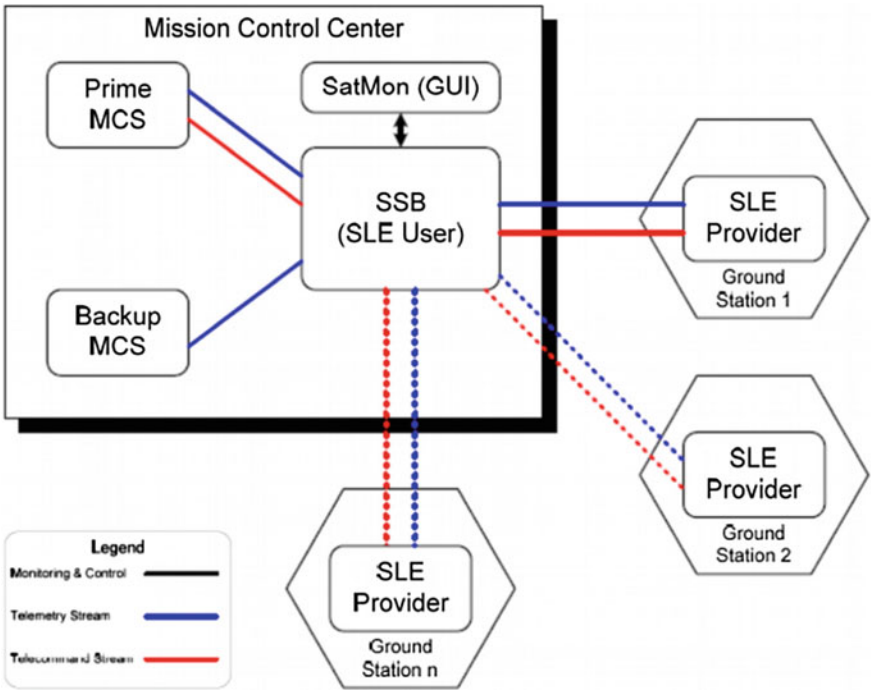


Fig. 13.7 SSB system context. © DLR

- NCTRS protocol adaptation
- Support for transparent ground station hand over (i.e. capability to switch telemetry and telecommand streams)
- Logging capabilities

For a better understanding of the primary intention and operation of the SLE Switchboard software the diagram of Fig. 13.7 shows the general processing context:

The SSB provides the capability to operate multiple independent service instances, multiple independent SCOS ports and the possibility to change the routing of the data streams flowing between the SCOS systems and the ground stations.

The SSB can support the connection of up to two ground stations for TC and up to eight for TM. Each SSB instance allows up to four SCOS systems to receive TM data but only one SCOS to connect to the TC port. SSB performs ground station handover through its routing functionality (see Fig. 13.8).

Fig. 13.8 SSB data flow. © DLR

13.3.1.2 SLE Provider (WSP-C)

The default TM/TC interface of Weilheim ground station is the standardized CCSDS SLE (Space Link Extension) protocol. The ground station takes the role of SLE Provider (please also refer to Fig. 13.7), being able to provide the TM from the baseband equipment to the control center and forward the TC from the control center to the equipment.

The WSP-C (Weilheim SLE Provider) is the SLE Provider used by DLR. It implements a Cortex adapter being able to receive the TM from and to send the TC to the Cortex.

TM from ground stations can be received at the control center (IRS) and can be forwarded to the spacecraft Mission Control System (SCOS-2000). TCs from the control center's spacecraft mission control system can only be forwarded to one of the ground stations at a time. The SLE service provider data flow is shown in Fig. 13.9.

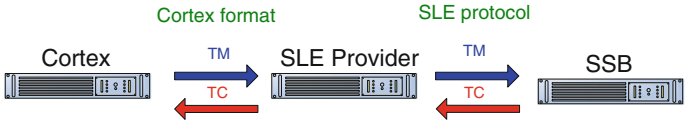


Fig. 13.9 SLE provider data flow (WSP-C). © DLR

The SLE protocol implemented in the WSP-C offers the following functionalities:

- TM (3 modes are available):
  - Online complete: all TM is delivered. If the network gets congested, TMs are queued and are sent later.
  - Online timely: TM is delivered in real-time. If the network gets congested, some TM frames are discarded at the provider side to maintain the real-time quality of the link.
- Filtering:
  - TM can be filtered by Virtual Channel (VC)
- TC:
  - All TCs are received in sequence by the SLE provider. TCs can be configured to be sent only at a dedicated time-tag to the TT&C equipment.

13.3.2 Automated File Distribution System

The Automated File Distribution System (AFD) is a file distribution system available for satellite projects (see Fig. 13.10). The AFD is developed as open source by the German Weather Service (Deutscher Wetterdienst).

In case of the FLP project with IRS the AFD system will be used for the transfer of orbit information (TLE) from the IRS to the ground station Weilheim and in opposite direction providing the tracking information (angle data from the antenna) to IRS for further orbit estimations.

The AFD system is located in a multi-mission environment, and so each project defines its own file transfer matrix which acts as an input for the AFD configuration. The matrix defines what kind of files shall be transferred where and how often. As soon as the configuration is activated, the AFD system starts the monitoring of desired directories and performs transfers fully autonomously. The monitoring of its activity is available at GSOC (Fig. 13.11). The respective check and alerting in case of issues belongs to tasks of the systems operator in the Systems Room. A detailed description of the AFD can be found at the website http://www.dwd.de/AFD.

The AFD tool has the following features:

- The AFD can send/retrieve any type of file, regardless of contents and the name of the file.
- Only the files are being distributed, no other additional information is sent with the file.
- The AFD has a very sophisticated and compact X11 interface to monitor and control the distribution of files. It has the ability to do extensive logging of all activities, even down to tracing each individual FTP/SMTP command.

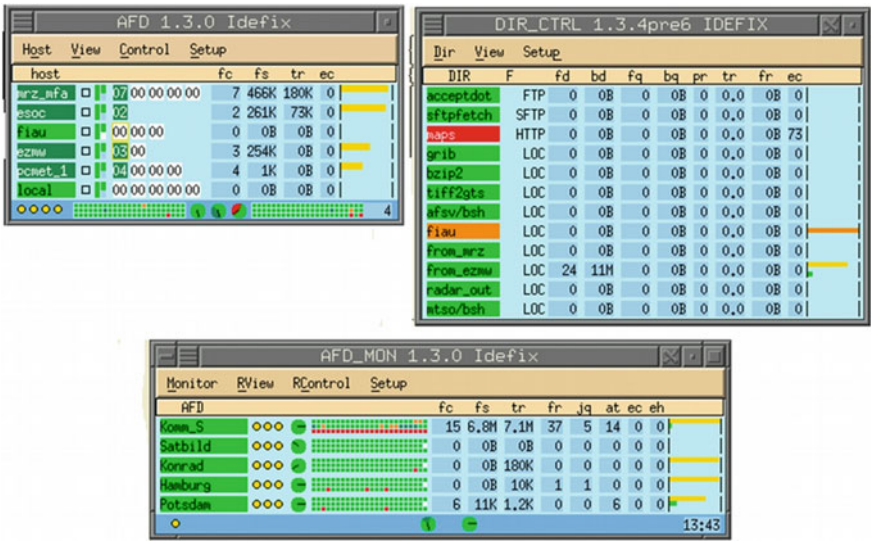


Fig. 13.10 Monitoring interface of the Automated File Distribution System. © DLR

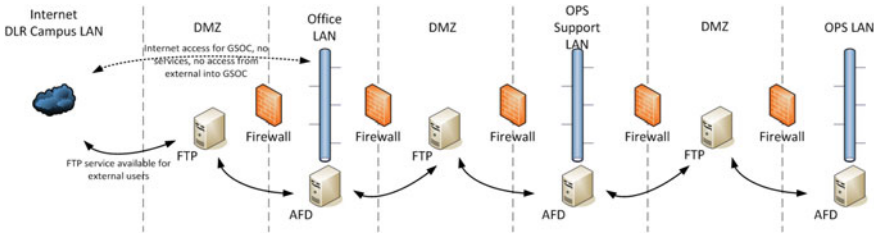


Fig. 13.11 AFD configuration in GSOC network. © DLR

- Files can be submitted to receivers where no AFD is installed.
- There can be more than one distributing process submitting files to a single host. This has the advantage that when a large file is currently under transfer, other files are not kept pending in transmission.
- Where net capacity is limited, it is possible to distribute files by priority.
- If the transfer gets interrupted, AFD can append files. So that data that has already been transmitted will not be resent again.

13.4 Operations Workflow

This section describes the operations workflow for the mission support provided by GSOC to IRS—comprising:

- Scheduling Process
- LEOP Phase
- Commissioning Phase and
- Routine Phase

as explained in the following sub-sections.

13.4.1 Scheduling Process

The integral part of the DLR/GSOC operations is the GDS scheduling office, which takes care of planning and coordination of spacecraft contacts via DLR or external stations. The tasks of scheduling are:

- Receive ground station support requests from projects and coordinate allocation to the organization's own and to external ground stations.
- Perform contact planning according to mission requirements applying mission priority rules.
- Publish the weekly contact plan (schedule) for all MCC missions and resources.
- Provide support and propose solutions in case of conflicts.
- The scheduling process involves usage of a specific interface, which is described in the GSOC GDS Scheduling ICD [116].

The schedule interface allows XML based information exchange, which is in many cases the preferred choice. However in the case of the very short mission support period for the "Flying Laptop" and for the sake of simplicity, the scheduling interface between IRS and GSOC is realized based on text form schedule information provided via e-mail. An example of such a scheduling request is provided in Table 13.2.

The overall schedule generation process is depicted in Fig. 13.12.

Table 13.2 DLR/GSOC spacecraft specific schedule

DATE	SAT	STN	REVN	MAX ELE	AOS	LOS	OPRN
15 03 22	FLP	S67	00000	72.0	23:57:14	00:06:04	D/L ONLY
15 03 23	FLP	S67	00000	13.6	01:30:03	01:38:22	D/L ONLY
15 03 23	FLP	S67	00000	11.9	12:09:42	12:17:50	
15 03 23	FLP	S69	00000	31.9	23:25:41	23:35:36	

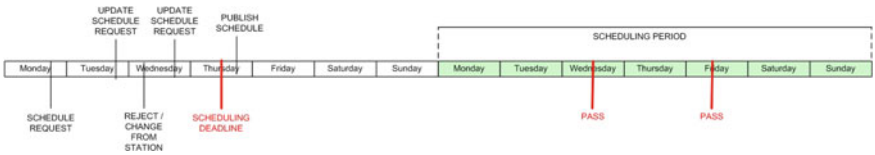


Fig. 13.12 Scheduling timeline. © DLR

Table 13.3 DLR/GSOC overall schedule

Monday PERIOD: 12.04.30 - 12.05.07 DTG OF REV: 12.04.26 / 1133utc REV: 0
REMARKS:

CW	DOY	DOW	DATE	START	STOP	SUP ID	STATIONS	ACTIVITY		ITM
18	121	MON	12.04.30	0000z	2400z	EUT/S	WHM/S	STBY FOR EMGY SPT	.	1
18	121	MON	12.04.30	0000z	2400z	EUT/KU	WHM-KU	STBY FOR EMGY SPT	.	2
18	121	MON	12.04.30	0000z	0016z	TX1	OHIG/OHG	SPT	.	3
18	121	MON	12.04.30	0001z	0011z	TX1	OHG	PASS	--	4
18	121	MON	12.04.30	0018z	0053z	TX1	INUV/INU	SPT	.	5
18	121	MON	12.04.30	0025z	0057z	GR1	WHM/S67	SPT	.	6
18	121	MON	12.04.30	0026z	0058z	GR2	NST/NSG	SPT	.	7
18	121	MON	12.04.30	0038z	0048z	TX1	INU	PASS	--	8
18	121	MON	12.04.30	0045z	0052z	GR1	S67	PASS	--	9
18	121	MON	12.04.30	0046z	0053z	GR2	NSG	PASS	--	10
18	121	MON	12.04.30	0201z	0234z	TD1	KSAT/SG6	SPT SX	.	11
18	121	MON	12.04.30	0221z	0229z	TD1	SG6	PASS SX	--	12
18	121	MON	12.04.30	0343z	0416z	TX1	NST/NSG	SPT	.	13
18	121	MON	12.04.30	0403z	0411z	TX1	NSG	PASS	--	14
18	121	MON	12.04.30	0516z	0552z	TX1	NST/NSG	SPT	.	15
18	121	MON	12.04.30	0518z	0554z	TD1	WHM/S69	SPT	.	16
18	121	MON	12.04.30	0518z	0554z	TX1	WHM/S67	SPT	.	17
18	121	MON	12.04.30	0536z	0547z	TX1	NSG	PASS	--	18
18	121	MON	12.04.30	0538z	0549z	TD1	S69	PASS	--	19
18	121	MON	12.04.30	0538z	0549z	TX1	S67	PASS	--	20
18	121	MON	12.04.30	0651z	0725z	TX1	NST/NSG	SPT	.	21

Key elements of the schedule generation process are:

- Analysis of project needs and mission event files based on current orbit information
- Selection of specific passes over a specific ground station
- Provision of the contact request to GSOC, according to the timeline (1 week before).
- In case of conflicts, support the Scheduling Office in resolution.
- Thursday before the scheduled week, the schedule information is published.
- In case of planning a LEOP phase typically the Sequence of Events (SoE) is generated by the customer (IRS), which includes LEOP passes and serves also as an input to the Scheduling Office. In such a case no extra schedule request is required.

Besides the spacecraft specific schedules DLR/GSOC generates overall schedules for all managed spacecraft and all GSOC controlled antennas—the station in Weilheim can be found in the example of Table 13.3 according to its ID “WHM”.

13.4.2 LEOP Phase

During LEOP phase DLR will build up RF ground contact with the spacecraft and the TC/TM operations will be performed by IRS from Stuttgart. LEOP operations will follow the Sequence of Events (SoE) being provided by IRS.

Before Launch, IRS will establish a telephone connection to Weilheim for operator synchronization. As soon as the RF contact between antenna station and spacecraft is established, the IRS operators take over commanding the spacecraft which will comprise (see also Sect. 11.6):

- Reading live telemetry
- Downloading stored HK-TM
- Monitoring spacecraft health and proper mode transitions based on received telemetry
- Activation GPS as soon as possible for improvement of orbit data information
- Failure management if required

After the LEOP phase is completed successfully the operations are directly transferring to the Commissioning Phase.

13.4.3 Commissioning Phase

During the Commissioning Phase the IRS will provide satellite tracking requests to DLR according to the process described in Sect. 13.4.1 and the GDS Scheduling ICD [116].

At the beginning, the orbit information and the resulting orbital events, which comprise the basis for schedule request, will be provided by GSOC Flight Dynamics. The Flight Dynamics team calculates the spacecraft orbit based initially on the injection vector provided by the Launch Provider, and later based on tracking (angle) data from the Weilheim antenna as well as from NORAD TLEs. As soon as on-board GPS data is available, the IRS will provide its own orbit estimation and from there onwards schedule requests will be based on that information.

Before each ground contact, IRS will establish a telephone connection to Weilheim for pre-pass briefing. As soon as the RF contact between antenna station and spacecraft is established, the IRS operators take over commanding the spacecraft which will comprise:

- Reading live telemetry
- Downloading stored HK-TM
- Consecutive activation of on-board sub-systems during Commissioning Phase (see also Sects. 11.6 and 11.7)
- Commanding of spacecraft to higher operational modes
- Commanding of timelines for the platform
- Failure management if required

Towards the end of the Commissioning Phase also payloads are taken into operations. At this stage spacecraft platform commanding is still performed via the DLR ground station while for payload science data downlink already the IRS ground station will be used.

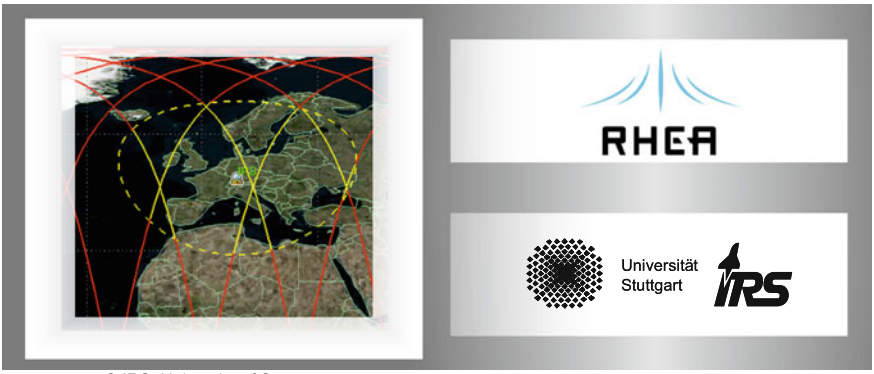
In the final stage also spacecraft platform operations (commanding and housekeeping telemetry acquisition) will be taken over by the Stuttgart ground station at IRS premises.

13.4.4 Routine Phase

The routine phase is performed with the IRS antenna in Stuttgart. GSOC with the Weilheim antenna is supporting emergency cases and performs maintenance for the involved GSOC equipment. For that monthly proficiency passes will be scheduled and performed.

Chapter 14
Earth Observation Mission Planning

Kai-Sören Klemich, Gianluca Cerrone and Wolfgang Heinen



Abstract The chapter explains the mission planning for an FLP-based satellite using SCOS and the Manufacturing and Operations Information System (MOIS). Mission planning considers the orbital events like begin/end of eclipse, begin/end of ground station visibility and begin/end of observation target visibility.

Keywords The Manufacturing and Operations Information System (MOIS) · Operations procedure modeling · MOIS scheduler · Orbit events · Mission planning workflow

K.-S. Klemich (✉)
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: klemich@irs.uni-stuttgart.de

G. Cerrone · W. Heinen
RHEA System S.A., Wavre, Belgium
e-mail: g.cerrone@rheagroup.com

W. Heinen
e-mail: w.heinen@rheagroup.com

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_14

14.1 MOIS Infrastructure

As already mentioned in Sect. 12.1.1 the IRS uses the "The Manufacturing and Operations Information System" (MOIS) for supporting operations tasks that are not directly covered by SCOS. These are

- the generation of Flight Procedures to command complex sequences from ground during spacecraft visibility
- and to generate flight schedules for combined platform and payload control to make the satellite perform its scientific tasks in orbit.

MOIS is an integrated set of tools for Spacecraft and ground segment testing, Operations preparation and Satellite Assembly, Integration and Test—developed by RHEA System S.A. in Belgium. The tools are used together with SCOS-2000 to create, execute, plan and export test and Flight Procedures (see Fig. 14.1).

The MOIS Flowcharter supports creating, editing and display of procedures through a flow chart display. It is used in parallel with MOIS Writer to define procedures. Figures 14.2 and 14.3 depict a procedure example in Flowcharter and Writer.

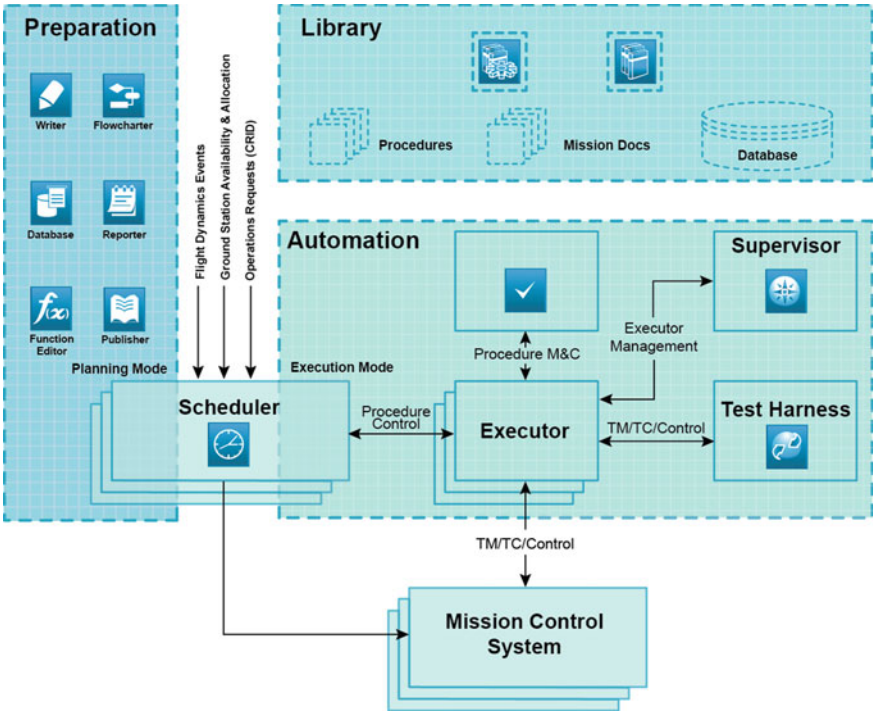


Fig. 14.1 The MOIS Infrastructure. © RHEA System S.A.

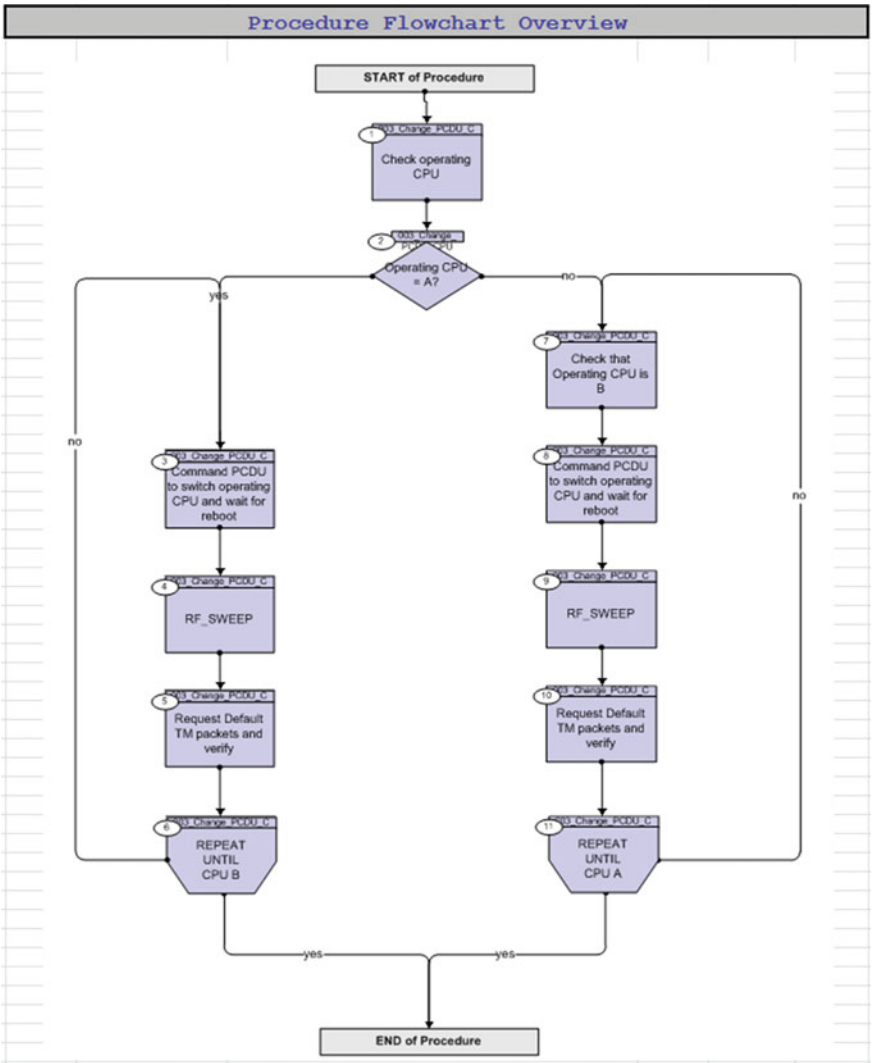
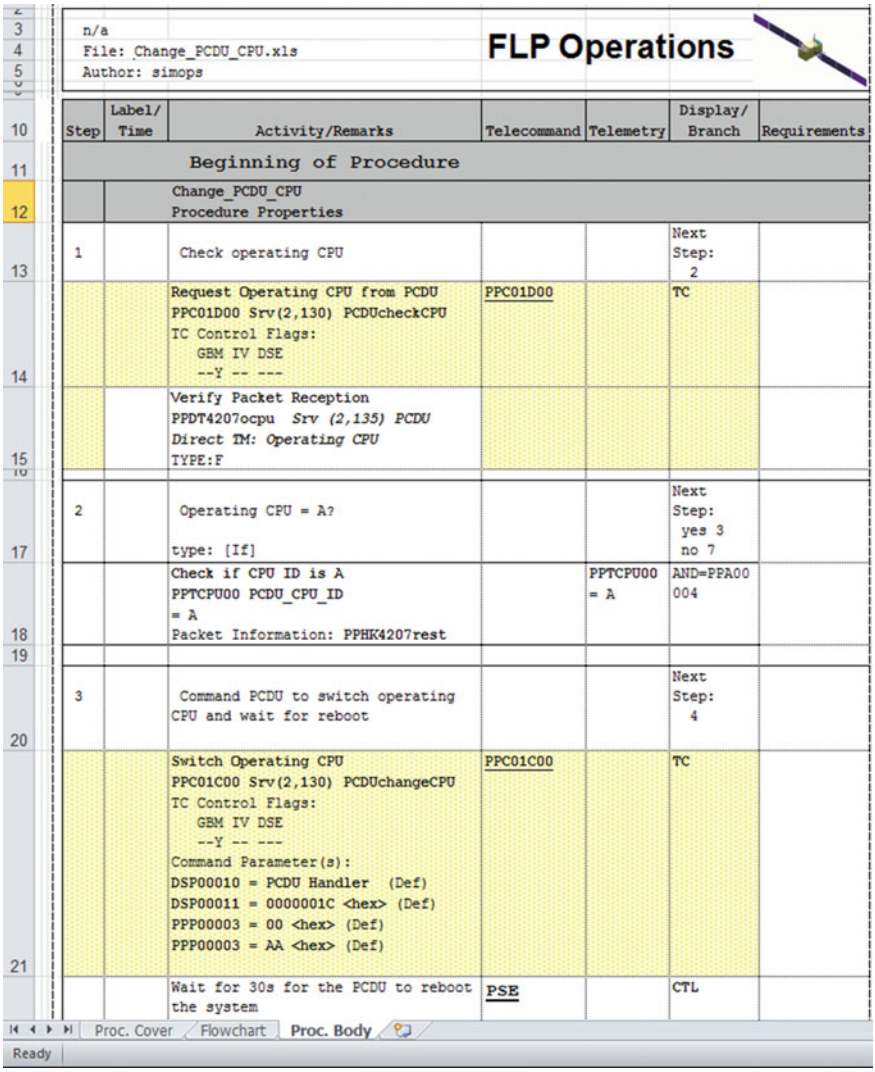


Fig. 14.2 Example procedure in Flowcharter. © IRS, University of Stuttgart and RHEA System S.A.

Writer is the MOIS infrastructure tool used for creating, modifying and displaying the operations procedures and timelines required for a space mission. Consistency with the chosen TM/TC database according to the required Mission Control System (SCOS in this case) is guaranteed.

MOIS Validator provides execution control for a predefined command procedure, independent to whether the procedure is connected to a test harness, a simulator or to the real spacecraft via the SCOS-2000 control system. Although called the MOIS



n/a
File: Change_PCDU_CPU.xls
Author: simops

FLP Operations

Step	Label/ Time	Activity/Remarks	Telecommand	Telemetry	Display/ Branch	Requirements
		Beginning of Procedure				
		Change_PCDU_CPU Procedure Properties				
1		Check operating CPU			Next Step: 2	
		Request Operating CPU from PCDU PPC01D00 Srv(2,130) PCDUcheckCPU TC Control Flags: GBM IV DSE --Y -- ---	PPC01D00		TC	
		Verify Packet Reception PPDT4207ocpu *Srv (2,135) PCDU* *Direct TM: Operating CPU* TYPE:F				
2		Operating CPU = A? type: [If]			Next Step: yes 3 no 7	
		Check if CPU ID is A PPTCPU00 PCDU_CPU_ID = A Packet Information: PPHK4207rest		PPTCPU00 = A	AND=PPA00 004	
3		Command PCDU to switch operating CPU and wait for reboot			Next Step: 4	
		Switch Operating CPU PPC01C00 Srv(2,130) PCDUchangeCPU TC Control Flags: GBM IV DSE --Y -- --- Command Parameter(s): DSP00010 = PCDU Handler (Def) DSP00011 = 0000001C <hex> (Def) PPP00003 = 00 <hex> (Def) PPP00003 = AA <hex> (Def)	PPC01C00		TC	
		Wait for 30s for the PCDU to reboot the system	PSE		CTL	

Proc. Cover | Flowchart | Proc. Body

Ready

Fig. 14.3 Example procedure in Writer with further details. © IRS, University of Stuttgart and RHEA System S.A.

Procedure Validator this application is also the user interface for manual and semi-automatic execution of procedures. The operator can choose whether to execute one step at a time, or he can predefine break points at which execution should halt. A halted execution may be resumed from the same point, alternatively a different point to resume from can be selected. The status and results of each step or statement are shown in real-time on the procedure table display (e.g. telemetry values and command verification statuses). In addition there are several other

windows and dialogue boxes for displaying monitoring information or interacting with the data. A screenshot of the MOIS Validator was already depicted in Fig. 12.4.

The integrated man-machine interface provided by Validator enables to verify whether or not the procedure is executed successfully. Such results can be recorded at both the level of the procedure (including nested procedures) and of a particular step or statement. The validation results are updated by the user according to the activity (steps and/or substeps) that are performed (e.g. in case of expected TM values the user manually clears off the related step by a tick-mark ✓). Validation can also be automatically linked to the execution status that identifies whether a step has been successfully executed (independently of the TM value). Validation information, including comments, can be entered on the display and submitted back to the procedure databases to keep a record of procedure execution during spacecraft validation tests and simulations. Additional functionality includes a stopwatch feature to time the execution of the procedure or particular step sequences.

The MOIS Scheduler is a Mission Planning System for satellite operations and station scheduling. Scheduler integrates full resources and constraints (e.g. flight dynamics events, ground station availability) modeling capability into the procedure development and execution environments. In 'preparation' mode, this permits the generation and analysis of conflict-free schedules with time-tagged commands for on-board execution. A schedule can also be executed by linking Scheduler to the MOIS Executor and running procedures, viewing the overall execution status and results in a Gantt chart, or by using Validator to examine a specific procedure. A screen-shot of the MOIS Scheduler is provided in Fig. 14.4.

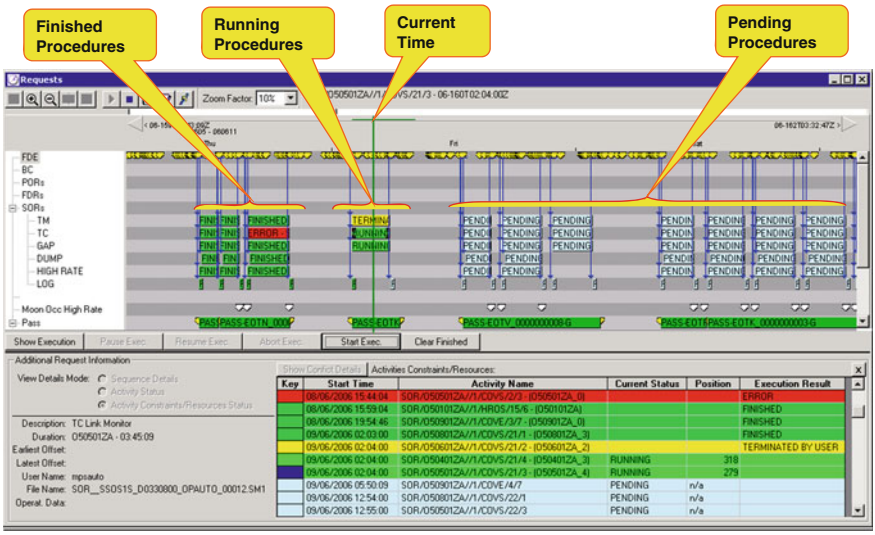


Fig. 14.4 MOIS Scheduler. © RHEA System S.A.

The MOIS Supervisor controls and monitors the MOIS Executors. It controls the CORBA connections to the SCOS-2000 Control System. The MOIS Supervisor is used to couple MOIS with SCOS for the automated execution of MOIS procedures.

Finally MOIS Library provides configuration control for all mission data stored and created within MOIS.

MOIS Publisher allows to produce and print documents of procedure listings, along with additional configuration control information such as issue identification and a record of updates.

14.2 Scheduler Resource Model

A resource model of the satellite system is defined in the MOIS Scheduler (see Sect. 14.3). The structure of the model is described in [119]. The detailed implementation of the model is beyond the scope of this manual, only some general concepts are explained. The implementation for the Flying Laptop mission can be found in [122].

The model is defined to allow automatic checks of a MOIS created schedule for conflicts, e.g. against exceeding of onboard resources (onboard battery charge status or other) or against procedures being incompatible when being executed in parallel. For this purpose status and capacity resources can be defined in the model.

Status resources have several states, which are changed by Flight Procedures being planned or by orbital events. Examples are the visibility of a ground station with the states IN_VISIBILITY and OUT_OF_VISIBILITY and the spacecraft mode with the states Safe Mode, Idle Mode, etc.

Capacity resources have a scalar capacity, which is influenced by the state of status resources. For example, a payload mode in which earth observation is performed decreases the available on-board memory, whereas a mode in which payload data is downlinked and memory is freed, increases it.

For the Flying Laptop mission, status resources are defined for the system and TTC subsystem modes, which are changed by MDE-type Flight Procedures (see Sect. 15.2.1.1). In addition so-called USAGE status resources are defined for all devices and subsystems, indicating whether a device or subsystem is being used by a procedure for something other than a mode change, for example for downlink of a picture taken by the STR. Furthermore, boolean status resources are defined for modeling ground station contact times and eclipse phases.

Capacity resources are defined for the battery State-of-Charge and on-board memory.

14.3 Mission Planning Infrastructure and Work Flow

The mission planning for FLP spacecraft missions is based primarily on the MOIS Scheduler (see Sect. 14.1), similar to the approach described in [155]. In general, two kinds of Flight Procedures will be commonly executed during the mission:

- Procedures related to the satellite bus, e.g. test or commissioning procedures, calibrations, maintenance procedures etc., which are defined by the spacecraft operators.
- Procedures for payload operations, e.g. earth observation campaigns, which are mostly requested and defined by external partner organizations.

The work flow for mission operations regarding the Flight Procedures is shown in Fig. 14.5. Both kinds of procedures are prepared in MOIS Writer and Flowcharter (step 1), based on input from the Mission Information Base (MIB). The procedures are then tested and validated on EM /FM hardware or the spacecraft simulator using the MOIS Validator (step 2). The procedures can then be imported into the MIB as command sequences. Finally, they can be imported to the MOIS Scheduler for mission planning (step 3).

Importing the tested procedures manually into the schedule is mainly used for the S/C platform related procedures, for which the parameters are defined by the spacecraft operators at IRS. Payload operations however, especially for Earth observation, can be requested by external partners via an online form. The necessary data like observation time, coordinates of the observation target and instrument settings are entered and automatically converted into so-called CRFs (Command Request File), which contain the necessary procedure information, including commands and command parameters. These files are then imported into the MOIS Scheduler (see step 3 in Fig. 14.5).For mission planning, orbital events like ground station contact times and eclipse phases need to be included in the schedule. This task is covered by the IRS flight dynamics infrastructure which was already cited in Sect. 12.2 and depicted in Fig. 12.21 and which is based on the Systems Tool Kit (STK) software developed by AGI [109]. In STK, orbital events and ground station access times can be computed and converted to the Flight Dynamic Event (FDE) format. These files are subsequently imported into the MOIS Scheduler to complete the mission plan (see also step 3 in Fig. 14.5).

Sun/Eclipse phase information, ground station visibility start/end or target visibility start/end information are pairs of orbital events imported into the MOIS

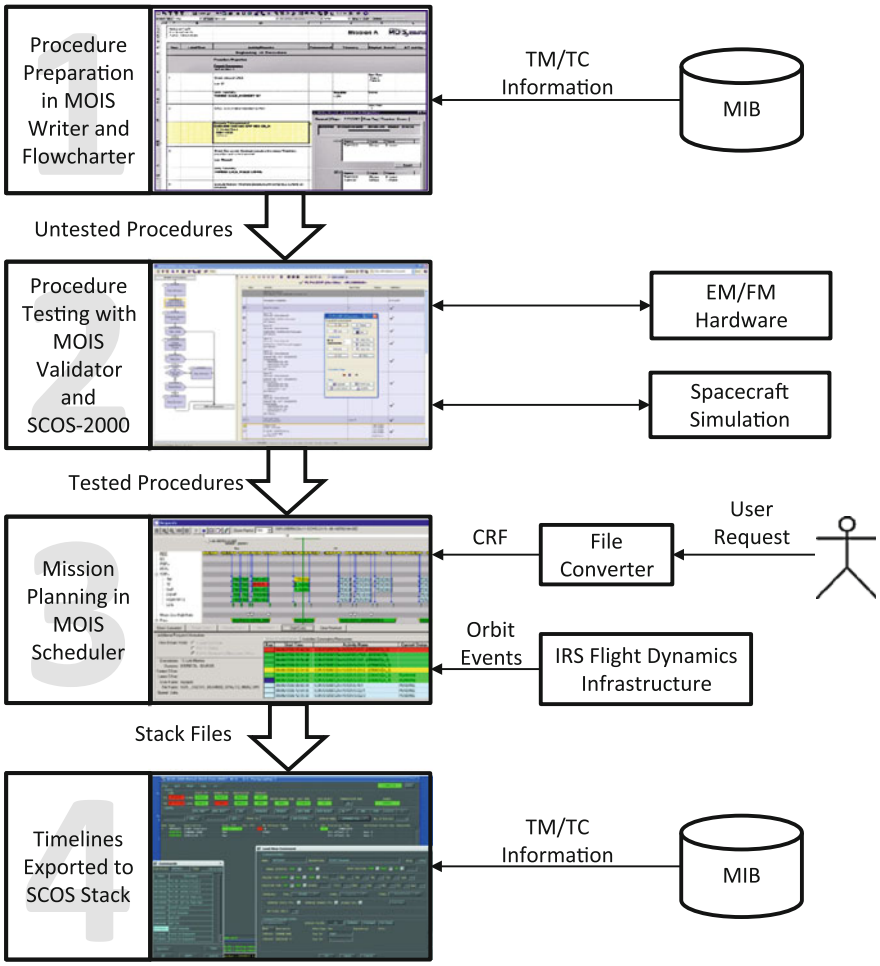


Fig. 14.5 Operations workflow. © IRS, University of Stuttgart and RHEA System S.A.

scheduler. Therefore all are included in the same FDE input file. An example for an observation of Capetown—South Africa—and ground station visibilities for IRS Stuttgart is provided to explain the process in more detail—see also Figs. 14.6 and 14.7.

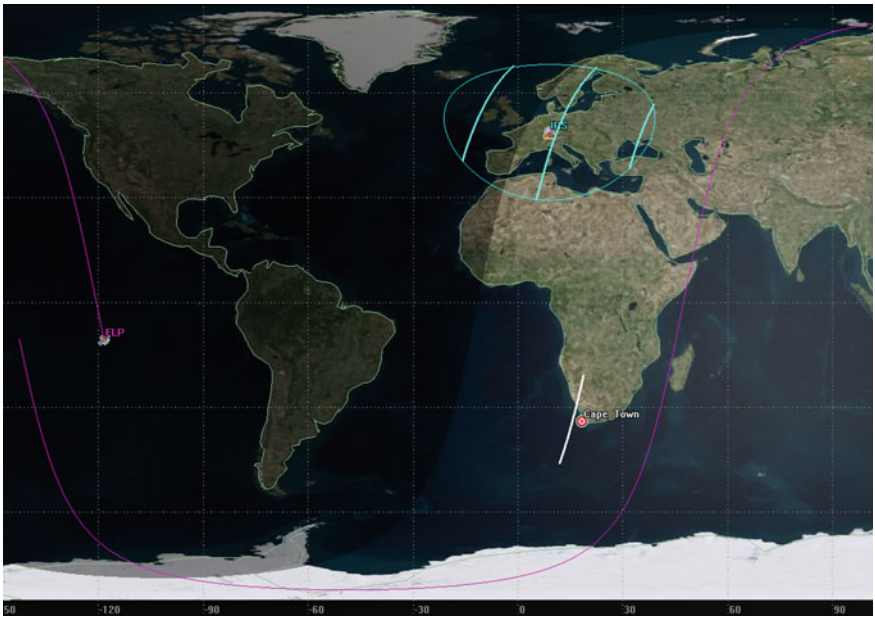


Fig. 14.6 Planning example—2D-view—with IRS and target visibilities and sun/eclipse. © Airbus DS

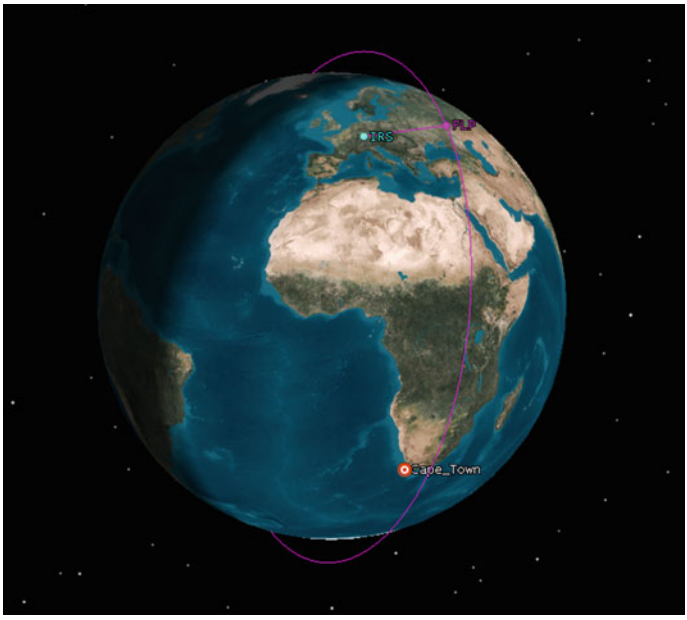


Fig. 14.7 Example of orbit over Europe and Capetown as target. © Airbus DS

The resulting Flight Dynamics Events for this scenario are listed in the text file below:

```
IRSA     1  P  15-050T09:12:36.294Z    0  IRS GS AOS
IRSL     1  P  15-050T09:17:45.201Z    0  IRS GS LOS
IRSA     1  P  15-050T10:46:28.685Z    0  IRS GS AOS
IRSL     1  P  15-050T10:56:55.915Z    0  IRS GS LOS
IRSA     1  P  15-050T12:23:08.203Z    0  IRS GS AOS
IRSL     1  P  15-050T12:30:45.897Z    0  IRS GS LOS
ECLS     1  P  15-050T08:25:00.206Z    0  Eclipse
ECLE     1  P  15-050T08:59:19.162Z    0  End of Eclipse
ECLS     1  P  15-050T10:01:48.321Z    0  Eclipse
ECLE     1  P  15-050T10:36:07.295Z    0  End of Eclipse
ECLS     1  P  15-050T11:38:36.439Z    0  Eclipse
ECLE     1  P  15-050T12:12:55.426Z    0  End of Eclipse
EOTS     1  P  15-050T09:33:40.607Z    0  EO Target Visibility
EOTE     1  P  15-050T09:40:33.632Z    0  End of EO Target Visibility
```

The entry data are in detail:

- The Event ID
- A count field: This field is ignored in the example. The FDE file contains a “1”
- A field which has to contain a “P”
- Time of event
- A field which has to contain a “0”
- A field with explanatory text

Once the orbital events and the procedures are imported into the Scheduler, the actual plan can be rearranged in order to create a conflict free schedule. Furthermore, resources like power or available on-board memory are also modeled in the Scheduler and are tracked over time to avoid that procedures are planned for which the resources are insufficient. Conflicts in the plan are detected by the Scheduler and can be resolved manually (see Sect. 14.2)

When a mission plan for the upcoming days is finished, the necessary command stack files for the execution of the procedures are generated by the Scheduler. These contain time-tagged command sequences only, which are executed when a predefined on-board time is reached. These stack files are imported to the SCOS auto stack application (step 4 in Fig. 14.5), from which they are uplinked to the spacecraft during the next ground station pass.

Procedures which are to be executed during a ground station pass, mainly contain “normal” commands which are to be executed immediately upon reception on-board and they in most cases do not contain time-tagged commands.

These procedures are not included in the files exported from the Scheduler. This is because the Scheduler exclusively supports stack file exports for procedures with

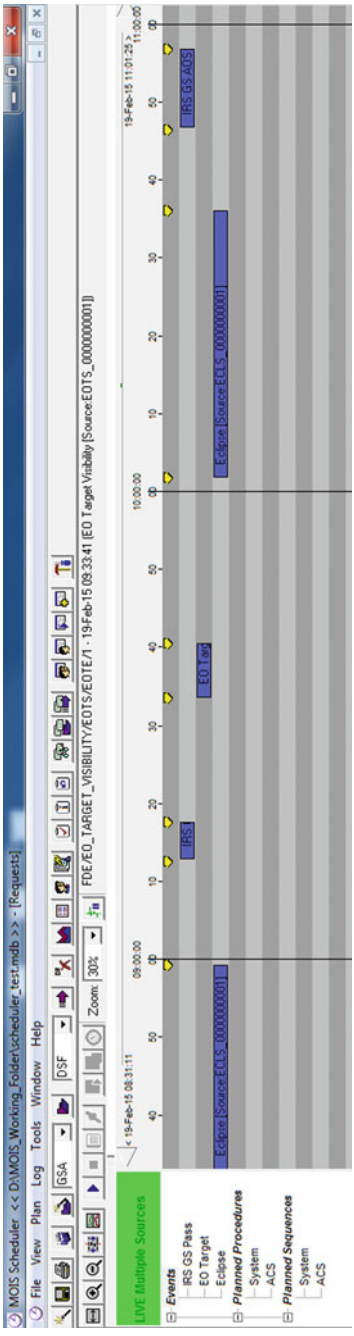


Fig. 14.8 Corresponding MOIS flight dynamics schedule. © IRS, University of Stuttgart and RHEA System S.A.

pure time-tagged commands. Procedures containing non-time-tagged commands are imported into the MIB as command sequences and are loaded on the SCOS manual stack. See Chap. 15 for details.

The events are used by the MOIS Scheduler to change the state of the corresponding resources (see Sect. 14.2).

- The resource "IRS GS Pass" is set to "IN_VISIBILITY" at IRSA.
- At IRSL it is reset to "OUT_OF_VISIBILITY".

The resources are depicted in the plan—see example in Fig. 14.8. Further resources like power or ACS attitude are not included in this simple example.

The schedule shows the ground station visibilities and the target visibility for the applied example scenario. The steps from the schedule to the exact spacecraft telecommand sequence shall be explained in more detail.

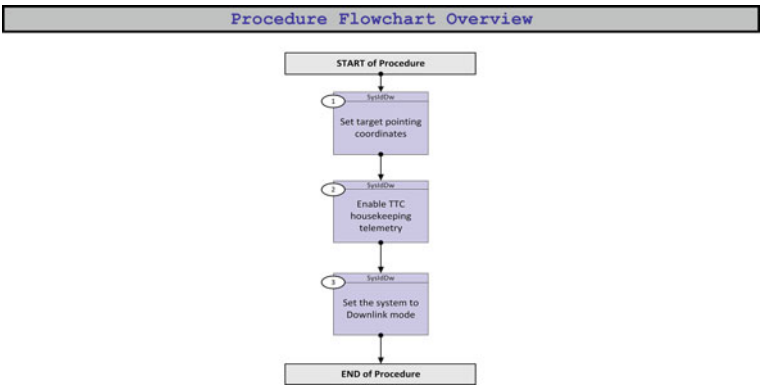


Fig. 14.9 Example procedure for execution at start of GS visibility (flow). © IRS, University of Stuttgart and RHEA System S.A.

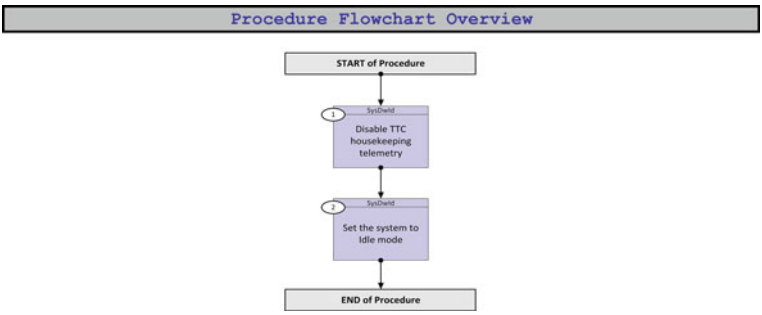


Fig. 14.10 Example procedure for execution at end of GS visibility (flow). © IRS, University of Stuttgart and RHEA System S.A.

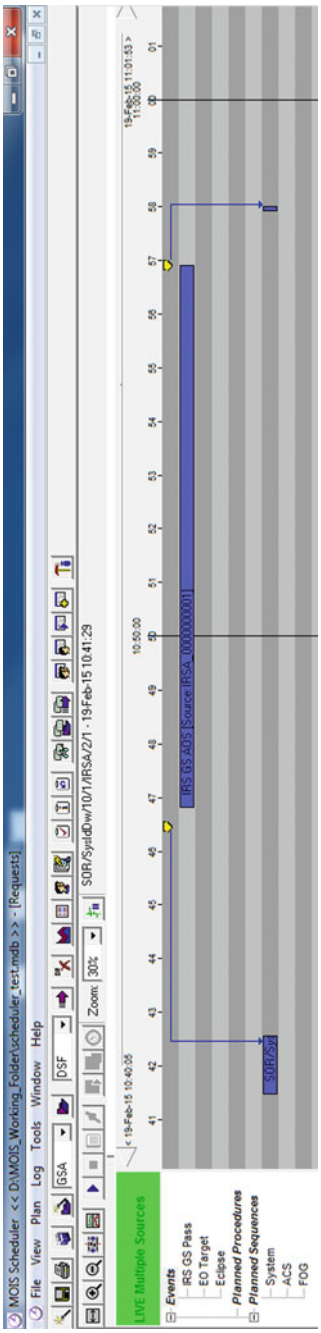


Fig. 14.11 Schedule detail for GS passover. © IRS, University of Stuttgart and RHEA System S.A.

The key technique is, that start or end of any MOIS Scheduler activity (e.g. ground station contact) can be connected to a MOIS procedure comprising spacecraft commands or sequences of commands.

Command flows can be defined, which are to be executed with an optional time offset at the start or at the end of a scheduled activity, e.g. a ground station contact—see Figs. 14.9 and 14.10 as examples. Detailing of these flows again is performed with MOIS Writer—see Fig. 14.3 and the Flight Procedure examples in Chap. 15.

Importing the released MIB into SCOS permits to use the sequence command stacks as Scheduler time-tagged version from the mission planning or as direct commands.

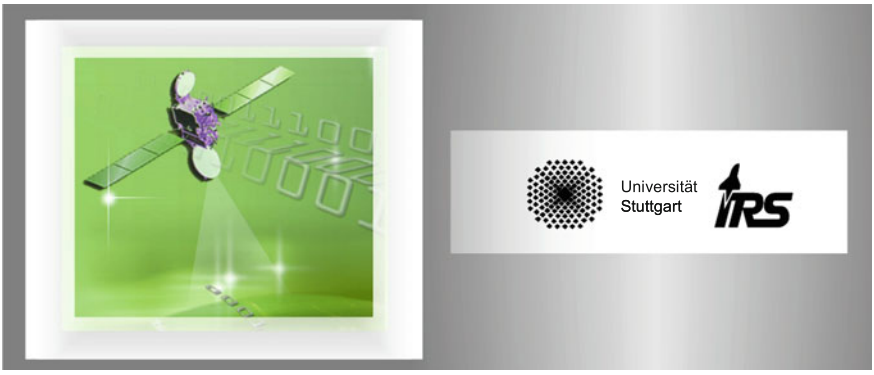
A final example of a detailed scheduled GS passover with

- planned sequences on system level SOR "System Idle to Downlink"—see Fig. 14.9
- and SOR "System Downlink to Idle"—Fig. 14.10
- and with according delta times for sequence execution

can be found in Fig. 14.11.

Chapter 15
Flight Procedures

Kai-Sören Klemich and Jens Eickhoff



Abstract This chapter describes how flight procedures for an FLP-based satellite are defined—again based on the ground infrastructure subsystems SCOS and MOIS. Procedure types and the naming conventions are provided as well as a detailed example. A table of the predefined flight procedures for the satellite platform including a short description is provided in the book's annexes.

Keywords FLP flight procedures · Flight procedure types · Naming conventions · Flight procedure example

The FLP Flight Procedures (FP) are defined using the MOIS toolkit which already was introduced in Chap. 14. Besides a more detailed explanation on the Flight Procedure workflow, a Flight Procedure example and an overview on the Flight Procedure catalog are provided—together with some definition of terms.

K.-S. Klemich (✉)
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: klemich@irs.uni-stuttgart.de

J. Eickhoff
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_15

15.1 Definitions for FLP Flight Procedures

- **Flight Procedure (FP)** An (FP) for short—is implemented in MOIS and thereafter is exported into the SCOS Mission Information Database (MIB) as a command sequence. It is to be executed—i.e. planned and triggered—by a spacecraft operator (see also Chap. 14). FPs do not include complex on-board activities which are triggered by a single command, as for example mode changes of low level objects (like device handlers). Due to restrictions in the MOIS/SCOS infrastructure FPs cannot call sub-procedures.
- **Formal Parameter** Formal parameters are used in FPs to allow adaptation of command parameters in a procedure each time it is applied. An example would be the coordinates for a target pointing mode change FP. These coordinates are different for each observed target. Formal parameters of a procedure can be compared to arguments of a function in programming.
- **Statement** A statement in a MOIS procedure is defined as follows:
  "Each → step usually contains one or more (but sometimes zero) statements. A statement is the basic information object in a procedure, and could be for example a telemetry or telecommand, or a comment. All other procedure structure is defined to organise or group statements into a meaningful order."[1]
- **Step** A step in a MOIS procedure is defined as follows:
  "A procedure may contain one or more steps. The concept of a step is used to help break up a procedure into small manageable sections, enabling each macro-process to be broken up into micro-processes, at a level higher than the individual telecommand or telemetry. A step is used to group one or more → statement objects, such as telecommands and telemetry or comments, as a single entity (see footnote 1)."

15.2 Types of Procedures

There exist several different types of procedures for the FLP, which can be distinguished by their content or by their usage scenario. These types are explained below.

15.2.1 Types of Procedures by Content

This section groups the procedure types by their content.

[1]Cited from the MOIS online help.

15.2.1.1 High-Level Mode Change Procedures (MDE)

Mode change procedures are implemented in dedicated PUS Services in the FLP OBSW. The general idea is that a mode of a high level on-board object (system and subsystems) is defined in a table consisting of several child objects. Examples for such mode tables are all the subsystem mode change Tables 4.5, 5.8, 6.12, 7.16 and 8.6.

If a mode change of the subsystem is commanded, the modes of the subsystem's controller's child objects are changed accordingly by the OBSW in the order they appear in the table. To perform a more complex mode change, several of these table based mode changes can be performed in a sequence. For the entire system and subsystems, these tables and sequences can be defined freely. The only restriction is a maximum number of tables, table entries, sequences, etc. to avoid an overrun in the OBSW. As the changes are performed in such a way that the next step is only performed if the previous step was executed successfully, the mode change procedures offer inherent success checks.

Top level (system) mode changes, e.g. from system safe mode to idle mode, consist of different subsystem level MDE procedures, e.g. from ACS Safe Mode to ACS Idle Mode. The lowest level changes triggered by MDE procedures are mode changes of controllers and device handlers.

Only System and TTC subsystem mode changes shall be commanded by the operators during nominal operations. Thus all these nominal (non-contingency) procedures are of type MDE. They are actually implemented in MOIS as FLP Flight Procedures to be used during operations and to allow *System* mode changes and TTC subsystem mode changes.

The FPs of this type influence the MODE_SYSTEM or MODE_TTC_SUBSYS status resources in the Mission Planning System (MPS)—see Sect. 14.2.

15.2.1.2 General Procedures (PRC)

A General Procedure (PRC) can be any kind of operational procedure which implies more than a single Service 2 device command. Also if a procedure is merely a mode change of a single object and—where applicable—its children, it is an MDE procedure. The PRCs may be triggered by single commands to any on-board object or by several commands which are defined in a MOIS sequence and are uplinked as normal or time-tagged commands.

In general, there may be more than one way to execute a procedure. Most procedures should be executable by single low level commands, which can be combined in a MOIS procedure. If the procedures are long, are used frequently and/or are easy to automate, hard coded macro commands may be introduced into the OBSW to simplify the execution of these procedures. Thus to achieve the same goal, there may be standard high level procedures and procedures consisting of many low level commands for contingency situations.

As PRCs are generally FPs, they may consist of high and low level commands as well as low level object mode changes (e.g. if a device is to be commissioned in a system mode in which it is usually not switched on). However, a PRC may not trigger MDEs (these should be defined separately) and a PRC may not call other PRCs.

The FPs of this type influence the USAGE status resources in the Mission Planning System (MPS)—see Sect. 14.2.

15.2.1.3 Contingency Procedures (CTG)

Contingency Procedures are procedures for contingency cases which are not handled in the FDIR or which can be triggered from ground for troubleshooting purposes. They are PRCs for non-nominal cases.

A trained operator may use CTG procedures or single commands in contingency situations, which trigger lower level mode changes.

15.2.2 Types of Procedures by Usage Scenario

The FPs furthermore can be grouped by their usage scenario as detailed below.

15.2.2.1 Procedures Executed Used During Ground Contact

Procedures used during ground station contact are FPs, which are only used when live TM and TC contact is established between Mission Control Center and the satellite. The procedures are implemented in such a way that they can be used with the manual stack in SCOS-2000. During the execution of these procedures, commands are sent manually and the received reply TM from the spacecraft is available for the operators. These procedures may therefore contain TM checks, which are to be executed by the operator before the next command is submitted. They may also contain IF conditions, for which the following rules apply:

- There must be one and only one branch in the procedure flowchart, which contains currently executed steps, i.e. commands and TM checks.
- Each IF-condition must be configured in such a way that both of its two branches lead to another step of the main branch without any steps in between. This may be the following step or any other step further downstream the current IF-condition step. For an example of such a semi-linear flow see Fig. 15.2.

This feature can be used for example to attach TM checks to an IF-condition and to proceed with the procedure normally when successful or jump to the end of the procedure otherwise[2]—without performing any steps in between.

This is necessary to assure that the command sequence which is derived from the procedure and is available in SCOS stays "linear" so that for example no commands have to be repeated for an unknown amount of times and that the order of commands in the sequence is correct in any case. Thus the only action that may become necessary once the sequence is loaded in the manual stack, is to delete a certain amount of commands from the stack. This is required in order to avoid errors caused by manipulating stacks manually.

This kind of procedure may also contain execution or release of time-tagged commands and interlocks if useful. For example to safely deactivate a device—which is used during the procedure execution—in case where the link is lost unexpectedly. The FP may include time-tagged commands being executed just after the ground contact and thus can deactivate the device after ground contact end or in case of an unexpected contact loss. Such a procedure may not contain any loops.

In general, procedures in which one of the following types of action is performed, should be performed during ground contact wherever possible in order to allow live TM checks:

- Activating of devices for the first time during commissioning, which may be critical
- Switching to previously unused system modes for the first time during commissioning
- Setting configuration parameters of controllers, on-board TM storage, TC scheduling, timers, etc., which should be checked by TM checks immediately
- Performing any kind of contingency measures like reboots, resets, changes to the redundant side, etc.
- Payload data takes which may be interesting during target pointing to a ground station.

15.2.2.2 Procedures Executed Without Ground Contact

It is possible to define procedures which are executed when no live TM/TC link is available. As there is no possibility to check TM on-board, these procedures may not contain any TM checks, branches or loops. For these procedures the execution based auto stack is used, which implies that all commands must have an execution time tag assigned. These procedures will be executed autonomously on board and must be uplinked to the satellite during a ground station contact beforehand.

[2]Where potentially necessary final configurations may be set.

Procedures in which one of the following types of action is performed, should be available as procedures without ground station contact:

- Performing all sorts of payload data takes and calibrations which need to be executable at all times
- Performing mode changes when they have been tested before
- Preparing devices for operations, e.g. heating them up, configuring them.

15.3 Flight Procedure Naming Convention

As mentioned, FPs implemented in MOIS are then are exported into text files for import into the MIB as command sequences (see also Chap. 14). Their naming convention therefore follows the general guidelines in the MIB (see also Sect. 16.2). A FP name is 8 characters long and follow the pattern XYTGPzzz, where

- X is the subsystem, for example "A" for ACS, "T" for TCS, etc.
- Y is the "subsubsystem", for example "F" for FOG or "H" for heaters, etc.
- T is the type of DB entry and is set to "Q" for all sequences, i.e. for all FPs.
- G indicates whether this procedure is used during ground station contact (1) or without ground station contact (0), i.e. the type of the procedure by usage scenario.
- P indicates whether this procedure is an MDE (0), a PRC (1–5) or a CTG (6–9), i.e. the type of procedure by content, see Sect. 15.2.1.
- zzz is a counter used to identify single sequences.

For more details please also refer to the FLP Procedure Guidelines [121]. As an example, the name of a nominal FP of the RWL which is to be executed during ground contact could be "ARQ11005".

15.4 Flight Procedure Example

The following example depicts the Flight Procedure for the initial STR start-up and checkout of the Star Tracker during the Commissioning Phase of the FLP platform in orbit. The FP is provided with

- its descriptive header (Fig. 15.1)
- the flow block-diagram as overview (Fig. 15.2)
- and the detailed FP body with the diverse steps and commands (Fig. 15.3).

Procedure Summary

Objectives

This procedure is used as the first checkout test of the STR and shall be performed during GS visibility. For this purpose, the STR HK TM is activated, the STR Handler is switched on and its TM is checked. If any value is out of limits, the STR is switched off and the failure needs to be analysed. At first the constraints and spacecraft configuration are checked and the STR TM is activated. Then only the DPU is switched on using the STR Handler and the amperage and temperature TM is checked. Then CHU A is added and the changed amperage, the CHU's temperature and the calculated attitude quaternion are checked. This is repeated for CHU B, with CHU A being switched off. Finally, the STR Handler is switched off, the HK TM is disabled and its handler's health flag is set to "healthy".

Summary of Constraints

GS Visibility, Battery SoC >80%, STR Component Temp: -30°C to +60°C

Spacecraft Configuration

Start of Procedure

System in Safe Mode, Nominal TTC Tx active, STR Handler off and healthy, necessary housekeeping TM: 5200,5800,4900

End of Procedure

Same as before Start of Procedure

References

Requirements 0

ASQ11000

Referenced Displays

ANDs	GRDs	SLDs
TYA00000		
YMA10000		
PBA00000		
PPA00002		
TSA00000		
ASA00000		
YMA20000		

Configuration Control Information

DATE	FOP ISSUE	VERSION	MODIFICATION DESCRIPTION	AUTHOR	SPR REF
11/05/2015		1	Created	flatsat	
11/05/2015		2	Changed Procedure from Sequence to Generic	flatsat	
11/05/2015		3	Thorough check of all statements including some minor fixes	flatsat	
12/05/2015		4	added STR Handler Mode ON at the beginning, changed current limits in TM checks	flatsat	

Fig. 15.1 FP header example. © IRS, University of Stuttgart

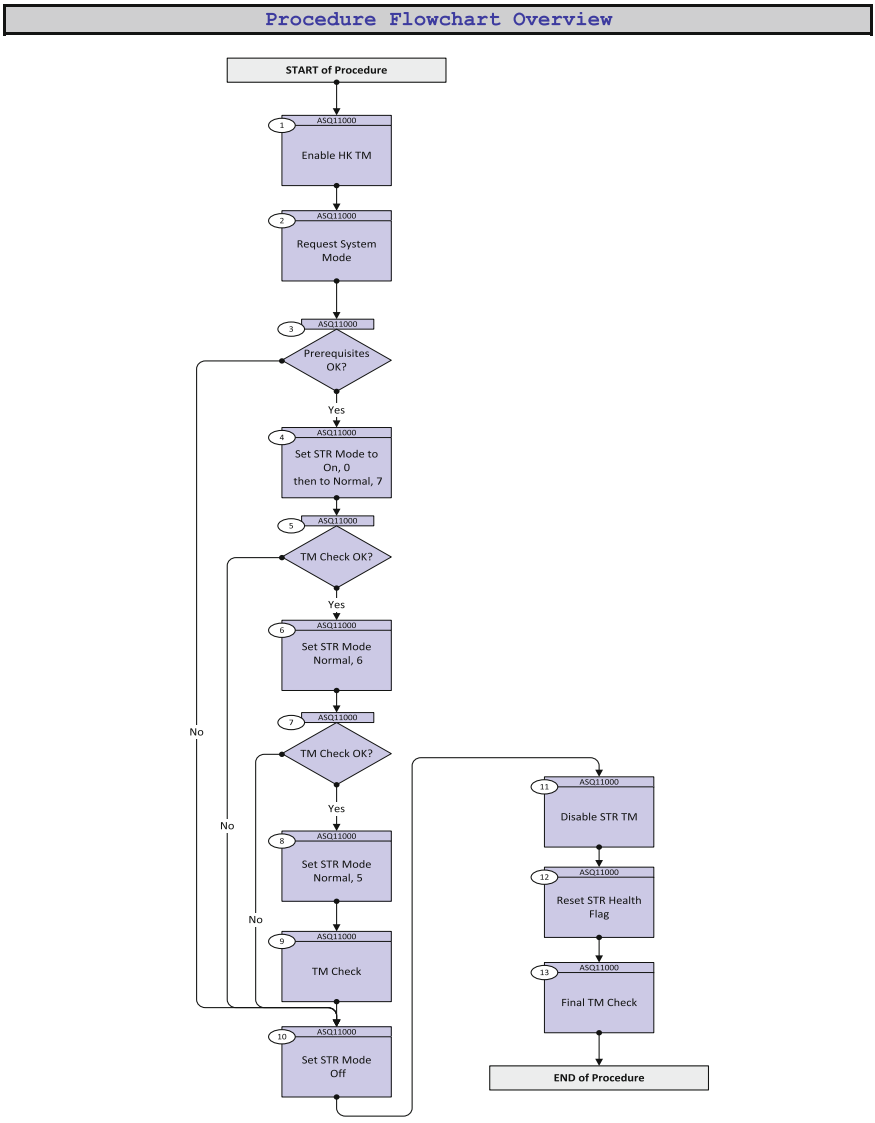


Fig. 15.2 FP flowchart example. © IRS, University of Stuttgart

The detailed TCs of the FP body steps shall not be explained here. The interested reader should be able to understand the example procedure using the "Objectives" description on the procedure header and by comparing the FP Body step by step with the provided FP flow of Fig. 15.2.

Step	Label/Time	Activity/Remarks	Telecommand	Telemetry	Display/ Branch	Requirements
		Beginning of Procedure				
		ASQ11000 **STR Test with GS Vis**				
1		Enable HK TM			Next Step: 2	
		Execute Telecommand **DSC00005 Srv(3,5) Enable HKrepGen** TC Control Flags: GBM IV DSE --Y -- --- Command Parameter(s): **DSP00007 = 2 <dec>** **DSP00000 = STR Housekeeping** **DSP00000 = STR Attitude TM**	DSC00005		**TC**	
2		Request System Mode			Next Step: 3	
		Execute Telecommand **YMC22002 Srv(200,2) Read Mode Sys** TC Control Flags: GBM IV DSE --Y -- --- Command Parameter(s): **YMP00012 = System** (Def)	YMC22002		**TC**	
3		Prerequisites OK? type: [If]			Next Step: Yes 4 No 10	
		Verify Telemetry **TYTCTP25 COMP_TEMP_STR** >= -30.0 degC <= 60.0 degC Packet Information: **TYHK5800temp**		TYTCTP25 >= -30.0 degC <= 60.0 degC	AND=TYA00000	
		Verify Telemetry **YMTMD072 SYSTEM_MODE** = Safe Packet Information: **YMEV1572mode**		YMTMD072 = Safe	AND=YMA10000	
		Verify Telemetry **PBTSCC03 BATT_SOC_PSS** >= 80.0 % Packet Information: **PYHK4900elec**		PBTSCC03 >= 80.0 %	AND=PBA00000	
4		Set STR Mode to On, 0 then to Normal, 7			Next Step: 5	
		Set STR Handler to a mode where both CHUs are off				
		Execute Telecommand **YMC20001 Srv(200,1) Set Mode Devi** TC Control Flags: GBM IV DSE --Y -- --- Command Parameter(s): **YMP00000 = STR Handler** **YMP00001 = On** **YMP00002 = 0 <dec>**	YMC20001		**TC**	
		Execute Telecommand **YMC20001 Srv(200,1) Set Mode Devi** TC Control Flags: GBM IV DSE --Y -- --- Command Parameter(s): **YMP00000 = STR Handler** **YMP00001 = Normal** **YMP00002 = 7 <dec>**	YMC20001		**TC**	

Fig. 15.3 FP body with command sequence example. © IRS, University of Stuttgart

Step	Label/Time	Activity/Remarks	Telecommand	Telemetry	Display/ Branch	Requirements
5		TM Check OK? type: [If]			Next Step: Yes 6 No 10	
		Verify Telemetry **PPTFSC10 FUSECURR PL,TMTX** > 0.59 A < 0.75 A Packet Information: **PYHK4900elec**		**PPTFSC10** > 0.59 A < 0.75 A	AND=PPA00002	
		Verify Telemetry **YMTMD012 STR_HANDLER_MODE** = Normal Packet Information: **YMEV1512mode**		**YMTMD012** = Normal	AND=YMA10000	
		Verify Telemetry **YMTSM012 STR_HANDLER_SMOD** = 7 <dec> Packet Information: **YMEV1512mode**		**YMTSM012** = 7 <dec>	AND=YMA10000	
		Verify Telemetry **TSTSTR00 STR_CPU_TEMP_C** >= -30.0 degC <= 60.0 degC Packet Information: **TSHK5200tbus**		**TSTSTR00** >= -30.0 degC <= 60.0 degC	AND=TSA00000	
		Verify Telemetry **TSTSTR01 STR_POWER_T_C** >= -30.0 degC <= 60.0 degC Packet Information: **TSHK5200tbus**		**TSTSTR01** >= -30.0 degC <= 60.0 degC	AND=TSA00000	
		Verify Telemetry **TSTSTR02 STR_CHU_A_T_C** >= -30.0 degC <= 60.0 degC Packet Information: **TSHK5200tbus**		**TSTSTR02** >= -30.0 degC <= 60.0 degC	AND=TSA00000	
		Verify Telemetry **TSTSTR03 STR_CHU_B_T_C** >= -30.0 degC <= 60.0 degC Packet Information: **TSHK5200tbus**		**TSTSTR03** >= -30.0 degC <= 60.0 degC	AND=TSA00000	
6		Set STR Mode Normal, 6			Next Step: 7	
		Set STR Handler to a mode where CHU A is on				
		Execute Telecommand **YMC20001 Srv(200,1) Set Mode Devi** TC Control Flags: GBM IV DSE --Y -- --- Command Parameter(s): **YMP00000 = STR Handler** **YMP00001 = Normal** **YMP00002 = 6 <dec>**	YMC20001		**TC**	
7		TM Check OK? type: [If]			Next Step: Yes 8 No 10	
		Verify Telemetry **PPTFSC10 FUSECURR PL,TMTX** > 0.62 A < 0.79 A Packet Information: **PYHK4900elec**		**PPTFSC10** > 0.62 A < 0.79 A	AND=PPA00002	
		Verify Telemetry **YMTMD012 STR_HANDLER_MODE** = Normal Packet Information: **YMEV1512mode**		**YMTMD012** = Normal	AND=YMA10000	
		Verify Telemetry **YMTSM012 STR_HANDLER_SMOD** = 6 <dec> Packet Information: **YMEV1512mode**		**YMTSM012** = 6 <dec>	AND=YMA10000	
		Verify Telemetry **ASTQUA00 STR_A_QUAT** <> 0.0 <dec> Packet Information: **ASHK1500atti**		**ASTQUA00** <> 0.0 <dec>	AND=ASA00000	
		Verify Telemetry **ASTQUA01 STR_A_QUAT** <> 0.0 <dec> Packet Information: **ASHK1500atti**		**ASTQUA01** <> 0.0 <dec>	AND=ASA00000	

Fig. 15.3 (continued)

Step	Label/Time	Activity/Remarks	Telecommand	Telemetry	Display/ Branch	Requirements
		Verify Telemetry **ASTQUA02 STR_A_QUAT** <> 0.0 <dec> Packet Information: **ASHK1500atti**		**ASTQUA02** <> 0.0 <dec>	AND=ASA00000	
		Verify Telemetry **ASTQUA03 STR_A_QUAT** <> 0.0 <dec> Packet Information: **ASHK1500atti**		**ASTQUA03** <> 0.0 <dec>	AND=ASA00000	
8		Set STR Mode Normal, 5			Next Step: 9	
		Set STR Handler to a mode where CHU B is on				
		Execute Telecommand **YMC20001 Srv(200,1) Set Mode Devi** TC Control Flags: GBM 1V DSE --Y -- --- Command Parameter(s): **YMP00000 = STR Handler** **YMP00001 = Normal** **YMP00002 = 5 <dec>**	**YMC20001**		**TC**	
9		TM Check			Next Step: 10	
		Verify Telemetry **PPTFSC10 FUSECURR PL,TMTX** > 0.62 A < 0.79 A Packet Information: **PYHK4900elec**		**PPTFSC10** > 0.62 A < 0.79 A	AND=PPA00002	
		Verify Telemetry **YMTMD012 STR_HANDLER_MODE** = Normal Packet Information: **YMEV1512mode**		**YMTMD012** = Normal	AND=YMA10000	
		Verify Telemetry **YMTSM012 STR_HANDLER_SMOD** = 5 <dec> Packet Information: **YMEV1512mode**		**YMTSM012** = 5 <dec>	AND=YMA10000	
		Verify Telemetry **ASTQUA04 STR_B_QUAT** <> 0.0 <dec> Packet Information: **ASHK1500atti**		**ASTQUA04** <> 0.0 <dec>	AND=ASA00000	
		Verify Telemetry **ASTQUA05 STR_B_QUAT** <> 0.0 <dec> Packet Information: **ASHK1500atti**		**ASTQUA05** <> 0.0 <dec>	AND=ASA00000	
		Verify Telemetry **ASTQUA06 STR_B_QUAT** <> 0.0 <dec> Packet Information: **ASHK1500atti**		**ASTQUA06** <> 0.0 <dec>	AND=ASA00000	
		Verify Telemetry **ASTQUA07 STR_B_QUAT** <> 0.0 <dec> Packet Information: **ASHK1500atti**		**ASTQUA07** <> 0.0 <dec>	AND=ASA00000	
10		Set STR Mode Off			Next Step: 11	
		Execute Telecommand **YMC20001 Srv(200,1) Set Mode Devi** TC Control Flags: GBM 1V DSE --Y -- --- Command Parameter(s): **YMP00000 = STR Handler** **YMP00001 = Off** **YMP00002 = 0 <dec>**	**YMC20001**		**TC**	
11		Disable STR TM			Next Step: 12	

Fig. 15.3 (continued)

Step	Label/Time	Activity/Remarks	Telecommand	Telemetry	Display/ Branch	Requirements
		Execute Telecommand **DSC00005 Srv(3,5) Enable HKrepGen** TC Control Flags: GBM IV DSE --Y -- --- Command Parameter(s): **DSP00007 = 2 <dec>** **DSP00000 = STR Attitude TM** **DSP00000 = STR Housekeeping**	DSC00005		TC	
12		Reset STR Health Flag			Next Step: 13	
		The flag is automatically set to "External Ctrl" when the handler mode is changed at the start of the procedure				
		Execute Telecommand **YMC20101 Srv(201,1)SetHealFlagDev** TC Control Flags: GBM IV DSE --Y -- --- Command Parameter(s): **YMP00000 = STR Handler** **YMP00003 = Healthy**	YMC20101		TC	
13		Final TM Check			Next Step: END	
		Verify Telemetry **YMTMD012 STR_HANDLER_MODE** = Off Packet Information: **YMEV1512mode**		YMTMD012 = Off	AND=YMA10000	
		Verify Telemetry **YMTHE012 STR_HANDLER_HEAL** = Healthy Packet Information: **YMEV4012heal**		YMTHE012 = Healthy	AND=YMA20000	
End of Procedure						

Fig. 15.3 (continued)

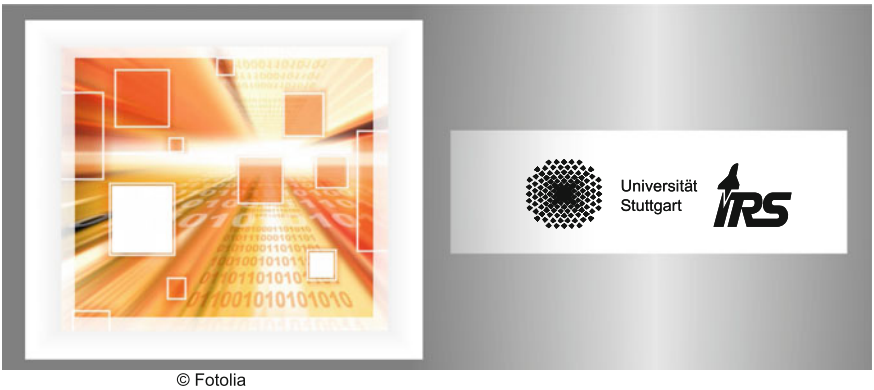
15.5 List of FLP Flight Procedures

The list of FPs which are defined for the FLP Generation 1 plus some specific Nominal Observation Mode FPs for the "Flying Laptop" mission is provided in annex Sect. 17.4. The FPs are grouped according to the corresponding system/subsystem and subsequently by equipment.

During the flight phase further FPs can be defined. For other FLP based missions with modified ACS/AOCS, other payloads and functional changes (e.g. solar array drive), the FP list and definitions may need adaptations.

Chapter 16
FLP Mission Information Database

Kai-Sören Klemich and Jens Eickhoff



© Fotolia

Abstract The chapter provides an introductory overview on the Mission Information Database which is the underlying foundation of the Mission Control System. The definition of telecommand packets, telemetry packets, their parameters and more details are covered. The lists of FLP telecommand packets, telemetry packets and Event telemetry packets are provided in the book's annexes.

Keywords Mission information database · Definition of telecommand packets · Definition of telemetry packets · TC/TM parameters and formats · TC sequences · TM displays

K.-S. Klemich (✉)
Institute of Space Systems, University of Stuttgart, Stuttgart, Germany
e-mail: klemich@irs.uni-stuttgart.de

J. Eickhoff
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_16

16.1 Introduction

The FLP supports commanding of the spacecraft via the ECSS Packet Utilization Standard [37]. The telecommand (TC) and telemetry (TM) packets and the parameters included in the packets as well as the data types, their calibrations and decalibrations are defined in the underlying database of the Mission Control System (MCS).

For the FLP the Mission Control System preferably is a ESA/ESOC SCOS-2000 system (see [85–91]) or any of its successor systems like Terma CCS5 (see [92, 93]) or later a European Ground Systems Common Core (EGS-CC) based system [94].

The database with all the mentioned definitions is called Mission Information Base (MIB). The first FLP mission "Flying Laptop" uses a SCOS system. The MIB is setup as a MySQL database. The interface between the database and SCOS is described in the SCOS-2000 Database Import ICD [86]. The definition of all relevant TC/TM packets, the information of TC and TM parameters and many further features like SCOS alphanumeric or graphical displays etc. can be made easiest by using a dedicated MIB editor. Multiple such SCOS MIB editors are available from diverse missions as the EEMCS Database Application [95] used in the ESA missions CryoSat and GOCE or the SCOS-2000 Database Editing Subsystem [96] used for the Integral mission.

In the first FLP mission "Flying Laptop" at the University of Stuttgart the mySQL MIB database content is exported to text files, which are then imported into SCOS-2000 and the MOIS tool suite (see Sect. 12.1.1) for operational use.

In this section, the contents of the MIB and the naming conventions for the FLP are described. As the MIB is defined by a SCOS-2000 ICD, many concepts are correlated to specific SCOS-2000 features. As several of these features are not used by the FLP mission, only the used MIB tables are discussed here.

16.2 General Concepts

Some concepts which are important to understand are introduced here.

Parameters
A parameter in the MIB is a variable with a fixed data type (see Sect. 16.7), which can be found in the telemetry (TM parameter) or in a telecommand (TC parameter). TM packets and TCs can contain zero or more parameters. Parameters have a type (integer, float etc.).

Calibration/Decalibration
Parameters can be calibrated (TM) or decalibrated (TC). For TM parameters, calibration means that the value of a parameter is not displayed as the actual raw value found in the TM packet, but as an "engineering value" derived from the raw data through a computational conversion.

Similarly, decalibration for TC parameters means that an engineering value can be entered when submitting a TC and the corresponding raw value is automatically calculated by SCOS-2000.

Calibration and decalibration can in general be numerical, e.g. by converting a raw integer value to a float value according to a given formula, or can be textual, e.g. by converting the integer "0" to the string "Off" and "1" to the string "On". For the FLP numerical calibrations are rarely used, as all numerical parameters are handled already in engineering values on board the spacecraft, i.e. in SI units. Therefore, most numerical TM parameters are downlinked as engineering values already. In these cases, a numerical calibration is only used to display the correct unit of the values on the SCOS-2000 displays.

Command Sequences
Command sequences can consist of commands or other sequences which are executed (submitted to the spacecraft) by the MCS in the specified order under certain conditions. They can also include so-called formal parameters, which can be used as input arguments for the sequence and their values can be assigned to e.g. TC parameters.

TM Packets
TM parameters are found in TM packets. There are two types of packets, fixed and variable packets. Fixed packets always have the same structure, while variable packets contain fields defining the amount of further following fields. Variable packets are necessary for some PUS defined services, e.g. to report existing housekeeping report definitions (see Sect. 2.4.3). However, for the FLP mainly fixed packets are used, especially for housekeeping TM.

Displays
In SCOS-2000 telemetry parameters can be displayed in different ways. TM parameters in fixed packets are displayed by the TM Desktop application (see [88]). For this application, the displays are defined in the MIB. The following types of displays exist (see also Figs. 12.2 and 12.3):

- Alphanumeric displays displaying up to 64 parameter names and their (calibrated) values
- Graphical displays displaying a graph of one or several parameters as a function of the time
- Scrolling displays, displaying a line of up to five parameters, which is scrolled down as more and more packets containing these parameters are received.

TM parameters from variable packets can only be displayed using the VPD (Variable Packet Display) application (see [89]).

TC Acknowledge
Depending on the command flags of a TC it will be acknowledged by the OBSW back to ground. In the MIB, commands can be associated to different verification stages, which are checked and displayed by the SCOS-2000 command history. When the expected success messages (see Sect. 2.4.1) have been received, the

command is considered to have been successful and is marked green. If an acknowledge failure message is received, the command is marked red (see [90]).

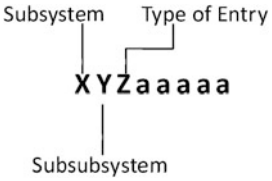
Limit Checks

The values of TM parameters are automatically checked by SCOS-2000. If certain limits which are defined in the MIB are violated, the COS ground monitoring will display these parameter values in highlighted color. The Out Of Limits Display application (see [91]) is used to get an overview of the parameters which are out of limits.

Such ground limits are defined in the MCS and ground monitoring must not to be confused with parameter limit checks on board being cyclically performed by the OBSW (on-board monitoring). Depending on the parameter the ground limits can be defined more or less constrained than the flight limits on board. Not for all parameters that are checked on board additional ground limits must be defined.

MIB entry format

There exists a general eight character name string format, which is used to identify all sorts of MIB entries, e.g. TC parameters, TCs, TM parameters etc.



The general format code is "**XYZaaaaa**", where

- **X** is a character specifying the subsystem the entry belongs to, for example ACS, TCS, etc. The possible characters for the subsystems are provided in Table 16.1.
- **Y** is a character specifying the "subsubsystem", which can be a device (for example FOG, MICS, PCDU, etc.), a certain software instance (FDIR, OBSW) or the subsystem level. The possible characters are also provided in Table 16.1.
- **Z** is the type of DB entry, for example a command parameter, a command, a TM parameter, a TC, a user defined constant, etc. Options are listed in Table 16.2.
- **aaaaa** is used to identify dedicated instances of the DB entries, for example a dedicated command or a parameter. The exact format is defined separately for each type of entry.

The possible system/subsystem (X/Y) character combinations for the FLP are listed in Table 16.1.

The diverse entities in SCOS (TM parameters, TC parameters, etc.) are distinguished by the third character (see Table 16.2):

Example:

- ARTMCR02 is a TM parameter of the ACS subsystem. In particular a parameter of reaction wheel number 2.

Table 16.1 SCOS entry ranges by subsystem

Subsystem	Character	ID Range
ACS	**A**	**10–19**
FOG	AF	10
GPS	AG	11
MGM	AM	12
MGT	AQ	13
RW	AR	14
STR	AS	15
SuS	AU	16
Subsystem	AY	18–19
CDH	**D**	**20–29**
CCSDS-Board	DC	20
I/O-Board	DI	21
Memory	DM	22–23
OBC	DO	24–25
OBSW	DS	26–27
Subsystem	DY	28–29
PL	**L**	**30–39**
PLOC	LR	35–37
Subsystem	LY	38–39
PSS	**P**	**40–49**
Battery	PB	40
PCDU	PP	41–47
Solar Panels	PS	48
Subsystem	PY	49
TCS	**T**	**50–59**
Heater	TH	50–51
Sensors	TS	52–53
Subsystem	TY	58–59
TTC	**X**	**60–69**
TTC N	XN	60–61
TTC R	XR	62–63
Subsystem	XY	68–69
Satellite System	**Y**	**70–79**
FDIR	YF	70–73
System Manager	YM	74–75
On-Board Queue (Scheduler)	YS	76
Time Managing	YT	77
System	YY	78–79

Table 16.2 Options for the type of DB entries

Type of Entry	ShortName
TM Parameter	T
Valid Flag TM Parameter	V
TM Parameter Numerical Calibration	N
TM Parameter Textual Calibration	X
TC (Packet)	C
TC Parameter	P
TC Parameter Numerical Decalibration	I
TC Parameter Textual Decalibration	Z
TC Parameter Range Set	R
TC Sequence	Q
Alphanumerical Display	A
Graphical Display	G
Scrolling Display	S

16.3 Telecommands and TC Sequence Tables

Telecommands generally consist of a packet and a data field header, the actual command data, containing zero or more command parameters, and a CRC field at the end (see Sect. 2.2). The basic definition of a TC is done in the CCF table (Command Characteristics File) in SCOS. In this table, the commands are defined with a name (see Sect. 16.2), a description, their APID (see Sect. 2.3), their PUS Service type and subtype (see Sect. 2.4) and other basic information, which is used to build the TC packet and data field headers.

TC parameters, which make up the actual command data, are defined in the CPC table (Command Parameter Characteristics). They are identified by a name, comprise a description, a format (see Sect. 16.7), and possibly a default value and a decalibration (see Sect. 16.2).

The CDF table (Command Details File) is used to define which TC parameters appear in which TC packet at which position. The table contains the name of the command, the name of the TC parameter, its location within the TC and possibly an associated value.

The TC parameter decalibrations are defined in two tables for each type of decalibration. For the numerical decalibration these are the CCA (Parameter Calibration Curve File) and CCS (Calibration Curve Set File) files, for textual decalibration, which are also called "alias definitions", these are the PAF (Parameter Alias File) and PAS (Parameter Alias Set) tables.

Telecommand reception and execution on board can be verified by acknowledge TM or by TM parameter values. For the FLP, only the acknowledge packets are used for this purpose. The CVS table (Verification Stages File) contains the definitions of different verification stages according to the different verification stages

of the acknowledge packets defined in [35]. The CVP table (Command Verification Profiles) contains entries for each verification stage of each TC, which is to be checked.

Sequences of TCs are also defined as part of the MIB. FLP Flight Procedures which are to be executed without ground station contact are defined in MOIS and are imported to the MIB as sequences (see also Chap. 15). The general characteristics of the sequences like their name, description, number of elements are defined in the CSF table (Command Sequence File), the single TCs or sequences which are part of the sequence are defined in the CSS table (Command Sequence Set) and the TC parameters within these TCs are defined in the SDF table (Sequence Details File). Formal parameters of the sequences are defined in the CSP table (Command Sequence Parameter File).

Furthermore, there are tables defining the detailed structure of the TC headers, namely the TCP, PCPC and PCDF tables. Within these tables the headers defined in Sect. 2.2 are defined once and used for each TC.

16.4 TM Packet Definitions and Identification

There exist two kinds of TM packets, namely fixed and variable packets. Fixed packets are uniquely identified on ground by the "SCOS-2000 Packet ID" (SPID). Variable packets are identified by the SPID and a "TM Packet Structure Definition" (TPSD). Both IDs are defined to be identical for a given packet for the FLP.

TM packet names have a 12 character format.

Packet Naming by SPID

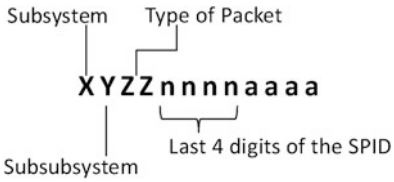
The SCOS-2000 Packet ID (SPID) is used as the main identifier for TM packets. It is a 4–5 digit decimal number, in which

Table 16.3 SPID first digit versus Packet Type

Packet Type	SPID 1st digit
Srv. 6 (Memory), Srv. 15 (TM Storage), Srv. 17 (Ping), Srv. 200-202 (Mode/Health)	None
Srv. 1 (Acknowledge)	1
Srv. 2 (Device Commanding)	2
Srv. 3 (Housekeeping)	3
Srv. 12 (Monitoring)	4
Srv. 5 (Event Reporting)	5
Srv. 5 (Event Reporting)	6
	7
Srv 8. (Function), Srv 9 (Time)	8
	9

- the 1st digit (or the lack of it) indicates the type of packet—see Table 16.3,
- the following 2 digits indicate the subsystem and subsubsystem, as defined in the "ID Range" column of Table 16.1
  - exception: Event TM packets may have any digits here.

The following 2 digits are used as a counter.



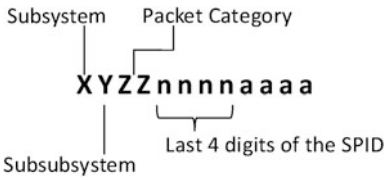
Examples:

- 4935: Srv (6,136) PSS Memory Dump
- 12603: Srv (1,3) Execution Started Success

Note that the ID range (SPID range) given in Table 16.1 in decimal values is identical to the 1st byte of the any parameter ID in hexadecimal values in the same subsystem/assembly/device. This is due to the fact that the parameter ID will be interpreted in hexadecimal by the OBSW (see Sect. 16.6) and that the SPID will mostly be used on ground as a decimal value.With a given SPID, the parameters contained in a TM packet can be extracted and displayed by SCOS-2000.

Packet Naming by PCAT

SCOS TM packets also have a name with the format according to the packet category:



The definitions by packet category following the convention in Table 16.4 is in principle a further detailing of the specifications in Table 16.3.

Example:

- PPHK4201swi2
  (PCDU switch status 2)
  PCDU HK TM Srv. (3,25), SPID 34201

The SPID—which is SCOS specific—is however not included explicitly in the TM packet which is transmitted from spacecraft to ground. So the exact type of a received packet must be identified on ground by other means.

Table 16.4 TM Packet Category definitions

Packet Category	Code (ZZ)	Srv.	Subsrv.
Acknowledge Success	AK	1	1, 3, 5, 7
Acknowledge Failure	AF	1	2, 4, 6,~8
Direct TM	DT	2	135
Raw TM	RD	2	128,134
Housekeeping	HK	3	25
Diagnostics	DG	3	26
Event	EV	5	all
Memory Dump	MD	6	6,136
Memory Check	MC	6	10
Function, Mode/Health	FC	8, 200–202	all
Time	TI	9	all
Monitoring Report	MO	12	12
TM Storage Information	SI	15	6
Test Service (Ping)	TE	17	2

The following IDs are 209 used, which are part of the packet headers or data fields:

- APID in the TM Source Packet Header
- Service Type and Subtype in the Data Field Header
- 0–2 additional *packet identification fields (PI1, PI2)* in the TM Source Data Field

If the APID and the service type and subtype are not sufficient for identification, the additional identification fields must be used. Their presence and locations are defined in the PIC table for a given service type and subtype. It is therefore not possible to identify packets of the same subservice by different means. The additional identification fields are used for all services, for which many different packets of the same type exist, most importantly for housekeeping and event TM.

Service (3,25) housekeeping TM packets as defined in [34] for example, comprise a so-called Structure ID (SID) identifying the packet on ground together with service type, subtype and APID (see also Fig. 2.7). From these values contained in an incoming TM packet, the SPID of the packet can be derived by SCOS-2000. The *PI1* field is used for the SID value; the *PI2* value is not used for service 3 packets. The SID consists of the last four digits (unsigned integer in decimal) of the SPID (see above), which are also part of the packet name. In the example above, the SID of the packet named PPHK4201swi2 is 4201.

Similar to the SIDs, event reports are identified by Report IDs (RID) as defined in [34] as the *PI1* field of the TM packet. Again, the RID is equal to the last four digits of the packet's SPID. Note that for the FLP, instead of RID the term "Event ID" is preferred.

Note however that due to the OBSW Event reporting mechanism, the RID is not sufficient to identify an Event TM packet. It is possible that the same Event is generated by different on-board objects, because they share a similar functionality. For example mode change events can be generated by a multiplicity of objects. Therefore, the object-ID of the object generating the event is used as the *PI2* field of the event packets to distinguish between events with the same RID but from different on-board objects.

16.5 TM Parameter and Displays Tables

The TM parameters are defined in the MIB in the PCF table (Parameter Characteristics File). This table contains the name, description, ID, format, calibration and further information on the parameter.

For downlink to the ground, TM parameters are integrated into TM packets. It is possible that a dedicated parameter appears in more than one TM packet and also that within one packet several instances of the same parameter sampled at different times can be present. TM packets are defined with their name, ID and size in the TPCF table (Telemetry Packet Characteristics File). In the PID table (Packet Identification File, not to be confused with the parameter ID also abbreviated PID) their type, subtype, description and additional identification fields are defined.

The location of the parameters within fixed packets are defined in the PLF table (Parameter Location File). The parameter locations within variable packets are given in the VPD (Variable Packet Definition File). The location and size of additional identification fields present in different kinds of TM packets identified by their type/subtype are defined in the PIC table (Packet Identification Criteria File).

Parameters are automatically checked by SCOS-2000 if they are within a predefined range, otherwise warnings are issued. These limits and their associated checks are defined in the OCP (Out Of Limits Definition File) and OCF (Out Of Limits Checks File) respectively.

The parameters are displayed on one of the three already mentioned kinds of displays, i.e. alphanumerical displays (AND) showing the newest available values of up to 64 parameters as numbers or text, graphical displays (GRD) showing one or more graphs of the values of certain parameters as a function of time and scrolling displays (SCD) showing a single row of up to five parameters scrolling down with time. For each of these display types, there is one table defining the general attributes of the display (name, type, number of graphs, etc.) and one table defining the parameters to be displayed and their locations. For the AND, these tables are DPF (Alphanumeric Display Proforma File) and DPC (Alphanumeric Display Proforma Definition File), for the GRD, these tables are GPF (Graphic Display Proforma File) and GPC (Graphic Display Proforma Definition File) and for the SCD, these tables are SPF (Scrolling Display Proforma File) and SPC (Scrolling Display Proforma Definition File).

Telemetry parameters can also be calibrated textually or numerically. The general attributes of textual calibrations are defined in the TXF table (Text Strings Calibration Curve File) and the actual calibration mapping is given in the TXP table (Text Strings Calibration Curve Definition File). Similarly for numerical calibration, the tables CAF (Calibration Curve File) and CAP (Calibration Curve Definition File) are used. It is also possible to use a polynomial to calibrate a TM parameter, defined in the MCF table (Polynomial Calibration Curve Definitions).

16.6 TM Parameter Identification

In the MIB, there are different kinds of TM parameters defined in the PCF table. The most important ones are the parameters which are referred to as *datapool* variables in terms of the OBSW, as defined in Sect. 3.2.2. These parameters are periodically written into the on-board *datapool* if the handler of the associated object is in Normal mode (see Sect. 3.2.2). In the FLP MIB, they are marked with an additional field in the PCF table. These parameters are defined in [34, 35] to be the "lowest level of elementary data item on board". They are uniquely defined by a *parameter number (#)* for on-board identification. This parameter number is defined in the PCF table of the MIB as the parameter ID (PID). Examples for this kind of parameters include housekeeping TM and sensor values, as attitude quaternions, temperatures, etc.

Less important TM parameters defined in the MIB are those device TM parameters which are not frequently updated in the *datapool* and are not used on board. These parameters are not needed by any on-board process and cannot be downlinked by Service 3. Instead, they can be downlinked via Service 2 using raw or direct commanding to the devices, to assemblies and controllers (see Sect. 2.4.2) or via Services 6 (see Sect. 2.4.6) and via Service 8 (see Sect. 2.4.7). Examples for this kind of parameters are the PCDU operating CPU ID, version information of software used. Also, raw TM parameters downlinked fall in this category.

Finally, there are TM parameters in the MIB which are necessary to define TM packets of all sorts, as for example acknowledge TM error codes, length parameters, event packets parameters etc. These parameters are not used on board, but are necessary to keep in the MIB for packet parsing and display purposes.

- As the parameter ID is also used in the *datapool* of the OBSW, it is defined bytewise and not as a decimal number. The interpretation is necessary, because in the OBSW *datapool* all parameters are defined as arrays with an identifier, the so-called "Datapool Access ID", pointing to the start of the array. However, parameters can only be defined as scalar values, not as arrays, in the MIB. Note that these arrays in the OBSW can have the length 1, thus allowing scalar parameters. Therefore, the parameter ID consists of the Datapool Access ID and an index to identify the position within an array.

- The parameter#/ID consists of 4 bytes:
  - Table 16.1 shows the first byte of the parameter ID for the different subsystems and devices in hexadecimal values.
  - The second byte can be used in each subsystem to define which of the single instances the parameter belongs to, if more than one is present e.g. for redundancy reasons, or to distinguish between raw and engineering parameters.
  - The third byte is used as an actual counter to distinguish different arrays.
  - The fourth byte gives the index of the array starting from 0x00.

The first three bytes are combined to the Datapool Access ID.

Note that TM parameters (as well as TM packets, displays etc.) are not always associated to the devices they were acquired in by their name (see Sect. 16.2). For example in the PCDU parameters of the battery, the solar panels, the SuS and some temperature sensors are measured. However, these parameters—in their calibrated version as present in the *datapool*—are associated by their name and ID to the subsystems by which they are used primarily, i.e. the battery system, the solar panel system, the SuS system and the TCS respectively.

Raw parameters however, which are part of raw packets and which are directly downlinked from a device and which are not calibrated before downlink, are always associated to the device they were acquired in, as they are part of raw TM packets originating from the device.

TM parameters are identified on ground by both their parameter ID and the parameter name which follows the format described in Sect. 16.2. The five last characters "**aaaaa**" of the name are divided into two blocks "**nnncc**", where

- **nnn** is a unique name for each parameter or array of parameters and
- **cc** is a counter/index for vector variables or if more than one of the same type exist

For example, *ARTIMO00* would be a TM parameter of the RW with a parameter ID starting with 0x14 (see Table 16.1); *ARTIMO01* would be the same parameter for RW1.

It needs to be noted that parameters of data type *string* are handled differently from others. In the MIB, TM parameters as for example TLE data, is defined as a string with PTC 8 and PFC giving the number of characters the string contains (see Sect. 16.7).

However, in the OBSW, strings are handled as arrays of characters, i.e. for each string with PFC > 1, there is an array of *datapool* variables with consecutive PIDs, only the first of which is defined in the MIB as a string. This has two implications:

- First, there can be no array of strings in the MIB, i.e. there are never two TM parameters with PTC 8 with PIDs being identical in the first three bytes.

- Secondly, in order to downlink these strings using Service 3, the dedicated OBSW parameters have to be defined in the MIB in the Service (3,1) TC. This way, SCOS interprets the relevant fields of the TM packet as a string value and displays it as such.

16.7 Parameter Format Definition

The format of any parameter is defined in SCOS-2000 and thus in the MIB using two fields named

- Parameter Type Code (PTC) giving the general type (unsigned int, floating point, etc.) and
- Parameter Format Code (PFC) giving the length or precision of the parameter.

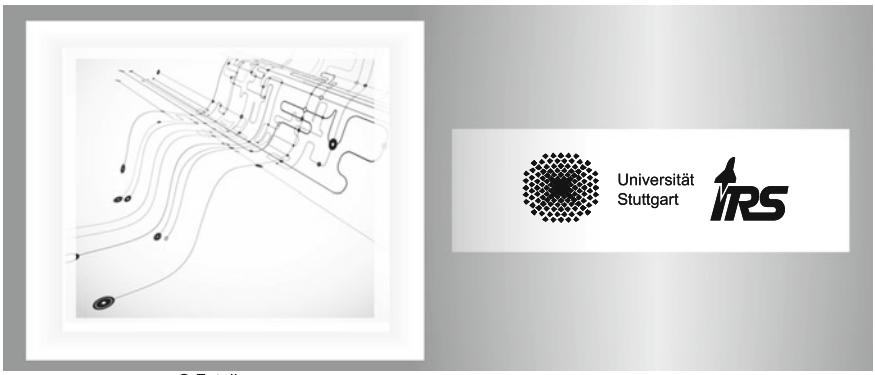
The exact definition of these is given in [86], some of the most common PTC/PFC combinations are provided in Table 16.5. Note that there are no data formats for vectors, arrays or matrices for numerical parameters. These need to be defined implicitly by other means than the PTC/PFC (see Sect. 16.6).

Table 16.5 Most commonly used SCOS-2000 format codes

PTC	PFC	Common name
1	0	Boolean (only valid flags)
2	Any	Unsigned integer, PFC gives number of bits
3	4	Unsigned integer, length 1 byte
3	12	Unsigned integer, length 2 bytes
3	14	Unsigned integer, length 4 bytes
4	4	Signed integer, length 1 byte
4	12	Signed integer, length 2 bytes
4	14	Signed integer, length 4 bytes
5	1	Float, length 4 bytes
5	2	Double, length 8 bytes
7	Any	Octet String, PFC gives number of bytes
8	Any	String, PFC gives the number of characters

Chapter 17
Annexes and Data Sheets

Jens Eickhoff



Abstract The chapter comprises tables, listings and figures complementing the information in the previous chapters. The topics cover the areas cited in the keyword list.

Keywords Software constants and FDIR limits · Polling sequence table · Telecommand/telemetry/event packet definitions · Flight Procedures · TTC data sheets · Link budgets · Power subsystem data sheets · Mass budget · Example orbit analysis · Skin connector pinout · Red/green tagged item list · Two Line Element definitions

J. Eickhoff (✉)
Airbus DS GmbH, Friedrichshafen, Germany
e-mail: jens.eickhoff@airbus.com

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5_17

17.1 Software Constants and FDIR Limits

See Table 17.1.

Table 17.1 Constants and FDIR limits

Description	Value
Interval for State Vector Storage	300 s
Interval for State Vector Transmission to secondary I/O-Board	900 s
Time after which the PCDU begins OBC reconfiguration	10 s
Time after which the PCDU performs the next reconfiguration step	30 s
Counter Limit for missing or corrupted replies of the PCDU, after whose exceeding FDIR prepares for the OBC reconfiguration sequence	10
Software watchdog reset timer	5 s
Interval for SV storage in primary I/O-Board	300 s
Interval for SV transmission to secondary I/O-Board	900 s
Waiting time to see whether I/O-Board switch worked and SpW link is up	5 s
PCDU initiates Shutdown mode	20 V
Device Level	
Upper limit for the failure counter of a device	FAILURE_LIMIT = 4600
Increment of the failure counter if a device is identfied as faulty in one cycle	FAILURE_INCREMENT = 1000
Decrement of the failure counter if a device is not faulty in one cycle	FAILURE_DECREMENT = 1
Upper limit for the boot counter for a device	BOOT_LIMIT = 9009
Increment of the boot counter for a device if a device was rebooted	BOOT_INCREMENT = 9000
Decrement for boot counter if a device did not need rebooting in the last cycle	BOOT_DECREMENT = 1
Thermal Control	
Temperature gradient between two measurements	10 K/s (Only applicable if the computation is executed with 1 … 5 Hz)
Upper limit for T sensor sanity	+150 °C
Lower limit for T sensor sanity	−50 °C
TCS control limits are given in [129]	
Maximum heating time	3600 s
Maximum heating time in Safe mode	1800 s

17.2 Polling Sequence Table for I/O-Board Access

Handler	Time [ms]	Action
RW_HANDLER_0	0	Send RMAP Read Command to IO Board
RW_HANDLER_1	0	Send RMAP Read Command to IO Board
RW_HANDLER_2	0	Send RMAP Read Command to IO Board
RW_HANDLER_3	0	Send RMAP Read Command to IO Board
MGT_HANDLER_N	0	Send RMAP Read Command to IO Board
MGT_HANDLER_R	0	Send RMAP Read Command to IO Board
DDS_HANDLER_N	0	Send RMAP Read Command to IO Board
DDS_HANDLER_R	0	Send RMAP Read Command to IO Board
PLOC_VTX_ROUTER_N	0	Send RMAP Read Command to IO Board
RW_HANDLER_0	6	Get Read Reply from RMAP Buffer
RW_HANDLER_1	6	Get Read Reply from RMAP Buffer
RW_HANDLER_2	6	Get Read Reply from RMAP Buffer
RW_HANDLER_3	6	Get Read Reply from RMAP Buffer
FOG_HANDLER	6	Get Read Reply from RMAP Buffer
MGT_HANDLER_N	6	Get Read Reply from RMAP Buffer
MGT_HANDLER_R	6	Get Read Reply from RMAP Buffer
DDS_HANDLER_N	6	Get Read Reply from RMAP Buffer
DDS_HANDLER_R	6	Get Read Reply from RMAP Buffer
PLOC_VTX_ROUTER_N	6	Get Read Reply from RMAP Buffer
IO_ASSEMBLY	6	Reset RMAP Channel
FDIR_CONTROLLER	6	Send RMAP Write Command to IO Board
RW_HANDLER_0	6	Send RMAP Write Command to IO Board
RW_HANDLER_1	6	Send RMAP Write Command to IO Board
RW_HANDLER_2	6	Send RMAP Write Command to IO Board
RW_HANDLER_3	6	Send RMAP Write Command to IO Board
MGM_HANDLER_N	6	Send RMAP Write Command to IO Board
MGM_HANDLER_R	6	Send RMAP Write Command to IO Board
PCDU_HANDLER	6	Send RMAP Write Command to IO Board
FOG_HANDLER	6	Send RMAP Write Command to IO Board
MGT_HANDLER_N	6	Send RMAP Write Command to IO Board
MGT_HANDLER_R	6	Send RMAP Write Command to IO Board
DDS_HANDLER_N	6	Send RMAP Write Command to IO Board
DDS_HANDLER_R	6	Send RMAP Write Command to IO Board
PLOC_VTX_ROUTER_N	6	Send RMAP Write Command to IO Board
STR_HANDLER	6	Send RMAP Write Command to IO Board
GPS_HANDLER_0	6	Send RMAP Write Command to IO Board
GPS_HANDLER_1	6	Send RMAP Write Command to IO Board
GPS_HANDLER_2	6	Send RMAP Write Command to IO Board
IO_ASSEMBLY	6	Send RMAP Write Command to IO Board

(continued)

(continued)

Handler	Time [ms]	Action
PLOC_POWER_HANDLER_N	6	Send RMAP Write Command to IO Board
PLOC_POWER_HANDLER_R	6	Send RMAP Write Command to IO Board
PLOC_CC_HANDLER_N	6	Send RMAP Write Command to IO Board
PLOC_CC_HANDLER_R	6	Send RMAP Write Command to IO Board
DDS_SWITCH	6	Send RMAP Write Command to IO Board
TTC_RX_HANDLER_N	6	Send RMAP Write Command to IO Board
TTC_RX_HANDLER_R	6	Send RMAP Write Command to IO Board
TTC_TX_HANDLER_N	6	Send RMAP Write Command to IO Board
TTC_TX_HANDLER_R	6	Send RMAP Write Command to IO Board
GPS_CDM_HANDLER	6	Send RMAP Write Command to IO Board
RW_HANDLER_0	115	Get Write Reply from RMAP Buffer
RW_HANDLER_1	115	Get Write Reply from RMAP Buffer
RW_HANDLER_2	115	Get Write Reply from RMAP Buffer
RW_HANDLER_3	115	Get Write Reply from RMAP Buffer
RW_HANDLER_0	115	Send RMAP Read Command to IO Board
RW_HANDLER_1	115	Send RMAP Read Command to IO Board
RW_HANDLER_2	115	Send RMAP Read Command to IO Board
RW_HANDLER_3	115	Send RMAP Read Command to IO Board
FOG_HANDLER	115	Get Write Reply from RMAP Buffer
MGT_HANDLER_N	115	Get Write Reply from RMAP Buffer
MGT_HANDLER_R	115	Get Write Reply from RMAP Buffer
DDS_HANDLER_N	115	Get Write Reply from RMAP Buffer
DDS_HANDLER_R	115	Get Write Reply from RMAP Buffer
PLOC_VTX_ROUTER_N	115	Get Write Reply from RMAP Buffer
FOG_HANDLER	115	Send RMAP Read Command to IO Board
MGT_HANDLER_N	115	Send RMAP Read Command to IO Board
MGT_HANDLER_R	115	Send RMAP Read Command to IO Board
DDS_HANDLER_N	115	Send RMAP Read Command to IO Board
DDS_HANDLER_R	115	Send RMAP Read Command to IO Board
PLOC_VTX_ROUTER_N	115	Send RMAP Read Command to IO Board
RW_HANDLER_0	120	Get Read Reply from RMAP Buffer
RW_HANDLER_1	120	Get Read Reply from RMAP Buffer
RW_HANDLER_2	120	Get Read Reply from RMAP Buffer
RW_HANDLER_3	120	Get Read Reply from RMAP Buffer
RW_HANDLER_0	120	Send RMAP Write Command to IO Board
RW_HANDLER_1	120	Send RMAP Write Command to IO Board
RW_HANDLER_2	120	Send RMAP Write Command to IO Board
RW_HANDLER_3	120	Send RMAP Write Command to IO Board
MGM_HANDLER_N	120	Get Write Reply from RMAP Buffer
MGM_HANDLER_R	120	Get Write Reply from RMAP Buffer

(continued)

(continued)

Handler	Time [ms]	Action
STR_HANDLER	120	Get Write Reply from RMAP Buffer
GPS_HANDLER_0	120	Get Write Reply from RMAP Buffer
GPS_HANDLER_1	120	Get Write Reply from RMAP Buffer
GPS_HANDLER_2	120	Get Write Reply from RMAP Buffer
IO_ASSEMBLY	120	Get Write Reply from RMAP Buffer
PLOC_POWER_HANDLER_N	120	Get Write Reply from RMAP Buffer
PLOC_POWER_HANDLER_R	120	Get Write Reply from RMAP Buffer
PLOC_CC_HANDLER_N	120	Get Write Reply from RMAP Buffer
PLOC_CC_HANDLER_R	120	Get Write Reply from RMAP Buffer
TTC_RX_HANDLER_N	120	Get Write Reply from RMAP Buffer
TTC_RX_HANDLER_R	120	Get Write Reply from RMAP Buffer
GPS_CDM_HANDLER	120	Get Write Reply from RMAP Buffer
STR_HANDLER	120	Send RMAP Read Command to IO Board
GPS_HANDLER_0	120	Send RMAP Read Command to IO Board
GPS_HANDLER_1	120	Send RMAP Read Command to IO Board
GPS_HANDLER_2	120	Send RMAP Read Command to IO Board
IO_ASSEMBLY	120	Send RMAP Read Command to IO Board
PLOC_POWER_HANDLER_N	120	Send RMAP Read Command to IO Board
PLOC_POWER_HANDLER_R	120	Send RMAP Read Command to IO Board
PLOC_CC_HANDLER_N	120	Send RMAP Read Command to IO Board
PLOC_CC_HANDLER_R	120	Send RMAP Read Command to IO Board
TTC_RX_HANDLER_N	120	Send RMAP Read Command to IO Board
TTC_RX_HANDLER_R	120	Send RMAP Read Command to IO Board
GPS_CDM_HANDLER	120	Send RMAP Read Command to IO Board
FOG_HANDLER	120	Get Read Reply from RMAP Buffer
MGT_HANDLER_N	120	Get Read Reply from RMAP Buffer
MGT_HANDLER_R	120	Get Read Reply from RMAP Buffer
DDS_HANDLER_N	120	Get Read Reply from RMAP Buffer
DDS_HANDLER_R	120	Get Read Reply from RMAP Buffer
PLOC_VTX_ROUTER_N	120	Get Read Reply from RMAP Buffer
FOG_HANDLER	120	Send RMAP Write Command to IO Board
MGT_HANDLER_N	120	Send RMAP Write Command to IO Board
MGT_HANDLER_R	120	Send RMAP Write Command to IO Board
DDS_HANDLER_N	120	Send RMAP Write Command to IO Board
DDS_HANDLER_R	120	Send RMAP Write Command to IO Board
PLOC_VTX_ROUTER_N	120	Send RMAP Write Command to IO Board
PCDU_HANDLER	150	Get Write Reply from RMAP Buffer
PCDU_HANDLER	150	Send RMAP Read Command to IO Board
STR_HANDLER	150	Get Read Reply from RMAP Buffer
GPS_HANDLER_0	150	Get Read Reply from RMAP Buffer

(continued)

(continued)

Handler	Time [ms]	Action
GPS_HANDLER_1	150	Get Read Reply from RMAP Buffer
GPS_HANDLER_2	150	Get Read Reply from RMAP Buffer
IO_ASSEMBLY	150	Get Read Reply from RMAP Buffer
PLOC_POWER_HANDLER_N	150	Get Read Reply from RMAP Buffer
PLOC_POWER_HANDLER_R	150	Get Read Reply from RMAP Buffer
PLOC_CC_HANDLER_N	150	Get Read Reply from RMAP Buffer
PLOC_CC_HANDLER_R	150	Get Read Reply from RMAP Buffer
TTC_RX_HANDLER_N	150	Get Read Reply from RMAP Buffer
TTC_RX_HANDLER_R	150	Get Read Reply from RMAP Buffer
GPS_CDM_HANDLER	150	Get Read Reply from RMAP Buffer
MGM_HANDLER_N	150	Send RMAP Read Command to IO Board
MGM_HANDLER_R	150	Send RMAP Read Command to IO Board
RW_HANDLER_0	150	Get Write Reply from RMAP Buffer
RW_HANDLER_1	150	Get Write Reply from RMAP Buffer
RW_HANDLER_2	150	Get Write Reply from RMAP Buffer
RW_HANDLER_3	150	Get Write Reply from RMAP Buffer
FOG_HANDLER	150	Get Write Reply from RMAP Buffer
MGT_HANDLER_N	150	Get Write Reply from RMAP Buffer
MGT_HANDLER_R	150	Get Write Reply from RMAP Buffer
DDS_HANDLER_N	150	Get Write Reply from RMAP Buffer
DDS_HANDLER_R	150	Get Write Reply from RMAP Buffer
PLOC_VTX_ROUTER_N	150	Get Write Reply from RMAP Buffer
PCDU_HANDLER	170	Get Read Reply from RMAP Buffer
FOG_HANDLER	170	Send RMAP Read Command to IO Board
MGM_HANDLER_N	170	Get Read Reply from RMAP Buffer
MGM_HANDLER_R	170	Get Read Reply from RMAP Buffer

17.3 Spacecraft Telecommand/Telemetry Definitions

This section comprises the tables for spacecraft telecommand packets, telemetry packets and spacecraft Event packets. For parameters calibrations etc. refer to the FLP MIB.

17.3.1 Telecommand Definitions

See Table 17.2.

Table 17.2 S/C telecommand packet definitions

System TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
YYC00000	PUS Service 17 Ping Test	same TC as "PING"	17	1	53	0	C
YYC10000	COP Ctrl Frame: Unlock	COP-1 Control Frame (Type-BC): Unlock FARM	255	255	53	0	F
YYC10001	COP Ctrl Frame: Set V (R)	COP-1 Control Frame (Type-BC): Set V(R) (Receiver Frame Seq Nr)	255	255	53	0	F
System Manager TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
YMC20001	Srv(200,1) Set Mode Devi	Set Mode of any device object identified by its object-ID	200	1	53	11	C
YMC20002	Srv(200,2) Read Mode Dev	Read Mode of a single Device Object	200	2	53	11	C
YMC20003	Srv(200,3) ReadModRecDev	Read Mode of a device object and ist children recursively	200	3	53	11	C
YMC20101	Srv(201,1) SetHealFlagDev	Set Health Flag of a device object	201	1	53	11	C
YMC20104	Srv(201,4) ReadHealDevic	Read the health flag of a single device object (TM via Srv. 5)	201	4	53	11	C
YMC21001	Srv(200,1) Set Mode PL	Set Mode of any payload object identified by ist object-ID	200	1	53	11	C
YMC21002	Srv(200,2) Read Mode PL	Read Mode of a single payload Object	200	2	53	11	C
YMC21003	Srv(200,3) ReadModRecPL	Read Mode of a payload object and ist children recursively	200	3	53	11	C
YMC21101	Srv(201,1) SetHealFlagPL	Set Health Flag of a payload object	201	1	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

YMC21104	Srv(201,4) ReadHealPL	Read the health flag of a single payload object (TM via Srv. 5)	201	4	53	11	C
YMC22001	Srv(200,1) Set Mode Sys	Set Mode of the system object identified by its object-ID	200	1	53	11	C
YMC22002	Srv(200,2) Read Mode Sys	Read Mode of the system Object	200	2	53	11	C
YMC22003	Srv(200,3) ReadModRecSys	Read Mode of the system object and ist children recursively	200	3	53	11	C
YMC22101	Srv(201,1) SetHealFlagSys	Set Health Flag of the system object	201	1	53	11	C
YMC22102	Srv(201,2) ReadAllHealSys	Read Health flags of all objects	201	2	53	11	C
YMC22104	Srv(201,4) ReadHealSys	Read the health flag of the system object (TM via Srv. 5)	201	4	53	11	C
YMC22201	Srv(202,1) AddModTableSys	Add Mode Change Table of the system	202	1	53	11	C
YMC22202	Srv(202,2) AddModeSeqSys	Add Mode Change Sequence of the system	202	2	53	11	C
YMC22203	Srv(202,3) DelModeTabSys	Delete Mode Change Table of the system	202	3	53	11	C
YMC22204	Srv(202,4) DelModeSeqSys	Delete Mode Change Sequence of the system	202	4	53	11	C
YMC22205	Srv(202,5) ListModTablSys	Request available mode Change Tables of the system	202	5	53	11	C
YMC22207	Srv(202,7) ListModeSeqSys	Request available Mode Change Sequences of the system	202	7	53	11	C
YMC22209	Srv(202,9) ReadModTablSys	Read Mode Change Table Data of the system	202	9	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

YMC22211	Srv(202,11) ReadModSeqSys	Read Mode Change Sequence Data of the system	202	11	53	11	C
YMC22213	Srv(202,13) RqMemTbSqSys	Request Available Memory Slots for Mode Change Tables and Sequence	202	13	53	11	C
YMC22215	Srv(202,15) RqObjMemTbSys	Request available memory slots for mode change tables of 1 system	202	15	53	11	C
YMC22217	Srv(202,17) RqObjMemSqSys	Request available memory slots for mode change sequence of 1 system	202	17	53	11	C
YMC23001	Srv(200,1) Set Mode Ass	Set Mode of any assembly/controller object	200	1	53	11	C
YMC23002	Srv(200,2) Read Mode Ass	Read Mode of a single assembly/controller Object	200	2	53	11	C
YMC23003	Srv(200,3) ReadModRecAss	Read Mode of an assembly/ctrl object and its children recursively	200	3	53	11	C
YMC23101	Srv(201,1) SetHealFlagAss	Set Health Flag of an assembly/controller object	201	1	53	11	C
YMC23104	Srv(201,4) ReadHealAss	Read the health flag of a assembly/ctrl object (TM via Srv. 5)	201	4	53	11	C
YMC24001	Srv(200,1) Set Mode Sub	Set Mode of a subsystem object identified by its object-ID	200	1	53	11	C
YMC24002	Srv(200,2) Read Mode Sub	Read Mode of a subsystem Object	200	2	53	11	C
YMC24003	Srv(200,3) ReadModRecSub	Read Mode of a subsystem object and ist children recursively	200	3	53	11	C
YMC24101	Srv(201,1) SetHealFlagSub	Set Health Flag of a subsystem object	201	1	53	11	C
YMC24104	Srv(201,4) ReadHealSub	Read health flag of a subsystem object (TM via Srv. 5)	201	4	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

YMC24201	Srv(202,1) AddModTableSub	Add Mode Change Table of a subsystem	202	1	53	11	C
YMC24202	Srv(202,2) AddModeSeqSub	Add Mode Change Sequence of a subsystem	202	2	53	11	C
YMC24203	Srv(202,3) DelModeTabSub	Delete Mode Change Table of a subsystem	202	3	53	11	C
YMC24204	Srv(202,4) DelModeSeqSub	Delete Mode Change Sequence of a subsystem	202	4	53	11	C
YMC24205	Srv(202,5) ListModTablSub	Request available mode Change Tables of a subsystem	202	5	53	11	C
YMC24207	Srv(202,7) ListModeSeqSub	Request available Mode Change Sequences of a subsystem	202	7	53	11	C
YMC24209	Srv(202,9) ReadModTablSub	Read Mode Change Table Data of a subsystem	202	9	53	11	C
YMC24211	Srv(202,11) ReadModSeqSub	Read Mode Change Sequence Data of a subsystem	202	11	53	11	C
YMC24215	Srv(202,15) RqObjMemTbSub	Request available memory slots for mode change tables of 1 system	202	15	53	11	C
YMC24217	Srv(202,17) RqObjMemSqSub	Request available memory slots for mode change sequence of 1 system	202	17	53	11	C
YMC24301	Srv(200,1) Set Mode ACS	Set Mode of the ACS Subsystem	200	1	53	11	C
YMC24302	Srv(200,1) Set Mode CDH	Set Mode of the CDH Subsystem	200	1	53	11	C
YMC24303	Srv(200,1) Set Mode PL	Set Mode of the PL Subsystem	200	1	53	11	C
YMC24304	Srv(200,1) Set Mode PSS	Set Mode of the PSS Subsystem	200	1	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

YMC24305	Srv(200,1) Set Mode TCS	Set Mode of the TCS Subsystem	200	1	53	11	C
YMC24306	Srv(200,1) Set Mode TTC	Set Mode of the TTC Subsystem	200	1	53	11	C
YMC25101	Srv(201,1) SetHealFlagHel	Set Health Flag of a health object	201	1	53	11	C
YMC25104	Srv(201,4) ReadHealHel	Read health flag of a health object (TM via Srv. 5)	201	4	53	11	C
Time Manager TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
YTC00010	Srv(9,128) Set CCS Time	Request to change the system time in CCS format (CCSDS)	9	128	53	9	C
Telecommand Scheduler TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
YSC00001	Srv(11,1) EnableTCrelease	Srv (11,1) Enable Release of TCs from On-Board TC Schedule	11	1	53	9	C
YSC00002	Srv(11,2) DisableTCreleas	Srv (11,2) Disable Release of TCs from On-Board TC Schedule	11	2	53	9	C
YSC00003	Srv(11,3) ResetTCschedule	Srv (11,3) Reset On-Board TC Schedule (Delete all entries)	11	3	53	9	C
YSC00004	Srv(11,4) LoadTimeTagged	Srv (11,4) Load Time Tagged TC to On-Board TC Schedule	11	4	53	9	C
Memory TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
DMC00000	Srv6 Load Memory AbsAd	Load Data to Memory at an absolute address	6	2	53	15	C
DMC00001	Srv6 Dump Memory AbsAd	Dump Data from Memory from an absolute address	6	5	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

DMC00002	Srv6 Check Memory AbsAd	Check Data in Memory at an absolute address	6	9	53	11	C
DMC00003	Srv6 Load Float to DP	Load a float value to the Data Pool	6	132	53	11	C
DMC00004	Srv6 Load TLE Row1 to DP	Srv6 Load TLE Row1 to Data Pool	6	132	53	11	C
DMC00005	Srv6 Load TLE Row2 to DP	Srv6 Load TLE Row2 to Data Pool	6	132	53	11	C
DMC10010	Srv(15,1) EnableTMStorage	Srv (15,1) Enable Storage in Packet Stores	15	1	53	11	C
DMC10020	Srv(15,2) DisableTMStorag	Srv (15,2) Disable Storage in Packet Stores	15	2	53	11	C
DMC10030	Srv(15,3) AddPktsStoreSel	Srv (15,3) Add Packets to Storage Selection Definition	15	3	53	11	C
DMC10040	Srv(15,4) DelPktsStoreSel	Srv (15,4) Delete Packets to Storage Selection Definition	15	4	53	11	C
DMC10050	Srv(15,5)ReportStoreSel	Srv (15,5) Report Storage Selection Definition	15	5	53	11	C
DMC10070	Srv(15,7)DownTMStore All	(15,7) Downlink Packet Store Contents for Packet range: All	15	7	53	11	C
DMC10071	Srv(15,7)DownTMStore Bet	(15,7) Downlink Packet Store Contents for Packet range: Between	15	7	53	11	C
DMC10072	Srv(15,7)DownTMStore Bef	(15,7) Downlink Packet Store Contents for Packet range: Before	15	7	53	11	C
DMC10073	Srv(15,7)DownTMStore Aft	(15,7) Downlink Packet Store Contents for Packet range: After	15	7	53	11	C
DMC10100	Srv(15,10)DelTMStore All	Srv (15,10) Delete TM Stores contents: All	15	10	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

DMC10102	Srv(15,10)DelTMStore Sel	Srv (15,10) Delete TM Stores contents: Select store, APID, SSC	15	10	53	11	C
DMC10120	Srv(15,12) ReportStorCata	Srv (15,12) Report Packet Store Catalogs (content overview)	15	12	53	11	C
OBC TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
DOC00001	Srv(8,1) SetPROMSleepStat	Srv (8,1) Set PROM Sleep State	8	1	53	11	C
DOC00002	Srv(8,1) SetPROMWriteStat	Srv (8,1) Set PROM Write State	8	1	53	11	C
DOC00003	Srv(8,1) Read PROM State	Srv (8,1) Read PROM State	8	1	53	11	C
DOC00010	Srv(8,1) Reset CPU Usage	Srv (8,1) Reset CPU Usage Statistics	8	1	53	11	C
DOC00011	Srv(8,1) Read CPU Usage	Srv (8,1) Read CPU Usage Statistics	8	1	53	11	C
DOC00020	Srv(8,1) Set Second Coun	Srv (8,1) Set Second Img. Boot Counter	8	1	53	11	C
DOC00021	Srv(8,1) Read Secon Coun	Srv (8,1) Read Second Img. Boot Counter	8	1	53	11	C
I/O-Board TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
DIC00000	Srv(2,2) Register Load	2	2	53	11	C	
OBSW TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
DSC00001	Srv(3,1) define new HK r	Srv(3,1) define new HK report	3	1	53	9	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

DSC00002	Srv(3,2) define new DG r	Srv(3,2) define new diagnostics report	3	2	53	9	C
DSC00003	Srv(3,3) ClearHK rep def	Srv(3,3) Clear Housekeeping report definitions	3	3	53	9	C
DSC00004	Srv(3,4) ClearDG rep def	Srv(3,4) Clear diagnostics report definition	3	4	53	9	C
DSC00005	Srv(3,5) Enable HKrepGen	Srv(3,5) Enable housekeeping report generation	3	5	53	9	C
DSC00006	Srv(3,6) DisableHKrepGen	Srv(3,6) Disable housekeeping report generation	3	6	53	9	C
DSC00007	Srv(3,7) Enable DGrepGen	Srv(3,7) Enable diagnostic report generation	3	7	53	9	C
DSC00008	Srv(3,8) DisableDGrepGen	Srv(3,8) Disable diagnostic report generation	3	8	53	9	C
DSC00009	Srv(3,9) Report HK Defs	Srv(3,9) Report housekeeping report definitions	3	9	53	9	C
DSC00011	Srv(3,11) Report DG Defs	Srv(3,11) Report diagnostic report definitions	3	11	53	9	C
DSC00050	Srv(5,5) EnableEventRep	Enable Event Report Generation	5	5	53	9	C
DSC00051	Srv(5,6) DisableEventRep	Disable Event Report Generation	5	6	53	9	C
DSC00055	Srv(5,128) Inject Event	Inject Event TC for test purposes	5	128	53	9	C
DSC00201	Srv(2,132) ChannelSwitch	Distribute Channel Switch Command	2	132	53	11	C
DSC00203	Srv(2,133) WiretappingOn	Enable Wiretapping, Downlink Device-DH communication via (2,129)	2	133	53	11	C
DSC00204	Srv(2,133) WiretappingOf	Disable Wiretapping,Downlink Device-DH communication via (2,129)	2	133	53	11	C
DSC01201	Srv(12,131) EnableMonito	Srv (12,131) Enable Monitoring of Parameters within one object	12	131	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

DSC01202	Srv(12,132) DisableMonit	Srv (12,132) Disable Monitoring of Parameters within one object	12	132	53	11	C
DSC01203	Srv(12,3) SetMaxRepDelay	Srv (12,3) Set maximum reporting delay	12	3	53	11	C
DSC01270	Srv(12,7) Modify MonitU8	Srv (12,7) Modify Monitor Information for uint8 parameters	12	7	53	11	C
DSC01271	Srv(12,7) ModifyMonitU16	Srv (12,7) Modify Monitor Information for uint16 parameters	12	7	53	11	C
DSC01272	Srv(12,7) ModifyMonitU32	Srv (12,7) Modify Monitor Information for uint32 parameters	12	7	53	11	C
DSC01273	Srv(12,7) Modify MonitI8	Srv (12,7) Modify Monitor Information for int8 parameters	12	7	53	11	C
DSC01274	Srv(12,7) ModifyMonitI16	Srv (12,7) Modify Monitor Information for int16 parameters	12	7	53	11	C
DSC01275	Srv(12,7) ModifyMonitI32	Srv (12,7) Modify Monitor Information for int32 parameters	12	7	53	11	C
DSC01276	Srv(12,7) ModifyMonitFLO	Srv (12,7) Modify Monitor Information for float parameters	12	7	53	11	C
DSC01277	Srv(12,7) ModifyMonitDOU	Srv (12,7) Modify Monitor Information for double parameters	12	7	53	11	C
DSC31000	Srv(3,1) DefineHK 31000	Srv(3,1) Define new HK report: 31000	3	1	53	9	C
Power Subsystem TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
PYC00100	(8,1) Set BattMaxCapacit	Set maximum capacity of battery string in the PSS Controller	8	1	53	9	C
PYC00200	(8,1) SetSoCestimatParaS0	Set SoC estimation parameters for String 0 in PSS Controller	8	1	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYC00201	(8,1) SetSoCestimatParaS1	Set SoC estimation parameters for String 1 in PSS Controller	8	1	53	11	C
PYC00202	(8,1) SetSoCestimatParaS2	Set SoC estimation parameters for String 2 in PSS Controller	8	1	53	11	C
PYC00300	(8,1) UpdatePCDUSoCparam	Update SoC estimation parameters in the PCDU from the PSS Ctrl	8	1	53	15	C
PYC12000	(12,7) PSSModifMonitInfo	Srv (12,7) Modify Monitoring Info for PSS Monitors	12	7	53	11	C
PYC12001	(12,137) SetMonitCompPwr	Srv (12,137) Modify Monitor Info for Component Power Monitor	12	137	53	11	C
PYC60000	Srv6 LoadBattSoCValues	Memory Load for PSS Battery SoC variables to PSS Ctrl	6	132	53	11	C
PYC60001	Srv6 LoadMaxDeltaCapacit	Memory Load Max Delta Capacity to PSS Ctrl	6	132	53	11	C
PYC60002	Srv6 LoadPCDUupdaInterv	Memory Load PCDU Update Interval to PSS Ctrl	6	132	53	11	C
PYC60003	Srv6 LoadPanDeployFlag	Memory Load Panel Deployment Flag to PSS Ctrl	6	132	53	11	C
PYC61000	Srv6 DumpBattlSoCVal	Memory Dump for PSS Battery SoC variables from PSS Ctrl	6	135	53	11	C
PYC61001	Srv6 DumpMaxDeltaCapacit	Memory Dump Max Delta Capacity from PSS Ctrl	6	135	53	11	C
PYC61002	Srv6 DumpPCDUupdaInterv	Memory Dump PCDU Update Interval from PSS Ctrl	6	135	53	11	C
PYC61003	Srv6 DumpPanDeployFlag	Memory Dump Panel Deployment Flag from PSS Ctrl	6	135	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPC01	HPC1 on OBCcoreN1 switch	HPC 1 OBCcoreN1 switch on	C				
PYCHPC02	HPC2 on OBCcoreN2 switch	HPC 2 OBCcoreR1 switch on	C				
PYCHPC03	HPC3 on OBCcoreR1 switch	HPC 3 OBCcoreR1 switch on	C				
PYCHPC04	HPC4 on OBCcoreR2 switch	HPC 4 OBCcoreR2 switch on	C				
PYCHPC05	HPC5 on IOboardN1 switch	HPC 5 IOboardN1 switch on	C				
PYCHPC06	HPC6 on IOboardN2 switch	HPC 6 IOboardN2 switch on	C				
PYCHPC07	HPC7 on IOboardR1 switch	HPC 7 IOboardR1 switch on	C				
PYCHPC08	HPC8 on IOboardR2 switch	HPC 8 IOboardR2 switch on	C				
PYCHPC09	HPC9 on CCSDS0 1 switch	HPC 9 CCSDS0 1 switch on	C				
PYCHPC0A	HPC10 on CCSDS0 2 switch	HPC 10 CCSDS0 2 switch on	C				
PYCHPC0B	HPC11 on CCSDS1 1 switch	HPC 11 CCSDS1 1 switch on	C				
PYCHPC0C	HPC12 on CCSDS1 2 switch	HPC 12 CCSDS1 2 switch on	C				
PYCHPC0D	HPC13 on TTC Rec 0 switc	HPC 13 TTC Rec 0 switch on	C				

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPC0E	HPC14 on TTC Rec 1 switc	HPC 14 TTC Rec 1 switch on	C				
PYCHPC0F		PL specific in first FLP mission	C				
PYCHPC10	HPC16 on PLOC Pow R1 swi	HPC 16 PLOC Pow R1 switch on	C				
PYCHPC11	HPC17 on PLOC Pow R2 swi	HPC 17 PLOC Pow R2 switch on	C				
PYCHPC12	HPC18 on DDS N switch	HPC 18 DDS N switch on	C				
PYCHPC13		PL specific in first FLP mission	C				
PYCHPC14	HPC20 on PLOC Pow N1 swi	HPC 20 PLOC Pow N1 switch on	C				
PYCHPC15	HPC21 on PLOC Pow N2 swi	HPC 21 PLOC Pow N2 switch on	C				
PYCHPC16	HPC22 on DDS R switch	HPC 22 DDS R switch on	C				
PYCHPC17		PL specific in first FLP mission	C				
PYCHPC18	HPC24 on TTCtransN1 swit	HPC 24 TTCtransN1 switch on	C				
PYCHPC19	HPC25 on TTCtransN2 swit	HPC 25 TTCtransN2 switch on	C				
PYCHPC1A	HPC26 on STR N switch	HPC 26 STR N switch on	C				
PYCHPC1B	HPC27 on RW 3 switch	HPC 27 RW 3 switch on	C				
PYCHPC1C	HPC28 on STR R switch	HPC 28 STR R switch on	C				
PYCHPC1D	HPC29 on FOG 3 switch	HPC 29 FOG 3 switch on	C				
PYCHPC1E	HPC30 on FOG 0 switch	HPC 30 FOG 0 switch on	C				

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPC1F	HPC31 on RW 0 switch	HPC 31 RW 0 switch on	C				
PYCHPC20	HPC32 on FOG 1 switch	HPC 32 FOG 1 switch on	C				
PYCHPC21	HPC33 on RW 1 switch	HPC 33 RW 1 switch on	C				
PYCHPC22	HPC34 on FOG 2 switch	HPC 34 FOG 2 switch on	C				
PYCHPC23	HPC35 on RW 2 switch	HPC 35 RW 2 switch on	C				
PYCHPC24	HPC36 on MGM 0 switch	HPC 36 MGM 0 switch on	C				
PYCHPC25	HPC37 on MGM 1 switch	HPC 37 MGM 1 switch on	C				
PYCHPC26		PL specific in first FLP mission	C				
PYCHPC27	HPC39 on MGT R switch	HPC 39 MGT R switch on	C				
PYCHPC28	HPC40 on GPS 1 switch	HPC 40 GPS 1 switch on	C				
PYCHPC29	HPC41 on GPS 0 switch	HPC 41 GPS 0 switch on	C				
PYCHPC2A	HPC42 on MGT N switch	HPC 42 MGT N switch on	C				
PYCHPC2B	HPC43 on GPS 2 switch	HPC 43 GPS 2 switch on	C				
PYCHPC2C		PL specific in first FLP mission	C				
PYCHPC2D			C				
PYCHPC2E	HPC46 on TTCtransR1 swit	HPC 46 TTCtransR1 switch on	C				
PYCHPC2F	HPC47 on TTCtransR2 swit	HPC 47 TTCtransR2 switc on	C				
PYCHPC30		PL specific in first FLP mission	C				
PYCHPC31			C				

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPC32	HPC50 on DDStransR1 swit	HPC 50 DDStransR1 switch on	C				
PYCHPC33	HPC51 on DDStransR2 swit	HPC 51 DDStransR2 switch on	C				
PYCHPC34	HPC52 on BimetHeaterR sw	HPC 52 BimetHeaterR switch on	C				
PYCHPC35	HPC53 on BimetHeaterN sw	HPC 53 BimetHeaterN switch on	C				
PYCHPC36	HPC54 on DDStransN1 swit	HPC 54 DDStransN1 switch on	C				
PYCHPC37	HPC55 on DDStransN2 swit	HPC 55 DDStransN2 switch on	C				
PYCHPC38	HPC56 on HeatPLmod01 swi	HPC 56 HeatPLmod01 switch on	C				
PYCHPC39	HPC57 on HeatPLmod02 swi	HPC 57 HeatPLmod02 switch on	C				
PYCHPC3A	HPC58 on HeatCOmod01 swi	HPC 58 HeatCOmod01 switch on	C				
PYCHPC3B	HPC59 on HeatCOmod02 swi	HPC 59 HeatCOmod02 switch on	C				
PYCHPC3C	HPC60 on HeatSRVmod01 sw	HPC 60 HeatSRVmod01 switch on	C				
PYCHPC3D	HPC61 on HeatSRVmod02 sw	HPC 61 HeatSRVmod02 switch on	C				
PYCHPC3E	HPC62 on HeatPLmod11 swi	HPC 62 HeatPLmod11 switch on	C				

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPC3F	HPC63 on HeatPLmod12 swi	HPC 63 HeatPLmod12 switch on	C				
PYCHPC40	HPC64 on HeatCOmod11 swi	HPC 64 HeatCOmod11 switch on	C				
PYCHPC41	HPC65 on HeatCOmod12 swi	HPC 65 HeatCOmod12 switch on	C				
PYCHPC42	HPC66 on HeatSRVmod11 sw	HPC 66 HeatSRVmod11 switch on	C				
PYCHPC43	HPC67 on HeatSRVmod12 sw	HPC 67 HeatSRVmod12 switch on	C				
PYCHPC44	HPC68 on RetaMech N1 swi	HPC 68 RetaMech N1 switch on	C				
PYCHPC45	HPC69 on RetaMech N2 swi	HPC 69 RetaMech N2 switch on	C				
PYCHPC46	HPC70 on RetaMech R1 swi	HPC 70 RetaMech R1 switch on	C				
PYCHPC47	HPC71 on RetaMech R2 swi	HPC 71 RetaMech R2 switch on	C				
PYCHPC48	HPC72 on DOM 1 switch	HPC 72 DOM 1 switch on	C				
PYCHPC49	HPC73 on DOM 2 switch	HPC 73 DOM 2 switch on	C				
PYCHPC4A			C				
PYCHPC4B		PL specific in first FLP mission	C				
PYCHPC4C			C				
PYCHPC4D			C				

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPC4E	HPC78 of OBCcoreN1 switc	HPC 78 PCDU_SWITCHES off	C				
PYCHPC4F	HPC79 of OBCcoreN2 switc	HPC 79 OBCcoreN2 switch off	C				
PYCHPC50	HPC80 of OBCcoreR1 switc	HPC 80 OBCcoreR1 switch off	C				
PYCHPC51	HPC81 of OBCcoreR2 switc	HPC 81 OBCcoreR2 switch off	C				
PYCHPC52	HPC82 of IOboardN1 switc	HPC 82 IOboardN1 switch off	C				
PYCHPC53	HPC83 of IOboardN2 switc	HPC 83 IOboardN2 switch off	C				
PYCHPC54	HPC84 of IOboardR1 switc	HPC 84 IOboardR1 switch off	C				
PYCHPC55	HPC85 of IOboardR2 switc	HPC 85 IOboardR2 switch off	C				
PYCHPC56	HPC86 of CCSDS0 1 switch	HPC 86 CCSDS0 1 switch off	C				
PYCHPC57	HPC87 of CCSDS0 2 switch	HPC 87 CCSDS0 2 switch off	C				
PYCHPC58	HPC88 of CCSDS1 1 switch	HPC 88 CCSDS1 1 switch off	C				
PYCHPC59	HPC89 of CCSDS1 2 switch	HPC 89 CCSDS1 2 switch off	C				
PYCHPC5A	HPC90 of TTC Rec 0 switc	HPC 90 TTC Rec 0 switch off	C				

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPC5B	HPC91 of TTC Rec 1 switc	HPC 91 TTC Rec 1 switch off	C				
PYCHPC5C		PL specific in first FLP mission	C				
PYCHPC5D	HPC93 of PLOC Pow N1 swi	HPC 93 PLOC Pow N1 switch off	C				
PYCHPC5E	HPC94 of PLOC Pow N2 swi	HPC 94 PLOC Pow N2 switch off	C				
PYCHPC5F	HPC95 of DDS N switch	HPC 95 DDS N switch off	C				
PYCHPC60		PL specific in first FLP mission	C				
PYCHPC61	HPC97 of PLOC Pow R1 swi	HPC 97 PLOC Pow R1 switch off	C				
PYCHPC62	HPC98 of PLOC Pow R2 swi	HPC 98 PLOC Pow R2 switch off	C				
PYCHPC63	HPC99 of DDS R switch	HPC 99 DDS R switch off	C				
PYCHPC64		PL specific in first FLP mission	C				
PYCHPC65	HPC101 of TTCtransN1 swi	HPC 101 TTCtransN1 switch off	C				
PYCHPC66	HPC102 of TTCtransN2 swi	HPC 102 TTCtransN2 switch off	C				
PYCHPC67	HPC103 of STR N switch	HPC 103 STR N switch off	C				
PYCHPC68	HPC104 of RW 3 switch	HPC 104 RW 3 switch off	C				
PYCHPC69	HPC105 of STR R switch	HPC 105 STR R switch off	C				
PYCHPC6A	HPC106 of FOG 3 switch	HPC 106 FOG 3 switch off	C				

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPC6B	HPC107 of FOG 0 switch	HPC 107 FOG 0 switch off	C				
PYCHPC6C	HPC108 of RW 0 switch	HPC 108 RW 0 switch off	C				
PYCHPC6D	HPC109 of FOG 1 switch	HPC 109 FOG 1 switch off	C				
PYCHPC6E	HPC110 of RW 1 switch	HPC 110 RW 1 switch off	C				
PYCHPC6F	HPC111 of FOG 2 switch	HPC 111 FOG 2 switch off	C				
PYCHPC70	HPC112 of RW 2 switch	HPC 112 RW 2 switch off	C				
PYCHPC71	HPC113 of MGM 0 switch	HPC 113 MGM 0 switch off	C				
PYCHPC72	HPC114 of MGM 1 switch	HPC 114 MGM 1 switch off	C				
PYCHPC73		PL specific in first FLP mission	C				
PYCHPC74	HPC116 of MGT R switch	HPC 116 MGT R switch off	C				
PYCHPC75	HPC117 of GPS 1 switch	HPC 117 GPS 1 switch off	C				
PYCHPC76	HPC118 of GPS 0 switch	HPC 118 GPS 0 switch off	C				
PYCHPC77	HPC119 of MGT N switch	HPC 119 MGT N switch off	C				
PYCHPC78	HPC120 of GPS 2 switch	HPC 120 GPS 2 switch off	C				
PYCHPC79		PL specific in first FLP mission	C				
PYCHPC7A			C				

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPC7B	HPC123 of TTCtransR1 swi	HPC 123 TTCtransR1 switch off	C				
PYCHPC7C	HPC124 of TTCtransR2 swi	HPC 124 TTCtransR2 switch off	C				
PYCHPC7D		PL specific in first FLP mission	C				
PYCHPC7E			C				
PYCHPC7F	HPC127 of DDStransR1 swi	HPC 127 DDStransR1 switch off	C				
PYCHPC80	HPC128 of DDStransR2 swi	HPC 128 DDStransR2 switch off	C				
PYCHPC81	HPC129 of BimetHeaterR s	HPC 129 BimetHeaterR switch off	C				
PYCHPC82	HPC130 of BimetHeaterN s	HPC 130 BimetHeaterN switch off	C				
PYCHPC83	HPC131 of DDStransN1 swi	HPC 131 DDStransN1 switch off	C				
PYCHPC84	HPC132 of DDStransN2 swi	HPC 132 DDStransN2 switch off	C				
PYCHPC85	HPC133 of HeatPLmod01 sw	HPC 133 HeatPLmod01 switch off	C				
PYCHPC86	HPC134 of HeatPLmod02 sw	HPC 134 HeatPLmod02 switch off	C				
PYCHPC87	HPC135 of HeatCOmod01 sw	HPC 135 HeatCOmod01 switch off	C				
PYCHPC88	HPC136 of HeatCOmod02 sw	HPC 136 HeatCOmod02 switch off	C				

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPC89	HPC137 of HeatSRVmod01 s	HPC 137 HeatSRVmod01 switch off	C				
PYCHPC8A	HPC138 of HeatSRVmod02 s	HPC 138 HeatSRVmod02 switch off	C				
PYCHPC8B	HPC139 of HeatPLmod11 sw	HPC 139 HeatPLmod11 switch off	C				
PYCHPC8C	HPC140 of HeatPLmod12 sw	HPC 140 HeatPLmod12 switch off	C				
PYCHPC8D	HPC141 of HeatCOmod11 sw	HPC 141 HeatCOmod11 switch off	C				
PYCHPC8E	HPC142 of HeatCOmod12 sw	HPC 142 HeatCOmod12 switch off	C				
PYCHPC8F	HPC143 of HeatSRVmod11 s	HPC 143 HeatSRVmod11 switch off	C				
PYCHPC90	HPC144 of HeatSRVmod12 s	HPC 144 HeatSRVmod12 switch off	C				
PYCHPC91	HPC145 of RetaMech N1 sw	HPC 145 RetaMech N1 switch off	C				
PYCHPC92	HPC146 of RetaMech N2 sw	HPC 146 RetaMech N2 switch off	C				
PYCHPC93	HPC147 of RetaMech R1 sw	HPC 147 RetaMech R1 switch off	C				
PYCHPC94	HPC148 of RetaMech R2 sw	HPC 148 RetaMech R2 switch off	C				
PYCHPC95	HPC149 of DOM 1 switch	HPC 149 DOM 1 switch off	C				

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPC96	HPC150 of DOM 2 switch	HPC 150 DOM 2 switch off	C				
PYCHPC97			C				
PYCHPC98			C				
PYCHPC99		PL specific in first FLP mission	C				
PYCHPC9A			C				
PYCHPC9B			C				
PYCHPC9C	HPC156 ActNomCoreCompone	HPC 156 Activate Nominal Core Components (OBC N, I/O-B N)	C				
PYCHPC9D	HPC157 ActRedCoreCompone	HPC 157 Activate Redundant Core Components (OBC R, I/O-B R)	C				
PYCHPC9E	HPC158 Act1stCoreCompCro	HPC 158 Activate 1st core component cross-coupling (OBC N,I/O R)	C				
PYCHPC9F	HPC159 Act2ndCoreCompCro	HPC 159 Activate 2nd core component cross-coupling (OBC R,I/O N)	C				
PYCHPCA0	HPC160 DeaNomCoreCompone	HPC 160 Deactivate Nominal Core Components (OBC N, I/O-B N)	C				
PYCHPCA1	HPC161 DeaRedCoreCompone	HPC 161 Deactivate Redundant Core Components (OBC R, I/O-B R)	C				
PYCHPCA2	HPC162 Dea1stCoreCompCro	HPC 162 Deactivate 1st Core Component CrossCoupl (OBC N,I/O-B R)	C				
PYCHPCA3	HPC163 Dea2ndCoreCompCro	HPC 163 Deactivate 2nd core component CrossCoupl (OBC R,I/O-B N)	C				
PYCHPCA4	HPC164 Deact all Palyoad	HPC 164 Deactivate all Payload switches	C				
PYCHPCA5	HPC165 DeactAllButSafe	HPC 165 Deactivate all switches except Safe mode and survival heater	C				

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PYCHPCA6	HPC166 DeactAllHeaterSwi	HPC 166 Deactivate all heater switches	C				
PYCHPCA7	HPC167 SetDOMactTimeMax	HPC 167 Set DOM activation timer to max value	C				
PCDU TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
PPC00200	Srv(2,130) CheckBattCap0	Check capacity level calculated by PCDU	2	130	53	15	C
PPC00201	Srv(2,130) CheckBattCap1	Check capacity level calculated by PCDU	2	130	53	15	C
PPC00202	Srv(2,130) CheckBattCap2	Check capacity level calculated by PCDU	2	130	53	15	C
PPC002FF	Srv(2,130) CheckBattCapA	Check capacity level calculated by PCDU, all strings	2	130	53	15	C
PPC00300	Srv(2,130) CheckBattSoC0	Check SoC level calculated by PCDU	2	130	53	15	C
PPC00301	Srv(2,130) CheckBattSoC1	Check SoC level calculated by PCDU	2	130	53	15	C
PPC00302	Srv(2,130) CheckBattSoC2	Check SoC level calculated by PCDU	2	130	53	15	C
PPC003FF	Srv(2,130) CheckBattSoCA	Check SoC level calculated by PCDU, all strings	2	130	53	15	C
PPC00400	Srv(2,130) set BatCap S0	Set capacity level calculated by OBC	2	130	53	15	C
PPC00401	Srv(2,130) set BatCap S1	Set capacity level calculated by OBC	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PPC00402	Srv(2,130) set BatCap S2	Set capacity level calculated by OBC	2	130	53	15	C
PPC004FF	Srv(2,130) set BatCap al	Set capacity level calculated by OBC	2	130	53	15	C
PPC00500	Srv(2,130) set BatSoC S0	Set SoC level calculated by OBC	2	130	53	15	C
PPC00501	Srv(2,130) set BatSoC S1	Set SoC level calculated by OBC	2	130	53	15	C
PPC00502	Srv(2,130) set BatSoC S2	Set SoC level calculated by OBC	2	130	53	15	C
PPC005FF	Srv(2,130) set BatSoC al	Set SoC level calculated by OBC	2	130	53	15	C
PPC00600	(2,130)SetEndChar Vol S0	Set End charging Voltage String 0	2	130	53	15	C
PPC00601	(2,130)SetEndChar Vol S1	Set End charging Voltage String 1	2	130	53	15	C
PPC00602	(2,130)SetEndChar Vol S2	Set End charging Voltage String 2	2	130	53	15	C
PPC00700	(2,130)Read EndChar Volt	Read end charging voltage	2	130	53	15	C
PPC00800	Srv(2,130) check Depl	Check deployment status of solar panel mechanisms	2	130	53	15	C
PPC00900	Srv(2,130) BCR 0 off	PCDU Set BCR 0 Off	2	130	53	15	C
PPC00901	Srv(2,130) BCR 0 on	PCDU Set BCR 0 On	2	130	53	15	C
PPC00910	Srv(2,130) BCR 1 off	PCDU Set BCR 1 Off	2	130	53	15	C
PPC00911	Srv(2,130) BCR 1 on	PCDU Set BCR 1 On	2	130	53	15	C
PPC00920	Srv(2,130) BCR 2 off	PCDU Set BCR 2 Off	2	130	53	15	C
PPC00921	Srv(2,130) BCR 2 on	PCDU Set BCR 2 On	2	130	53	15	C
PPC00A00	Srv(2,130) BCR status	Check BCR status	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PPC00B00	(2,130) SetHoldTimeOBCini	PCDU Set initial holding time before starting recovery process o	2	130	53	15	C
PPC00C00	(2,130) ReadHolTimeOBCini	PCDU Read holding time before recovery process OBC-I/O board [s]	2	130	53	15	C
PPC00D00	(2,130) SetHoldTimeOBCbet	PCDU Set holding time in between rec. process of OBC and I/O [s]	2	130	53	15	C
PPC00E00	(2,130) ReadHolTimeOBCbet	PCDU Read holding time in between rec. process OBC-I/O board [s]	2	130	53	15	C
PPC01100	(2,130)SetSoCLvlR1R2	Adjust SoC lvl for mode trans from 'rec. mode0' to 'rec. mode1'	2	130	53	15	C
PPC01200	(2,130) ReadSoCThreshR1R2	Check SoC lvl for mode trans from 'rec. mode0' to 'rec. mode1'	2	130	53	15	C
PPC01300	Srv(2,130) PCDUmodeMiniO	Mode Transition: MiniOps	2	130	53	15	C
PPC01301	Srv(2,130) PCDUmode Idle	Mode Transition: Idle	2	130	53	15	C
PPC01302	Srv(2,130) PCDUmodeShutD	Mode Transition: Shut Down	2	130	53	15	C
PPC01303	Srv(2,130) PCDUmodeNone	Mode Transition: None	2	130	53	15	C
PPC01400	Srv(2,130) PCDUcheckMode	Check status of PCDU mode	2	130	53	15	C
PPC01600	Srv(2,130) BistRelay0 Of	Command bi-stable relay 0 Off	2	130	53	15	C
PPC01601	Srv(2,130) BistRelay0 On	Command bi-stable relay 0 On	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PPC01610	Srv(2,130) BistRelay1 Of	Command bi-stable relay 1 Off	2	130	53	15	C
PPC01611	Srv(2,130) BistRelay1 On	Command bi-stable relay 1 On	2	130	53	15	C
PPC01700	Srv(2,130) BistRelStatus	Read status of bi-stable relay	2	130	53	15	C
PPC01800	(2,130) SetCPUToggleTime	Set PCDU CPUs waiting time before CPU toggle	2	130	53	15	C
PPC01900	(2,130) ReadCPUToggleTime	Check PCDU CPUs waiting time before CPU toggle	2	130	53	15	C
PPC01A00	(2,130)SetHolTimeTrans	Set holding time for mode transition from RM1 to MOM [s]	2	130	53	15	C
PPC01B00	(2,130) ReadHolTimeTrans	Check holding time for mode transition from RM1 to MOM [s]	2	130	53	15	C
PPC01C00	Srv(2,130) PCDUchangeCPU	Switch to other CPU	2	130	53	15	C
PPC01D00	Srv(2,130) PCDUcheckCPU	Check and respond operating CPU	2	130	53	15	C
PPC01E00	Srv(2,130) PCDUswitch Of	Change status of switch: Off	2	130	53	15	C
PPC01E01	Srv(2,130) PCDUswitch On	Change status of switch: On	2	130	53	15	C
PPC02000	(2,130)SetMax CompoCurr	Adjust value for max. current of the component on this switch	2	130	53	15	C
PPC02001	(2,130) ReadMaxCompCurr	Check value for max. current of the component on this switch	2	130	53	15	C
PPC02100	Srv(2,130) PCDU fuse On	Command status of fuse: on	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PPC02300	Srv(2,130) ReadFusesCurr	Read current of all fuses	2	130	53	15	C
PPC02900	Srv(2,130) InitTesStrMea	PCDU Initiate Test String Measurement	2	130	53	15	C
PPC02A00	Srv(2,130) ReadTesStrTM	PCDU Read Test String TM	2	130	53	15	C
PPC02C00	Srv(2,130) PCDUreadRAMhl	PCDU read RAM history log	2	130	53	15	C
PPC02CFF	Srv(2,130) PCDUcleaRAMhl	PCDU clear RAM history log	2	130	53	15	C
PPC02F00	(2,130) SetDeployAcFlagON	Set Activation flag for deployment to ON	2	130	53	15	C
PPC02F01	(2,130) SetDeplActFlagOFF	Set Activation flag for deployment to OFF	2	130	53	15	C
PPC02F03	(2,130) ReadDeployActFlag	Check activation flag for deployment	2	130	53	15	C
PPC03000	Srv(2,130) PCDUreadFLAhl	PCDU read FLASH history log	2	130	53	15	C
PPC030FF	Srv(2,130) PCDUcleaFLAhl	PCDU clear FLASH history log	2	130	53	15	C
PPC03200	Srv(2,130) PCDUreadWOCOl	PCDU read Working Components log	2	130	53	15	C
PPC0328F	Srv(2,130) PCDUreadWOCOa	PCDU read actual Working Components	2	130	53	15	C
PPC032FF	Srv(2,130) PCDUcleaWOCOl	PCDU clear Working Components Log	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PPC03300	(2,130)Set DeployTimer 0	Set the deploy timer 0 into a time in seconds	2	130	53	15	C
PPC03301	(2,130)Set DeployTimer 1	Set the Timer 1 for deployment in seconds	2	130	53	15	C
PPC03400	(2,130)Read DeployTimer0	Read deploy timer 0	2	130	53	15	C
PPC03401	(2,130) ReadDeployTimer 1	Read deploy timer 1	2	130	53	15	C
PPC03500	Srv(2,130) PCDU get TM	Get periodic TM Packet	2	130	53	15	C
PPC03600	(2,130) SetBatActFlagOff0	Set Battery Charge activation Flag Off for String 0	2	130	53	15	C
PPC03601	(2,130) SetBatActFlagOff1	Set Battery Charge activation Flag Off for String 1	2	130	53	15	C
PPC03602	(2,130) SetBatActFlagOff2	Set Battery Charge activation Flag Off for String 2	2	130	53	15	C
PPC03610	(2,130) SetBatActFlagOn0	Set Battery Charge activation Flag On for String 0	2	130	53	15	C
PPC03611	(2,130) SetBatActFlagOn1	Set Battery Charge activation Flag On for String 1	2	130	53	15	C
PPC03612	(2,130) SetBatActFlagOn2	Set Battery Charge activation Flag On for String 2	2	130	53	15	C
PPC03700	(2,130) ReadBatCharActFl	Read Battery Charge Activation Flag	2	130	53	15	C
PPC03800	(2,130) SetBatCharOffTime	Set Battery charge deactivation time	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

PPC03900	(2,130) ReadBatDeacTime	Read battery charge deactivation time	2	130	53	15	C
PPC03A00	Srv(2,130) PCDUreadDOMtim	Read DOM Timer in PCDU	2	130	53	15	C
PPC03B00	Srv(2,130) PCDUsetDOMtim	Set DOM Timer in PCDU	2	130	53	15	C
PPC03C00	Srv(2,130) Set SoC Param	PCDU Set SoC Parameters	2	130	53	15	C
PPC03D00	Srv(2,130) CheckSoCparam	PCDU Check SoC Parameters	2	130	53	15	C
PPC03E00	Srv(2,130) SetBattMaxCap	PCDU Set Battery Maximum Capacity	2	130	53	15	C
PPC03F00	Srv(2,130) ChekBatMaxCap	PCDU Check Battery Maximum Capacity	2	130	53	15	C
Attitude Control Subsystem TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
AYC00000	Srv8 ACS Activate ACS	8	1	42	9	C	
AYC00001	Srv8 ACS Deactivate ACS	8	1	42	9	C	
AYC00002	Srv8 ACS Upload 1 Const.	8	1	42	9	C	
AYC00004	Srv8 ACS Mode -> Detumbl	8	1	42	9	C	
AYC00005	Srv8 ACS Mode -> Idle	8	1	42	9	C	
AYC00006	Srv8 ACS Mode -> NadirPt	8	1	42	9	C	

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

AYC00007	Srv8 ACS Mode -> InertPt	8	1	42	9	C	
AYC00008	Srv8 ACS Mode -> TargetP	8	1	42	9	C	
Magnetometer TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
AMC00050	Srv(2,130)MGM N ReqMeasS	MGM N request measurement with slow sampling	2	130	53	15	C
AMC00051	Srv(2,130)MGM N ReqMeasM	MGM N request measurement with medium sampling	2	130	53	15	C
AMC00052	Srv(2,130)MGM N ReqMeasF	MGM N request measurement with fast sampling	2	130	53	15	C
AMC00098	Srv(2,130)MGM N PowerDow	MGM N power down	2	130	53	15	C
AMC00099	Srv(2,130)MGM N Power Up	MGM N power up	2	130	53	15	C
AMC00150	Srv(2,130)MGM R ReqMeasS	MGM R request measurement with slow sampling	2	130	53	15	C
AMC00151	Srv(2,130)MGM R ReqMeasM	MGM R request measurement with medium sampling	2	130	53	15	C
AMC00152	Srv(2,130)MGM R ReqMeasF	MGM R request measurement with fast sampling	2	130	53	15	C
AMC00198	Srv(2,130)MGM R PowerDow	MGM R power down	2	130	53	15	C
AMC00199	Srv(2,130)MGM R Power Up	MGM R power up	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

Magnetotorquer TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
AQC00100	Srv(2,130) MGTNsetCoilSta	MGT N Set Coil State and Read Status	2	130	53	15	C
AQC00101	Srv(2,130) MGTRsetCoilSta	MGT R Set Coil State and Read Status	2	130	53	15	C
Star Tracker TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
ASC00007	Srv(2,130) STR ReqStatus	AscRequestStatusTC	2	130	53	15	C
ASC00008	Srv(2,130)STR LoadParame	AscLoadParameterTC	2	130	53	15	C
ASC00009	Srv(2,130)STR storeImage	AscStoreImageTC	2	130	53	15	C
ASC00010	Srv(2,130)STR sendImage	AscSendImageTC	2	130	53	15	C
ASC00018	Srv(2,130)STR Time TC	AscTimeTC	2	130	53	15	C
ASC00019	Srv(2,130)STR Pvt TC	AscPvtTC	2	130	53	15	C
ASC00100	Srv(2,130)STR MIRUctrlOn	AscMiruControlTC	2	130	53	15	C
ASC00101	Srv(2,130)STR MIRUctrlOf	AscMiruControlTC Off	2	130	53	15	C
ASC00102	Srv(2,130)STR MIRUstoreB	AscMiruStoreDataTc CHU B	2	130	53	15	C
ASC00103	Srv(2,130)STR MIRUdumpDa	AscMiruDumpDataTC	2	130	53	15	C
ASC00104	Srv(2,130)STR MIRUclear	AscMiruClearDataTC	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

ASC10000	Srv(17,1) STR Ping	AscDummyTC	17	1	1937	C	
ASC10001	Srv8raw STR StandbyMode	AscStandbyModeTC	8	1	1937	9	C
ASC10002	Srv8raw STR AttitudeMode	AscAttitudeModeTC	8	1	1937	9	C
ASC10003	Srv6rawSTR Dump Memory	AscDumpMemoryTC	6	5	1937	9	C
ASC10004	Srv6raw STR Load Memory	AscLoadMemoryTC	6	2	1937	9	C
ASC10006	Srv8raw STR Boot TC	AscBootTC	8	1	1937	9	C
ASC10007	Srv6rawSTR ReqStatus	AscRequestStatusTC	6	5	1937	9	C
ASC10008	Srv6rawSTR LoadParameter	AscLoadParamterTC	6	2	1937	9	C
ASC10009	Srv8raw STR Store Image	AscStoreImageTC	8	1	1937	9	C
ASC10010	Srv6rawSTR Send Image	AscSendImageTC	6	5	1937	9	C
ASC10011	Srv8raw STR Sim Data	AscSimulationDataTC	8	1	1937	9	C
ASC10012	Srv8raw STR SimulatMode	AscSimulationModeTC	8	1	1937	9	C
ASC10013	Srv8rawSTR TestImageMode	AscTestImageModeTC	8	1	1937	9	C
ASC10014	Srv8raw STR HK force	AscHKForceTC	8	1	1937	9	C
ASC10017	Srv6raw STR Check Memory	AscCheckMemoryTC	6	9	1937	9	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

ASC10018	Srv6raw STR Time TC	AscTimeTC	9	129	1937	9	C
ASC10019	Srv9raw STR Pvt TC	AscPvtTC	9	132	1937	9	C
Fiber-optic Gyro TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
AFC00000	Srv(2,130) FOG req Data	FOG Request Measurement Data	2	130	53	15	C
GPS TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
AGC0AM00	(2,130) GPS0 SetAidingMo	GPS0 Set Aiding Mode (AM)	2	130	53	15	C
AGC0AM01	(2,130) GPS1 SetAidingMo	GPS1 Set Aiding Mode (AM)	2	130	53	15	C
AGC0AM02	(2,130) GPS2 SetAidingMo	GPS2 Set Aiding Mode (AM)	2	130	53	15	C
AGC0DR00	(2,130) GPS0 SetDataRate	GPS0 Set Data Rate (DR)	2	130	53	15	C
AGC0DR01	(2,130) GPS1 SetDataRate	GPS1 Set Data Rate (DR)	2	130	53	15	C
AGC0DR02	(2,130) GPS2 SetDataRate	GPS2 Set Data Rate (DR)	2	130	53	15	C
AGC0DW00	(2,130) GPS0 SetDoppWin	GPS0 Set Doppler Window (DW)	2	130	53	15	C
AGC0DW01	(2,130) GPS1 SetDoppWin	GPS1 Set Doppler Window (DW)	2	130	53	15	C
AGC0DW02	(2,130) GPS2 SetDoppWin	GPS2 Set Doppler Window (DW)	2	130	53	15	C
AGC0LO00	(2,130) GPS0 Load TLE	GPS0 Load TLE (LO)	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

AGC0LO01	(2,130) GPS1 Load TLE	GPS1 Load TLE (LO)	2	130	53	15	C
AGC0LO02	(2,130) GPS2 Load TLE	GPS2 Load TLE (LO)	2	130	53	15	C
AGC0TM00	(2,130) GPS0 SetTrackMod	GPS0 Set Tracking Mode (TM)	2	130	53	15	C
AGC0TM01	(2,130) GPS1 SetTrackMod	GPS1 Set Tracking Mode (TM)	2	130	53	15	C
AGC0TM02	(2,130) GPS2 SetTrackMod	GPS2 Set Tracking Mode (TM)	2	130	53	15	C
AGCCDM00	(2,130) GPSCDM ReadCloc	GPS CDM Read Clock Source	2	130	53	15	C
AGCXCT00	Srv(2,130) GPS0 XNS Mode	GPS0 XNS Mode Control (XNCTL)	2	130	53	15	C
AGCXCT01	Srv(2,130) GPS1 XNS Mode	GPS1 XNS Mode Control (XNCTL)	2	130	53	15	C
AGCXCT02	Srv(2,130) GPS2 XNS Mode	GPS2 XNS Mode Control (XNCTL)	2	130	53	15	C
AGCXP100	Srv(2,130)GPS0 XNS ListP	GPS0 XNS List Parameters (XNPAR1)	2	130	53	15	C
AGCXP101	Srv(2,130)GPS1 XNS ListP	GPS1 XNS List Parameters (XNPAR1)	2	130	53	15	C
AGCXP102	Srv(2,130)GPS2 XNS ListP	GPS2 XNS List Parameters (XNPAR1)	2	130	53	15	C
AGCXP200	Srv(2,130)GPS0 XNS DefPa	GPS0 XNS Restore Default Parameters (XNPAR2)	2	130	53	15	C
AGCXP201	Srv(2,130)GPS1 XNS DefPa	GPS1 XNS Restore Default Parameters (XNPAR2)	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

AGCXP202	Srv(2,130)GPS2 XNS DefPa	GPS2 XNS Restore Default Parameters (XNPAR2)	2	130	53	15	C
AGCXP300	Srv(2,130)GPS0 XNS SetPa	GPS0 XNS Set Parameter (XNPAR3)	2	130	53	15	C
AGCXP301	Srv(2,130)GPS1 XNS SetPa	GPS1 XNS Set Parameter (XNPAR3)	2	130	53	15	C
AGCXP302	Srv(2,130)GPS2 XNS SetPa	GPS2 XNS Set Parameter (XNPAR3)	2	130	53	15	C
Reaction Wheel TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
ARC03000	Srv(2,130) RW0 Reset	RW0 Reset Wheel	2	130	53	15	C
ARC03001	Srv(2,130) RW1 Reset	RW1 Reset Wheel	2	130	53	15	C
ARC03002	Srv(2,130) RW2 Reset	RW2 Reset Wheel	2	130	53	15	C
ARC03003	Srv(2,130) RW3 Reset	RW3 Reset Wheel	2	130	53	15	C
ARC03100	Srv(2,130) RW0 Send TM	RW0 Send RW Telemetry	2	130	53	15	C
ARC03101	Srv(2,130) RW1 Send TM	RW1 Send RW Telemetry	2	130	53	15	C
ARC03102	Srv(2,130) RW2 Send TM	RW2 Send RW Telemetry	2	130	53	15	C
ARC03103	Srv(2,130) RW3 Send TM	RW3 Send RW Telemetry	2	130	53	15	C
ARC03200	Srv(2,130) RW0 Disable	RW0 Disable Wheel	2	130	53	15	C
ARC03201	Srv(2,130) RW1 Disable	RW1 Disable Wheel	2	130	53	15	C
ARC03202	Srv(2,130) RW2 Disable	RW2 Disable Wheel	2	130	53	15	C
ARC03203	Srv(2,130) RW3 Disable	RW3 Disable Wheel	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

ARC03300	Srv(2,130) RW0 Enable	RW0 Enable Wheel	2	130	53	15	C
ARC03301	Srv(2,130) RW1 Enable	RW1 Enable Wheel	2	130	53	15	C
ARC03302	Srv(2,130) RW2 Enable	RW2 Enable Wheel	2	130	53	15	C
ARC03303	Srv(2,130) RW3 Enable	RW3 Enable Wheel	2	130	53	15	C
ARC03400	Srv(2,130) RW0 SetTorque	RW0 SetTorque	2	130	53	15	C
ARC03401	Srv(2,130) RW1 SetTorque	RW1 SetTorque	2	130	53	15	C
ARC03402	Srv(2,130) RW2 SetTorque	RW2 SetTorque	2	130	53	15	C
ARC03403	Srv(2,130) RW3 SetTorque	RW3 SetTorque	2	130	53	15	C
ARC03500	Srv(2,130) RW0 SetSpeed	RW0 SetSpeed	2	130	53	15	C
ARC03501	Srv(2,130) RW1 SetSpeed	RW1 SetSpeed	2	130	53	15	C
ARC03502	Srv(2,130) RW2 SetSpeed	RW2 SetSpeed	2	130	53	15	C
ARC03503	Srv(2,130) RW3 SetSpeed	RW3 SetSpeed	2	130	53	15	C
Thermal Control Subsystem TCs							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
TYC00001	(8,1)TCS SetTargStatComp	Srv (8,1) TCS Set Target State Component	8	1	53	9	C
TYC00002	(8,1)TCS SetTargStatModu	Srv (8,1) TCS Set Target State Module	8	1	53	9	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

TYC00100	(12,137) SetTempSensLimit	TCS Set Temperature Sensor Limits	12	137	53	11	C
TYC00101	(12,137) SetCompTempLimit	TCS Set Component Temperature Limits	12	137	53	11	C
TYC00110	(12,137) SetBatS0TmpLimit	TCS Set Battery String 0 Temperature Limits	12	137	53	11	C
TYC00111	(12,137) SetBatS1TmpLimit	TCS Set Battery String 1 Temperature Limits	12	137	53	11	C
TYC00112	(12,137) SetBatS2TmpLimit	TCS Set Battery String 2 Temperature Limits	12	137	53	11	C
TYC00120	(12,137)SetMGTN TmpLimit	TCS Set MGT N Temperature Limits	12	137	53	11	C
TYC00121	(12,137)SetMGTR TmpLimit	TCS Set MGT R Temperature Limits	12	137	53	11	C
TYC00130	(12,137)Set STR TmpLimit	TCS Set STR Temperature Limits	12	137	53	11	C
TYC00140	(12,137) SetPLOCNTmpLimit	TCS Set PLOC N Temperature Limits	12	137	53	11	C
TYC00141	(12,137) SetPLOCRTmpLimit	TCS Set PLOC R Temperature Limits	12	137	53	11	C
TYC60000	Srv6 LoadHeaterTimeout	Memory Load for heater timeout to TCS Controller	6	132	53	11	C
TYC60001	Srv6 LoadComponentConfig	Memory Load for Control Component Configuration to TCS Controller	6	132	53	11	C
TYC61000	Srv6 DumpHeaterTimeout	Memory Dump for heater timeout to TCS Controller	6	135	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

TYC61001	Srv6 DumpComponentConfig	Memory Dump for Control Component Configuration to TCS Controller	6	135	53	11	C
Payload Computer TCs (FLP Generation 1)							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
LRC00000	(2,130)PB N SetResetLine	Set PLOC Power-Board N Reset Line to high	2	130	53	15	C
LRC00001	(2,130)PB R SetResetLine	Set PLOC Power-Board R Reset Line to high	2	130	53	15	C
LRC02010	(2,130)VTX Get Version	PLOC VTX Get Version	2	130	53	15	C
LRC02020	(2,130)VTX Loopback Data	PLOC VTX Loopback Data from Command	2	130	53	15	C
LRC02110	(2,130)VTX GetRoutSettin	PLOC VTX Get MUX (data routing) Setting	2	130	53	11	C
LRC02111	(2,130)VTX Set Rout All	PLOC VTX Set MUX All	2	130	53	11	C
LRC02112	(2,130)VTX Set RoutSingl	PLOC VTX Set MUX (Data Routing) Single	2	130	53	11	C
LRC02130	(2,130)VTX Set Pattern	PLOC VTX Pattern Generator Set Pattern	2	130	53	11	C
LRC02131	(2,130) VTXEnableFixPatte	PLOC VTX Pattern Generator Enable Fix Pattern	2	130	53	11	C
LRC02132	(2,130)VTX DisablePatter	PLOC VTX Pattern Generator Disable Pattern	2	130	53	11	C
LRC02133	(2,130) VTXEnablCountPatt	PLOC VTX Pattern Generator Enable Counter Pattern	2	130	53	11	C
LRC02134	(2,130) VTXPatternGetInfo	PLOC VTX Pattern Generator: Get Info	2	130	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

LRC02201	(2,130)VTX DDSPRTogPRSta	PLOC VTX DDS Protocol Toggle Pseudorandom Bypass State	2	130	53	11	C
LRC02202	(2,130)VTX DDSPRGetPRSta	PLOC VTX DDS Protocol Get Pseudo Random Bypass State	2	130	53	11	C
LRC02211	(2,130)VTX DDSPRTogTxSta	PLOC VTX DDS Protocol Toggle Transmit State	2	130	53	11	C
LRC02212	(2,130)VTX DDSPRGetTxSta	PLOC VTX DDS Protocol Get Transmit State	2	130	53	11	C
LRC023EA	(2,130)VTX SW Reset	PLOC VTX Software Reset	2	130	53	11	C
LRC02401	(2,130)VTX GetDebugSpeed	PLOC VTX Debug Get Speed	2	130	53	11	C
LRC02402	(2,130)VTX GetDebugSourc	PLOC VTX Debug Get Pin Source	2	130	53	11	C
LRC02411	(2,130)VTX SetDebugSpeed	PLOC VTX Debug Set Speed	2	130	53	11	C
LRC02412	(2,130)VTX SetDebugSourc	PLOC VTX Debug Set Pin Source	2	130	53	11	C
LRC024CC	(2,130)VTX Get DebugInfo	PLOC VTX Debug Get Info	2	130	53	11	C
LRC02A10	(2,130)MMU Clock Off	PLOC MMU Switch Clock Off	2	130	53	15	C
LRC02A11	(2,130)MMU Clock On	PLOC MMU Switch Clock On	2	130	53	15	C
LRC02A12	(2,130)MMU GetClockState	PLOC MMU Get Clock State	2	130	53	15	C
LRC02A13	(2,130)MMU SetClockSpeed	PLOC MMU Set Clock Speed	2	130	53	15	C
LRC02A14	(2,130)MMU GetClockSpeed	PLOC MMU Get Clock Speed	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

LRC02A15	(2,130)MMU ActivateReset	PLOC MMU Activate Reset Line	2	130	53	15	C
LRC02A16	(2,130)MMU DeactivaReset	PLOC MMU Deactivate Reset Line	2	130	53	15	C
LRC02A17	(2,130)MMU GetResetState	PLOC MMU Get Reset State	2	130	53	15	C
LRC02A18	(2,130)MMU StopCurrActiv	PLOC MMU Stop Current Activity	2	130	53	15	C
LRC02A19	(2,130)MMU GetMonitoData	PLOC MMU Get Monitoring Data	2	130	53	15	C
LRC02AA1	(2,130)MMU Write Data	PLOC MMU Write Data to MMU	2	130	53	15	C
LRC02AA2	(2,130)MMU Read Data	PLOC MMU Read Data from MMU	2	130	53	15	C
LRC02AB1	(2,130)MMU GetPictureLis	PLOC MMU Get Picture List	2	130	53	15	C
LRC02AB2	(2,130)MMU GetBadBlockLi	PLOC MMU Get Bad Block List	2	130	53	15	C
LRC02AB3	(2,130)MMU GetPicBlockLi	PLOC MMU Get Picture Block List	2	130	53	15	C
LRC02AB4	(2,130)MMU GetFulBlockLi	PLOC MMU Get Full Block List	2	130	53	15	C
LRC02AC1	(2,130)MMU DeleteDataSet	PLOC MMU Delete Data Set	2	130	53	15	C
LRC02AC2	(2,130)MMU Format Flash	PLOC MMU Format Flash Chip	2	130	53	15	C
LRC02AC3	(2,130)MMU Reset Flash	PLOC MMU Reset Flash Chip	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

LRC02B0C	(2,130)VTX GetSSRAM info	PLOC VTX Get SSRAM Information	2	130	53	11	C
LRC06200	(2,130)CC N LoadFlashDat	PLOC CC N Load Flash Data	2	130	53	15	C
LRC06201	(2,130)CC R LoadFlashDat	PLOC CC R Load Flash Data	2	130	53	15	C
LRC06300	(2,130)CC N EraseProgram	PLOC CC N Erase Program	2	130	53	15	C
LRC06301	(2,130)CC R EraseProgram	PLOC CC R Erase Program	2	130	53	15	C
LRC06400	(2,130)CC N ReadRegister	PLOC CC N Read Register	2	130	53	15	C
LRC06401	(2,130)CC R ReadRegister	PLOC CC R Read Register	2	130	53	15	C
LRC06500	(2,130)CC N WriteRegiste	PLOC CC N Write Register	2	130	53	15	C
LRC06501	(2,130)CC R WriteRegiste	PLOC CC R Write Register	2	130	53	15	C
LRC06600	(2,130)CC N SwtchWatchdo	PLOC CC N Switch Watchdog State	2	130	53	15	C
LRC06601	(2,130)CC R SwtchWatchdo	PLOC CC R Switch Watchdog State	2	130	53	15	C
LRC06700	(2,130)CC N ClearTempAle	PLOC CC N Clear Temperature Sensor Alert	2	130	53	15	C
LRC06701	(2,130)CC R ClearTempAle	PLOC CC R Clear Temperature Sensor Alert	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

LRC06800	(2,130)CC N ReadFirmwVer	PLOC CC N Read Firmware Version	2	130	53	15	C
LRC06801	(2,130)CC R ReadFirmwVer	PLOC CC R Read Firmware Version	2	130	53	15	C
LRC06900	(2,130)CC N ProgramFlash	PLOC CC N Program Flash	2	130	53	15	C
LRC06901	(2,130)CC R ProgramFlash	PLOC CC R Program Flash	2	130	53	15	C
Science Data Downlink System TCs (FLP Generation 1)							
Name	**Description**	**Long Description**	**Type**	**Subt.**	**APID**	**Ack**	**CTYP**
LDC00000	(2,130)DDSN SetDataSpeed	DDS Data Routing N Set Data Speed	2	130	53	11	C
LDC00001	(2,130)DDSR SetDataSpeed	DDS Data Routing R Set Data Speed	2	130	53	11	C
LDCGDC00	Srv(2,130) DDSNGetDefConf	DDS N Get Default Configuration (GDC) TC	2	130	53	15	C
LDCGDC01	Srv(2,130) DDSRGetDefConf	DDS R Get Default Configuration (GDC) TC	2	130	53	15	C
LDCGMI00	Srv(2,130) DDSNGetDefMoID	DDS N Get Default Module Identification (GMI) TC	2	130	53	15	C
LDCGMI01	Srv(2,130) DDSRGetDefMoID	DDS R Get Default Module Identification (GMI) TC	2	130	53	15	C
LDCGMI10	Srv(2,130) DDSN GetModID	DDS N Get Module Identification (GMI) TC	2	130	53	15	C
LDCGMI11	Srv(2,130) DDSR GetModID	DDS R Get Module Identification (GMI) TC	2	130	53	15	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

LDCGRG00	Srv(2,130) DDSN GetRegis	DDS N Get Register (GRG) TC	2	130	53	15	C
LDCGRG01	Srv(2,130) DDSR GetRegis	DDS R Get Register (GRG) TC	2	130	53	15	C
LDCGRT00	Srv(2,130) DDSNGetRegTemp	DDS N Get Register Temporary (GRT) TC	2	130	53	15	C
LDCGRT01	Srv(2,130) DDSRGetRegTemp	DDS R Get Register Temporary (GRT) TC	2	130	53	15	C
LDCGSR00	Srv(2,130) DDSNGetStatReg	DDS N Get Status Register (GSR) TC	2	130	53	15	C
LDCGSR01	Srv(2,130) DDSRGetStatReg	DDS R Get Status Register (GSR) TC	2	130	53	15	C
LDCMFW00	Srv(2,130) DDSN MessForw	DDS N Message Forwarding (MFW) TC	2	130	53	11	C
LDCMFW01	Srv(2,130) DDSR MessForw	DDS R Message Forwarding (MFW) TC	2	130	53	11	C
LDCRST00	Srv(2,130) DDS N Reset	DDS N Reset (RST) Command, not the PLOC reset!	2	130	53	11	C
LDCRST01	Srv(2,130) DDS R Reset	DDS R Reset (RST) Command, not the PLOC reset!	2	130	53	11	C
LDCSAC00	Srv(2,130) DDSN SetAddre	DDS N Set Address TC (SAC)	2	130	53	11	C
LDCSAC01	Srv(2,130) DDSR SetAddre	DDS R Set Address TC (SAC)	2	130	53	11	C
LDCSDC00	Srv(2,130) DDSNSetDefConf	DDS N Set Default Configuration (SDC) TC	2	130	53	11	C
LDCSDC01	Srv(2,130) DDSRSetDefConf	DDS R Set Default Configuration (SDC) TC	2	130	53	11	C

(continued)

Table 17.2 S/C telecommand packet definitions (continued)

LDCSMI00	Srv(2,130) DDSN SetModId	DDS N Set Module Identification (SMI) TC	2	130	53	11	C
LDCSMI01	Srv(2,130) DDSR SetModId	DDS R Set Module Identification (SMI) TC	2	130	53	11	C
LDCSRG00	Srv(2,130) DDSN SetRegis	DDS N Set Register (SRG) TC	2	130	53	11	C
LDCSRG01	Srv(2,130) DDSR SetRegis	DDS R Set Register (SRG) TC	2	130	53	11	C
LDCSRT00	Srv(2,130) DDSNSetRegTemp	DDS N Set Register Temporary (SRT) TC	2	130	53	11	C
LDCSRT01	Srv(2,130) DDSRSetRegTemp	DDS R Set Register Temporary (SRT) TC	2	130	53	11	C

17.3.2 Telemetry Definitions

See Table 17.3.

17.3.3 Event Telemetry Definitions

The complete list of events is included in Table 17.4. It needs to be noted that in general an event can be triggered by several different objects on-board, but there are also device specific events, which can only be triggered by one object. An example of the former case would be the DEVICE_INTERPRETING_REPLY_FAILED event, which indicates that the reply from a device could not be interpreted correctly by a device handler; this event can be triggered by nearly all device handlers. An event which can only triggered by one object is for example the PANELS_DEPLOYED event, which indicates that the OBSW has determined that the solar panels are deployed; it can only be triggered by the PSS Controller.

Table 17.4 lists the events with their name in the OBSW (column "Event Name"), ID (also called Report ID or RID in [37]) and a description. It only shows the object-ID of the triggering object for those events, which can only be triggered by one object (column "Source ObjID"). Note that the object-ID 0 stands for the general OBSW. The first two digits of the RID indicate the subsystem, the events are assigned to, according to the column "SPID Range" in Table 16.1.

Table 17.4 also lists the events' severity according to [37], namely

- "I" for info events,
- "L" for low,
- "M" for medium and
- "H" for high severity events.

Furthermore, each events comprises two 4 byte parameters which can contain additional information on the event. The contents of these parameters are also given in the table (columns "Parameter 1" and "Parameter 2").

Table 17.3 S/C telemetry packet definitions

System TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
YYTE7800ping	7800	Srv (17,2) Test Ping Packet	53	17	2	0	0	−1	PING
System Manager TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
YMFC7400heal	7400	Srv (201,3) All Objects' health Flags Report	53	201	3	0	0	7400	YMC22102
YMFC7500avta	7500	Srv (202,6) Available Mode Change Tables Report	53	202	6	0	0	7500	YMC22205, YMC24205
YMFC7501avsq	7501	Srv (202,8) Available Mode Change Sequences Report	53	202	8	0	0	7501	YMC22207, YMC24207
YMFC7502tada	7502	Srv (202,10) Mode Change Table Data Report	53	202	10	0	0	7502	YMC22209, YMC24209
YMFC7503sqda	7503	Srv (202,12) Mode Change Sequence Data Report	53	202	12	0	0	7503	YMC22211, YMC24211
YMFC7504mesl	7504	Srv (202,14) Available Memory Slots for Mode Change Tables&Sequence	53	202	14	0	0	−1	YMC22213
YMFC7505metb	7505	Srv (202,16) Available memory slots for Object ModeChange Tables	53	202	16	0	0	−1	YMC22215, YMC24215
YMFC7506mesq	7506	Srv (202,18) Available memory slots for Object ModeChange Sequence	53	202	18	0	0	−1	YMC22217, YMC24217

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

Time Managing TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
YTHK7700time	37700	Srv (3,25) Basic Time Information	53	3	25	7700	0	−1	
Housekeeping TM									
Name	**SPID**	**Description**						**SID**	
AFHK1000data	31000	Srv (3,25) FOG Rate Measurements						1000	
AGHK1100f40m	31100	Srv (3,25) GPS0 TM found in the F40 message						1100	
AGHK1101f40m	31101	Srv (3,25) GPS1 TM found in the F40 message						1101	
AGHK1102f40m	31102	Srv (3,25) GPS2 TM found in the F40 message						1102	
AGHK1103f620	31103	Srv (3,25) GPS0 receiver raw data found in F62 message						1103	
AGHK1104f620	31104	Srv (3,25) GPS0 receiver raw data found in F62 message part 2						1104	
AGHK1105f621	31105	Srv (3,25) GPS1 receiver raw data found in F62 message						1105	
AGHK1106f621	31106	Srv (3,25) GPS1 receiver raw data found in F62 message part 2						1106	
AGHK1107f622	31107	Srv (3,25) GPS2 receiver raw data found in F62 message						1107	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

AGHK1108f622	31108	Srv (3,25) GPS2 receiver raw data found in F62 message, part 2						1108	
AGHK1109f80m	31109	Srv (3,25) GPS0 TM found in the F80 message: XNS **fi**ltered navig.						1109	
AGHK1110f80m	31110	Srv (3,25) GPS1 TM found in the F80 message: XNS **fi**ltered navig.						1110	
AGHK3111f80m	31111	Srv (3,25) GPS2 TM found in the F80 message: XNS **fi**ltered navig.						1111	
AMHK1200data	31200	Srv (3,25) MGM Data						1200	
AQHK1300data	31300	Srv (3,25) MGT Data						1300	
ARHK1400data	31400	Srv (3,25) RW Data						1400	
ASHK1500atti	31500	Srv (3,25) STR Attitude TM						1500	
ASHK1501hktm	31501	Srv (3,25) STR Housekeeping TM						1501	
ASHK1502miru	31502	Srv (3,25) STR MIRU TM						1502	
AUHK1600curr	31600	Srv (3,25) SuS Currents Measured by the PCDU						1600	
AYHK1800chse	31800	Srv (3,25) ACS Controller Checked Sensor Values						1800	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

AYHK1801calc	31801	Srv (3,25) ACS System attitude and orbit calc data						1801	
DCHK2000data	32000	Srv (3,25) CCSDS Board Housekeeping Data						2000	
DSHK2602swvr	32602	Srv (3,25) OBSW and Bootloader Version Info						2602	
LDHK3100powe	33100	Srv (3,25) DDS Output Power Data						3100	
LMHK3200temp	33200	Srv (3,25) MICS Housekeeping raw Temperature TM						3200	
LOHK3300temp	33300	Srv (3,25) OSIRIS Temperature TM						3300	
LRHK3500pwdn	33500	Srv (3,25) PLOC Power-Board N Data						3500	
LRHK3501pwdr	33501	Srv (3,25) PLOC Power-Board R Data						3501	
LRHK3600ccnd	33600	Srv (3,25) PLOC CC N Data						3600	
LRHK3601ccrd	33601	Srv (3,25) PLOC CC R Data						3601	
LRHK3700plcn	33700	Srv (3,25) PLOC N (CC, PWR) Housekeeping						3700	
LRHK3701plcr	33701	Srv (3,25) PLOC R (CC, PWR) Housekeeping TM						3701	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

PBHK4000elec	34000	Srv (3,25) Battery Electrical Values						4000	
PPHK4200swi1	34200	Srv (3,25) PCDU Switch status 1						4200	
PPHK4201swi2	34201	Srv (3,25) PCDU switch status 2						4201	
PPHK4202fuse	34202	Srv (3,25) PCDU Fuse status						4202	
PPHK4203fucu	34203	Srv (3,25) Fuse and Bus Currents, Bus Voltage measured by the PCDU						4203	
PPHK4206rsts	34206	Srv (3,25) PCDU raw Temperature Values						4206	
PPHK4207fupw	34207	Srv (3,25) Fuse Power Values						4207	
PPHK4210swst	34210	Srv (3,25) PCDU Switch States I						4210	
PPHK4211swfu	34211	Srv (3,25) PCDU Switch States II and Fuse States						4211	
PSHK4800elec	34800	Srv (3,25) Solar Panels Electrical Values Measured by the PCDU						4800	
PYHK4900elec	34900	Srv (3,25) PSS electrical values (currents, voltages, SoC)						4900	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

PYHK4901powe	34901	Srv (3,25) PSS Calculated Power Values						4901	
TSHK5200tbus	35200	Srv (3,25) TCS calibr Bus Measured Temperature Values						5200	
TSHK5201payl	35201	Srv (3,25) TCS calibr Payload Temperature Values						5201	
TYHK5800temp	35800	Srv (3,25) Component Temperatures Best Estimates						5800	
TYHK5900tast	35900	Srv (3,25) TCS Component Temperature Target States						5900	
TYHK5901cust	35901	Srv (3,25) TCS Component Temperature Current States						5901	
TYHK5902herq	35902	Srv (3,25) TCS Components Heater Request Status						5902	
XNHK6000data	36000	Srv (3,25) TTC N Housekeeping data						6000	
XRHK6200data	36200	Srv (3,25) TTC R Housekeeping Data						6200	
XYHK6900data	36900	Srv (3,25) TTC System Housekeeping Data						6900	
YTHK7700time	37700	Srv (3,25) Basic Time Information						7700	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

Memory TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
DMMC2200mech	2200	Srv (6,10) Memory Check Report	53	6	10	0	0	−1	DMC00002
DMMD2201dump	2201	Srv (6,6) Memory Dump TM packet, variable length	53	6	6	0	0	2201	DMC00001
DMSI2210ssdr	2210	Srv (15,6) TM Storage Selection Definition Report	53	15	6	0	0	2210	DMC10050
DMSI2211hkca	2211	Srv (15,13) Packet Store Catalogue Report: Housekeeping	53	15	13	1342181122	0	−1	DMC10120
DMSI2212evca	2212	Srv (15,13) Packet Store Catalogue Report: Event	53	15	13	1342181123	0	−1	DMC10120
DMSI2213mica	2213	Srv (15,13) Packet Store Catalogue Report: Miscellaneous	53	15	13	1342181124	0	−1	DMC10120
OBC TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
DOFC2400prom	82400	Srv (8,129) Function TM: OBC Core PROM State	53	8	129	1128464384	3	−1	DOC00003
DOFC2401cpuu	82401	Srv (8,129) Function TM: CPU Usage Statistics	53	8	129	1128464384	11	82401	DOC00010
DOFC2402secc	82402	Srv (8,129) Function TM: Secondary Image Boot Counter	53	8	129	1128464384	21	−1	DOC00021

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

CCSDS-Board TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
DCHK2000data	32000	Srv (3,25) CCSDS Board Housekeeping Data	53	3	25	2000	0	−1	
OBSW TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
DSAK2601acsu	12601	Srv (1,1) Acceptance Success	53	1	1	0	0	−1	
DSAF2602acfl	12602	Srv (1,2) Acceptance Failure	53	1	2	0	0	−1	
DSAK2603stsu	12603	Srv (1,3) Execution Started Success	53	1	3	0	0	−1	
DSAF2604stfl	12604	Srv (1,4) Execution Started Failure	53	1	4	0	0	−1	
DSAK2605prsu	12605	Srv (1,5) Execution Progress Success	53	1	5	0	0	−1	
DSAF2606prfl	12606	Srv (1,6) Execution Progress Failure	53	1	6	0	0	−1	
DSAK2607cpsu	12607	Srv (1,7) Execution Completed Success	53	1	7	0	0	−1	
DSAF2608cpfl	12608	Srv (1,8) Execution Completed Failure	53	1	8	0	0	−1	
DSRD2700rwtc	22700	Srv (2,134) TTC_RX_N_HANDL wiretapping TC	53	2	134	1140852224	0	−1	
DSRD2701rwtc	22701	Srv (2,134) TTC_RX_R_HANDL wiretapping TC	53	2	134	1140852480	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

DSRD2702rwtc	22702	Srv (2,134) TTC_TX_N_HANDL wiretapping TC	53	2	134	1140852736	0	−1	
DSRD2703rwtc	22703	Srv (2,134) TTC_TX_R_HANDL wiretapping TC	53	2	134	1140852992	0	−1	
DSRD2704rwtc	22704	Srv (2,134) STR_HANDLER wiretapping TC	53	2	134	1140854784	0	−1	
DSRD2705rwtc	22705	Srv (2,134) FOG_HANDLER wiretapping TC	53	2	134	1140855808	0	−1	
DSRD2706rwtc	22706	Srv (2,134) RW_HANDLER_0 wiretapping TC	53	2	134	1140856064	0	−1	
DSRD2707rwtc	22707	Srv (2,134) RW_HANDLER_1 wiretapping TC	53	2	134	1140856320	0	−1	
DSRD2708rwtc	22708	Srv (2,134) RW_HANDLER_2 wiretapping TC	53	2	134	1140856576	0	−1	
DSRD2709rwtc	22709	Srv (2,134) RW_HANDLER_3 wiretapping TC	53	2	134	1140856832	0	−1	
DSRD2710rwtc	22710	Srv (2,134) MGM_HANDLER_N wiretapping TC	53	2	134	1140857344	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

DSRD2711rwtc	22711	Srv (2,134) MGM_HANDLER_R wiretapping TC	53	2	134	1140857600	0	−1	
DSRD2712rwtc	22712	Srv (2,134) MGT_HANDLER_N wiretapping TC	53	2	134	1140857856	0	−1	
DSRD2713rwtc	22713	Srv (2,134) MGT_N_COIL_X wiretapping TC	53	2	134	1140857857	0	−1	
DSRD2714rwtc	22714	Srv (2,134) MGT_N_COIL_Y wiretapping TC	53	2	134	1140857858	0	−1	
DSRD2715rwtc	22715	Srv (2,134) MGT_N_COIL_Z wiretapping TC	53	2	134	1140857859	0	−1	
DSRD2716rwtc	22716	Srv (2,134) MGT_HANDLER_R wiretapping TC	53	2	134	1140858112	0	−1	
DSRD2717rwtc	22717	Srv (2,134) MGT_R_COIL_X wiretapping TC	53	2	134	1140858113	0	−1	
DSRD2718rwtc	22718	Srv (2,134) MGT_R_COIL_Y wiretapping TC	53	2	134	1140858114	0	−1	
DSRD2719rwtc	22719	Srv (2,134) MGT_R_COIL_Z wiretapping TC	53	2	134	1140858115	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

DSRD2720rwtc	22720	Srv (2,134) GPS_HANDLER_0 wiretapping TC	53	2	134	1140858624	0	−1	
DSRD2721rwtc	22721	Srv (2,134) GPS_HANDLER_1 wiretapping TC	53	2	134	1140858880	0	−1	
DSRD2722rwtc	22722	Srv (2,134) GPS_HANDLER_2 wiretapping TC	53	2	134	1140859136	0	−1	
DSRD2723rwtc	22723	Srv (2,134) GPS_CDM_HANDLE wiretapping TC	53	2	134	1140859392	0	−1	
DSRD2724rwtc	22724	Srv (2,134) PCDU_HANDLER wiretapping TC	53	2	134	1140863488	0	−1	
DSRD2725rwtc	22725	Srv (2,134) CORE_BOARD_N wiretapping TC	53	2	134	1145241600	0	−1	
DSRD2726rwtc	22726	Srv (2,134) CORE_BOARD_R wiretapping TC	53	2	134	1145241856	0	−1	
DSRD2727rwtc	22727	Srv (2,134) IO_BOARD_N_HAN wiretapping TC	53	2	134	1145372672	0	−1	
DSRD2728rwtc	22728	Srv (2,134) IO_BOARD_R_HAN wiretapping TC	53	2	134	1145372928	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

DSRD2729rwtc	22729	Srv (2,134) CCSDS_0_HANDLE wiretapping TC	53	2	134	1145503744	0	−1	
DSRD2730rwtc	22730	Srv (2,134) CCSDS_1_HANDLE wiretapping TC	53	2	134	1145504000	0	−1	
DSRD2731rwtc	22731	Srv (2,134) PLOC_VTX_HAND wiretapping TC	53	2	134	1146093568	0	−1	
DSRD2732rwtc	22732	Srv (2,134) PLOC_CC_N_HAND wiretapping TC	53	2	134	1146094080	0	−1	
DSRD2733rwtc	22733	Srv (2,134) PLOC_CC_R_HAND wiretapping TC	53	2	134	1146094336	0	−1	
DSRD2734rwtc	22734	Srv (2,134) PLOC_POWER_N wiretapping TC	53	2	134	1146094592	0	−1	
DSRD2735rwtc	22735	Srv (2,134) PLOC_POWER_R wiretapping TC	53	2	134	1146094848	0	−1	
DSRD2736rwtc	22736	Srv (2,134) PLOC_VTX_ROUTR wiretapping TC	53	2	134	1146095104	0	−1	
DSRD2737rwtc	22737	Srv (2,134) DDS_HANDLER_N wiretapping TC	53	2	134	1146095872	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

DSRD2738rwtc	22738	Srv (2,134) DDS_HANDLER_R wiretapping TC	53	2	134	1146096128	0	−1	
DSRD2739rwtc	22739	Srv (2,134) OSIRIS_HANDLER wiretapping TC	53	2	134	1146096384	0	−1	
DSRD2740rwtc	22740	Srv (2,134) DDS_ROUT_HAN_N wiretapping TC	53	2	134	1146096640	0	−1	
DSRD2741rwtc	22741	Srv (2,134) DDS_ROUT_HAN_R wiretapping TC	53	2	134	1146096896	0	−1	
DSRD2742rwtc	22742	Srv (2,134) AIS_HANDLER wiretapping TC	53	2	134	1146097152	0	−1	
DSRD2743rwtc	22743	Srv (2,134) AIS_ANTENNA wiretapping TC	53	2	134	1146097153	0	−1	
DSRD2744rwtc	22744	Srv (2,134) PAMCAM_HANDLER wiretapping TC	53	2	134	1146097408	0	−1	
DSRD2745rwtc	22745	Srv (2,134) MICS_HANDLER wiretapping TC	53	2	134	1146097664	0	−1	
DSRD2746rwtc	22746	Srv (2,134) MICS_GREEN wiretapping TC	53	2	134	1146097665	0	−1	
DSRD2747rwtc	22747	Srv (2,134) MICS_RED wiretapping TC	53	2	134	1146097666	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

DSRD2748rwtc	22748	Srv (2,134) MICS_NIR wiretapping TC	53	2	134	1146097667	0	−1	
DSHK2600drep	32600	Srv (3,10) HK Parameter Report Definitions Report	53	3	10	0	0	32600	DSC00009
DSDG2601drep	32601	Srv (3,12) DG Parameter Report Definitions Report	53	3	12	0	0	32601	DSC00011
DSHK2602swvr	32602	Srv (3,25) OBSW and Bootloader Version Info	53	3	25	2602	0	−1	
DSMO2600oolr	42600	Srv (12,12) OOL Check Transition Report	53	12	12	0	0	42600	
Power Subsystem TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
PYMD4900mdcp	4900	Srv (6,136) PSS Memory Dump: Max Delta Battery Capacity	53	6	136	1129316352	0	−1	PYC61001
PYMD4901pudi	4901	Srv (6,136) PSS Memory Dump: PCDU Capacity Update Interval	53	6	136	1129316352	1	−1	PYC61002
PYMD4902pade	4902	Srv (6,136) PSS Memory Dump: Panels Deployed Flag	53	6	136	1129316352	2	−1	PYC61003
PYMD4903s0cc	4903	Srv (6,136) PSS Memory Dump: Batt S0 Calibration Curr Threshold	53	6	136	1129317376	0	−1	PYC61000
PYMD4904s0cv	4904	Srv (6,136) PSS Memory Dump: Batt S0 Calibration Volt Threshold	53	6	136	1129317376	1	−1	PYC61000

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

PYMD4905s0cs	4905	Srv (6,136) PSS Memory Dump: Batt S0 Calibration SoC Threshold	53	6	136	1129317376	2	−1	PYC61000
PYMD4906s0tt	4906	Srv (6,136) PSS Memory Dump: Batt S0 Temperature Threshold	53	6	136	1129317376	3	−1	PYC61000
PYMD4907s0tl	4907	Srv (6,136) PSS Memory Dump: Batt S0 Thermal Capacity Loss	53	6	136	1129317376	4	−1	PYC61000
PYMD4908s0cl	4908	Srv (6,136) PSS Memory Dump: Batt S0 Charge Loss Factor	53	6	136	1129317376	5	−1	PYC61000
PYMD4909s0cf	4909	Srv (6,136) PSS Memory Dump: Batt S0 Charge Change Factor	53	6	136	1129317376	6	−1	PYC61000
PYMD4910s0dl	4910	Srv (6,136) PSS Memory Dump: Batt S0 Discharge Loss Factor	53	6	136	1129317376	7	−1	PYC61000
PYMD4911s0df	4911	Srv (6,136) PSS Memory Dump: Batt S0 Discharge Change Factor	53	6	136	1129317376	8	−1	PYC61000
PYMD4912s0mc	4912	Srv (6,136) PSS Memory Dump: Batt S0 Maximum Capacity	53	6	136	1129317376	9	−1	PYC61000
PYMD4913s0bc	4913	Srv (6,136) PSS Memory Dump: Batt S0 BCC Current	53	6	136	1129317376	10	−1	PYC61000

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

PYMD4914s1cc	4914	Srv (6,136) PSS Memory Dump: Batt S1 Calibration Curr Threshold	53	6	136	1129317632	0	−1	PYC61000
PYMD4915s1cv	4915	Srv (6,136) PSS Memory Dump: Batt S1 Calibration Volt Threshold	53	6	136	1129317632	1	−1	PYC61000
PYMD4916s1cs	4916	Srv (6,136) PSS Memory Dump: Batt S1 Calibration SoC Threshold	53	6	136	1129317632	2	−1	PYC61000
PYMD4917s1tt	4917	Srv (6,136) PSS Memory Dump: Batt S1 Temperature Threshold	53	6	136	1129317632	3	−1	PYC61000
PYMD4918s1tl	4918	Srv (6,136) PSS Memory Dump: Batt S1 Thermal Capacity Loss	53	6	136	1129317632	4	−1	PYC61000
PYMD4919s1cl	4919	Srv (6,136) PSS Memory Dump: Batt S1 Charge Loss Factor	53	6	136	1129317632	5	−1	PYC61000
PYMD4920s1cf	4920	Srv (6,136) PSS Memory Dump: Batt S1 Charge Change Factor	53	6	136	1129317632	6	−1	PYC61000
PYMD4921s1dl	4921	Srv (6,136) PSS Memory Dump: Batt S1 Discharge Loss Factor	53	6	136	1129317632	7	−1	PYC61000
PYMD4922s1df	4922	Srv (6,136) PSS Memory Dump: Batt S1 Discharge Change Factor	53	6	136	1129317632	8	−1	PYC61000

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

PYMD4923s1mc	4923	Srv (6,136) PSS Memory Dump: Batt S1 Maximum Capacity	53	6	136	1129317632	9	−1	PYC61000
PYMD4924s1bc	4924	Srv (6,136) PSS Memory Dump: Batt S1 BCC Current	53	6	136	1129317632	10	−1	PYC61000
PYMD4925s2cc	4925	Srv (6,136) PSS Memory Dump: Batt S2 Calibration Curr Threshold	53	6	136	1129317888	0	−1	PYC61000
PYMD4926s2cv	4926	Srv (6,136) PSS Memory Dump: Batt S2 Calibration Volt Threshold	53	6	136	1129317888	1	−1	PYC61000
PYMD4927s2cs	4927	Srv (6,136) PSS Memory Dump: Batt S2 Calibration SoC Threshold	53	6	136	1129317888	2	−1	PYC61000
PYMD4928s2tt	4928	Srv (6,136) PSS Memory Dump: Batt S2 Temperature Threshold	53	6	136	1129317888	3	−1	PYC61000
PYMD4929s2tl	4929	Srv (6,136) PSS Memory Dump: Batt S2 Thermal Capacity Loss	53	6	136	1129317888	4	−1	PYC61000
PYMD4930s2cl	4930	Srv (6,136) PSS Memory Dump: Batt S2 Charge Loss Factor	53	6	136	1129317888	5	−1	PYC61000
PYMD4931s2cf	4931	Srv (6,136) PSS Memory Dump: Batt S2 Charge Change Factor	53	6	136	1129317888	6	−1	PYC61000
PYMD4932s2dl	4932	Srv (6,136) PSS Memory Dump: Batt S2 Discharge Loss Factor	53	6	136	1129317888	7	−1	PYC61000

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

PYMD4933s2df	4933	Srv (6,136) PSS Memory Dump: Batt S2 Discharge Change Factor	53	6	136	1129317888	8	−1	PYC61000
PYMD4934s2mc	4934	Srv (6,136) PSS Memory Dump: Batt S2 Maximum Capacity	53	6	136	1129317888	9	−1	PYC61000
PYMD4935s2bc	4935	Srv (6,136) PSS Memory Dump: Batt S2 BCC Current	53	6	136	1129317888	10	−1	PYC61000
PYHK4900elec	34900	Srv (3,25) PSS electrical values (currents, voltages, SoC)	53	3	25	4900	0	−1	
PYHK4901powe	34901	Srv (3,25) PSS Calculated Power Values	53	3	25	4901	0	−1	
PCDU TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
PPDT4202depl	24202	Srv (2,135) PCDU Direct TM: Solar Panel Deployment Status	53	2	135	1140863488	8	−1	PPC00800
PPDT4203domt	24203	Srv (2,135) PCDU Direct TM: DOM Timer	53	2	135	1140863488	58	−1	PPC03A00
PPDT4204bcrs	24204	Srv (2,135) PCDU Direct TM: BCR Status	53	2	135	1140863488	10	−1	PPC00A00
PPDT4205mode	24205	Srv (2,135) PCDU Direct TM: PCDU Device Mode	53	2	135	1140863488	20	−1	PPC01400
PPDT4206bist	24206	Srv (2,135) PCDU Direct TM: Bi-stable Relay Status	53	2	135	1140863488	23	−1	PPC01700

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

PPDT4207ocpu	24207	Srv (2,135) PCDU Direct TM: Operating CPU	53	2	135	1140863488	29	−1	PPC01D00
PPDT4208fucu	24208	Srv (2,135) PCDU Direct TM: Fuse Currents	53	2	135	1140863488	35	−1	PPC02300
PPDT4210lapa	24210	Srv (2,135) PCDU Direct TM: Large TM packet	53	2	135	1140863488	53	−1	PPC03500
PPDT4211socp	24211	Srv (2,135) PCDU Direct TM: Battery SoC Estimation Parameters	53	2	135	1140863488	61	−1	PPC03D00
PPDT4212bfcp	24212	Srv (2,135) PCDU Direct TM: Battery Strings Full Capacity Values	53	2	135	1140863488	63	−1	PPC03F00
PPDT4220bcp0	24220	Srv (2,135) PCDU Direct TM: Battery Capacity String 0	53	2	135	1140863488	2	−1	PPC00200
PPDT4221bcp1	24221	Srv (2,135) PCDU Direct TM: Battery Capacity String 1	53	2	135	1140863488	258	−1	PPC00201
PPDT4222bcp2	24222	Srv (2,135) PCDU Direct TM: Battery Capacity String 2	53	2	135	1140863488	514	−1	PPC00202
PPDT4223bcpc	24223	Srv (2,135) PCDU Direct TM: Battery Capacity combined	53	2	135	1140863488	65282	−1	PPC002FF
PPDT4224bsc0	24224	Srv (2,135) PCDU Direct TM: Battery SoC String 0	53	2	135	1140863488	3	−1	PPC00300
PPDT4225bsc1	24225	Srv (2,135) PCDU Direct TM: Battery SoC String 1	53	2	135	1140863488	259	−1	PPC00301

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

PPDT4226bsc2	24226	Srv (2,135) PCDU Direct TM: Battery SoC String 2	53	2	135	1140863488	515	−1	PPC00302
PPDT4227bscc	24227	Srv (2,135) PCDU Direct TM: Battery SoC combined	53	2	135	1140863488	65283	−1	PPC003FF
PPDT4230chvo	24230	Srv (2,135) PCDU Direct TM: Battery Charging Voltage	53	2	135	1140863488	7	−1	PPC00700
PPDT4231bcaf	24231	Srv (2,135) PCDU Direct TM: Battery charge activation flag	53	2	135	1140863488	55	−1	PPC03700
PPDT4232badt	24232	Srv (2,135) PCDU Direct TM: Battery Deactivation Time	53	2	135	1140863488	57	−1	PPC03900
PPDT4233inht	24233	Srv (2,135) PCDU Direct TM: Initial Holding Time	53	2	135	1140863488	12	−1	PPC00C00
PPDT4234htib	24234	Srv (2,135) PCDU Direct TM: Holding Time in Between Recovery	53	2	135	1140863488	14	−1	PPC00E00
PPDT4235htrm	24235	Srv (2,135) PCDU Direct TM: Holding Time between RM1 and MOM	53	2	135	1140863488	27	−1	PPC01B00
PPDT4236strr	24236	Srv (2,135) PCDU Direct TM: SoC for Transition from R0 to R1	53	2	135	1140863488	18	−1	PPC01200
PPDT4237cptt	24237	Srv (2,135) PCDU Direct TM: CPU toggle time	53	2	135	1140863488	25	−1	PPC01900

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

PPDT4238cpcu	24238	Srv (2,135) PCDU Direct TM: Check Maximum Component Current	53	2	135	1140863488	288	−1	PPC02001
PPDT4239actf	24239	Srv (2,135) PCDU Direct TM: Deployment Activation Flag	53	2	135	1140863488	303	−1	PPC02F03
PPDT4240dpl0	24240	Srv (2,135) PCDU Direct TM: Deployment Timer 0	53	2	135	1140863488	52	−1	PPC03400
PPDT4241dpl1	24241	Srv (2,135) PCDU Direct TM: Deployment Timer 1	53	2	135	1140863488	308	−1	PPC03401
PPDT4242test	24242	Srv (2,135) PCDU Direct TM: Test String Measurement TM	53	2	135	1140863488	42	−1	PPC02A00
PPDT4280rlog	24280	Srv (2,135) PCDU Direct TM: RAM Log Data	53	2	135	1140863488	44	24280	PPC02C00
PPDT4281flog	24281	Srv (2,135) PCDU Direct TM: FLASH Log Data	53	2	135	1140863488	48	24281	PPC03000
PPDT4282wlog	24282	Srv (2,135) PCDU Direct TM: Working History Log Data	53	2	135	1140863488	50	24282	PPC03200
PPHK4200swi1	34200	Srv (3,25) PCDU Switch status 1	53	3	25	4200	0	−1	
PPHK4201swi2	34201	Srv (3,25) PCDU switch status 2	53	3	25	4201	0	−1	
PPHK4202fuse	34202	Srv (3,25) PCDU Fuse status	53	3	25	4202	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

PPHK4203fucu	34203	Srv (3,25) Fuse and Bus Currents, Bus Voltag Measured by the PCD	53	3	25	4203	0	−1	
PPHK4206rsts	34206	Srv (3,25) PCDU raw Temperature Values	53	3	25	4206	0	−1	
PPHK4207fupw	34207	Srv (3,25) Fuse Power Values	53	3	25	4207	0	−1	
PPHK4210swst	34210	Srv (3,25) PCDU Switch States I	53	3	25	4210	0	−1	
PPHK4211swfu	34211	Srv (3,25) PCDU Switch States II and Fuse States	53	3	25	4211	0	−1	
Battery TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
PBHK4000elec	34000	Srv (3,25) Battery Electrical Values	53	3	25	4000	0	−1	
Solar Panels TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
PSHK4800elec	34800	Srv (3,25) Solar Panels Electrical Values Measured by the PCDU	53	3	25	4800	0	−1	
PSDG4810sp1d	34810	Srv (3,26) Solar Panel Voltage and Current Diagnostic Packet	53	3	26	4810	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TTC Subsystem TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
XYHK6900data	36900	Srv (3,25) TTC System Housekeeping Data	53	3	25	6900	0	−1	
TTC-Nominal TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
XNHK6000data	36000	Srv (3,25) TTC N Housekeeping data	53	3	25	6000	0	−1	
TTC-Redundant TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
XRHK6200data	36200	Srv (3,25) TTC R Housekeeping Data	53	3	25	6200	0	−1	
Attitude Control Subsystem TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
AYHK1800chse	31800	Srv (3,25) ACS Controller Checked Sensor Values	53	3	25	1800	0	−1	
AYHK1801calc	31801	Srv (3,25) ACS System attitude and orbit calc data	53	3	25	1801	0	−1	
Sun Sensor TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
AUHK1600curr	31600	Srv (3,25) SuS Currents Measured by the PCDU	53	3	25	1600	0	−1	
Magnetometer TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
AMHK1200data	31200	Srv (3,25) MGM Data	53	3	25	1200	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

Magnetotorquer TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
AQHK1300data	31300	Srv (3,25) MGT Data	53	3	25	1300	0	−1	
Star Tracker TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
ASTE1500ping	1500	Srv (17,2) STR Test Ping Packet	1937	17	2	0	0	−1	ASC10000
ASMD1506dump	1506	Srv (6,6) STR Raw Memory Dump	1937	6	6	0	0	1506	ASC10003, ASC00007, ASC10007
ASMC1510mech	1510	Srv (6,10) STR Raw Memory Check TM	1937	6	10	0	0	−1	ASC10017
ASAK1501acsu	11501	Srv (1,1) STR Raw Acceptance Success (AscTCVerTM, success)	1937	1	1	0	0	−1	
ASAF1502acfl	11502	Srv (1,2) STR Raw Acceptance Failure (AscTCVerTM, failure)	1937	1	2	0	0	−1	
ASAK1507cpsu	11507	Srv (1,7) STR Raw Completion Success (AscTCVerTM, success)	1937	1	7	0	0	−1	
ASAF1508cpfl	11508	Srv (1,8) STR Raw Completion Failure (AscTCVerTM, failure)	1937	1	8	0	0	−1	
ASHK1500atti	31500	Srv (3,25) STR Attitude TM	53	3	25	1500	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

ASHK1501hktm	31501	Srv (3,25) STR Housekeeping TM	53	3	25	1501	0	−1	
ASHK1502miru	31502	Srv (3,25) STR MIRU TM	53	3	25	1502	0	−1	
ASTI1500time	81500	Srv (9,131) STR Raw Time TM (AscSyncTimeTM)	1937	9	131	0	0	−1	ASC10018
Fiber-optic Gyro TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
AFHK1000data	31000	Srv (3,25) FOG Rate Measurements	53	3	25	1000	0	−1	
GPS TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
AGHK1100f40m	31100	Srv (3,25) GPS0 TM found in the F40 message	53	3	25	1100	0	−1	
AGHK1101f40m	31101	Srv (3,25) GPS1 TM found in the F40 message	53	3	25	1101	0	−1	
AGHK1102f40m	31102	Srv (3,25) GPS2 TM found in the F40 message	53	3	25	1102	0	−1	
AGHK1103f620	31103	Srv (3,25) GPS0 receiver raw data found in F62 message	53	3	25	1103	0	−1	
AGHK1104f620	31104	Srv (3,25) GPS0 receiver raw data found in F62 message part 2	53	3	25	1104	0	−1	
AGHK1105f621	31105	Srv (3,25) GPS1 receiver raw data found in F62 message	53	3	25	1105	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

AGHK1106f621	31106	Srv (3,25) GPS1 receiver raw data found in F62 message part 2	53	3	25	1106	0	−1	
AGHK1107f622	31107	Srv (3,25) GPS2 receiver raw data found in F62 message	53	3	25	1107	0	−1	
AGHK1108f622	31108	Srv (3,25) GPS2 receiver raw data found in F62 message, part 2	53	3	25	1108	0	−1	
AGHK1109f80m	31109	Srv (3,25) GPS0 TM found in the F80 message: XNS filtered navig	53	3	25	1109	0	−1	
AGHK1110f80m	31110	Srv (3,25) GPS1 TM found in the F80 message: XNS filtered navig	53	3	25	1110	0	−1	
AGHK3111f80m	31111	Srv (3,25) GPS2 TM found in the F80 message: XNS filtered navig	53	3	25	1111	0	−1	
Reaction Wheel TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
ARHK1400data	31400	Srv (3,25) RW Data	53	3	25	1400	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

Thermal Control Subsystem TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
TYMD5800cnht	5800	Srv (6,136) TCS Memory Dump: Core Heater N Timeout	53	6	136	1129578609	0	−1	TYC61000
TYMD5801crht	5801	Srv (6,136) TCS Memory Dump: Core Heater R Timeout	53	6	136	1129578610	0	−1	TYC61000
TYMD5802snht	5802	Srv (6,136) TCS Memory Dump: Service Heater N Timeout	53	6	136	1129578611	0	−1	TYC61000
TYMD5803srht	5803	Srv (6,136) TCS Memory Dump: Service Heater R Timeout	53	6	136	1129578612	0	−1	TYC61000
TYMD5804pnht	5804	Srv (6,136) TCS Memory Dump: Payload Heater N Timeout	53	6	136	1129578613	0	−1	TYC61000
TYMD5805prht	5805	Srv (6,136) TCS Memory Dump: Payload Heater R Timeout	53	6	136	1129578614	0	−1	TYC61000
TYMD5806dtho	5806	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn BATTERY_S0	53	6	136	1129578496	1129317377	−1	TYC61001
TYMD5807dtho	5807	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn BATTERY_S1	53	6	136	1129578496	1129317633	−1	TYC61001
TYMD5808dtho	5808	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn BATTERY_S2	53	6	136	1129578496	1129317889	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5809dtho	5809	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn DDS_TX_N	53	6	136	1129578496	1146095873	−1	TYC61001
TYMD5810dtho	5810	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn DDS_TX_R	53	6	136	1129578496	1146096129	−1	TYC61001
TYMD5811dtho	5811	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn FOG	53	6	136	1129578496	1140855809	−1	TYC61001
TYMD5812dtho	5812	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn GPS_0	53	6	136	1129578496	1140858625	−1	TYC61001
TYMD5813dtho	5813	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn GPS_1	53	6	136	1129578496	1140858881	−1	TYC61001
TYMD5814dtho	5814	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn GPS_2	53	6	136	1129578496	1140859137	−1	TYC61001
TYMD5815dtho	5815	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn MGM_0	53	6	136	1129578496	1140857345	−1	TYC61001
TYMD5816dtho	5816	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn MGM_1	53	6	136	1129578496	1140857601	−1	TYC61001
TYMD5817dtho	5817	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn MGT_N	53	6	136	1129578496	1140857857	−1	TYC61001
TYMD5818dtho	5818	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn MGT_R	53	6	136	1129578496	1140858113	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5819dtho	5819	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn MICS	53	6	136	1129578496	1146097665	−1	TYC61001
TYMD5820dtho	5820	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn OBC_0	53	6	136	1129578496	1140851201	−1	TYC61001
TYMD5821dtho	5821	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn OBC_1	53	6	136	1129578496	1140851457	−1	TYC61001
TYMD5822dtho	5822	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn OSIRIS	53	6	136	1129578496	1146096385	−1	TYC61001
TYMD5823dtho	5823	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn PAMCAM	53	6	136	1129578496	1146097409	−1	TYC61001
TYMD5824dtho	5824	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn PCDU	53	6	136	1129578496	1140863489	−1	TYC61001
TYMD5825dtho	5825	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn PLOC_N	53	6	136	1129578496	1146094593	−1	TYC61001
TYMD5826dtho	5826	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn PLOC_R	53	6	136	1129578496	1146094849	−1	TYC61001
TYMD5827dtho	5827	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn RW_0	53	6	136	1129578496	1140856065	−1	TYC61001
TYMD5828dtho	5828	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn RW_1	53	6	136	1129578496	1140856321	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5829dtho	5829	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn RW_2	53	6	136	1129578496	1140856577	−1	TYC61001
TYMD5830dtho	5830	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn RW_3	53	6	136	1129578496	1140856833	−1	TYC61001
TYMD5831dtho	5831	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn STR	53	6	136	1129578496	1140854785	−1	TYC61001
TYMD5832dtho	5832	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn TTCN_RX	53	6	136	1129578496	1140852225	−1	TYC61001
TYMD5833dtho	5833	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn TTCN_TX	53	6	136	1129578496	1140852737	−1	TYC61001
TYMD5834dtho	5834	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn TTCR_RX	53	6	136	1129578496	1140852481	−1	TYC61001
TYMD5835dtho	5835	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn TTCR_TX	53	6	136	1129578496	1140852993	−1	TYC61001
TYMD5836dtho	5836	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOn AIS	53	6	136	1129578496	1146097153	−1	TYC61001
TYMD5837dthy	5837	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis BATTERY_S0	53	6	136	1129578496	1129317378	−1	TYC61001
TYMD5838dthy	5838	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis BATTERY_S1	53	6	136	1129578496	1129317634	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5839dthy	5839	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis BATTERY_S2	53	6	136	1129578496	1129317890	−1	TYC61001
TYMD5840dthy	5840	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis DDS_TX_N	53	6	136	1129578496	1146095874	−1	TYC61001
TYMD5841dthy	5841	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis DDS_TX_R	53	6	136	1129578496	1146096130	−1	TYC61001
TYMD5842dthy	5842	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis FOG	53	6	136	1129578496	1140855810	−1	TYC61001
TYMD5843dthy	5843	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis GPS_0	53	6	136	1129578496	1140858626	−1	TYC61001
TYMD5844dthy	5844	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis GPS_1	53	6	136	1129578496	1140858882	−1	TYC61001
TYMD5845dthy	5845	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis GPS_2	53	6	136	1129578496	1140859138	−1	TYC61001
TYMD5846dthy	5846	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis MGM_0	53	6	136	1129578496	1140857346	−1	TYC61001
TYMD5847dthy	5847	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis MGM_1	53	6	136	1129578496	1140857602	−1	TYC61001
TYMD5848dthy	5848	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis MGT_N	53	6	136	1129578496	1140857858	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5849dthy	5849	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis MGT_R	53	6	136	1129578496	1140858114	−1	TYC61001
TYMD5850dthy	5850	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis MICS	53	6	136	1129578496	1146097666	−1	TYC61001
TYMD5851dthy	5851	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis OBC_0	53	6	136	1129578496	1140851202	−1	TYC61001
TYMD5852dthy	5852	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis OBC_1	53	6	136	1129578496	1140851458	−1	TYC61001
TYMD5853dthy	5853	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis OSIRIS	53	6	136	1129578496	1146096386	−1	TYC61001
TYMD5854dthy	5854	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis PAMCAM	53	6	136	1129578496	1146097410	−1	TYC61001
TYMD5855dthy	5855	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis PCDU	53	6	136	1129578496	1140863490	−1	TYC61001
TYMD5856dthy	5856	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis PLOC_N	53	6	136	1129578496	1146094594	−1	TYC61001
TYMD5857dthy	5857	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis PLOC_R	53	6	136	1129578496	1146094850	−1	TYC61001
TYMD5858dthy	5858	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis RW_0	53	6	136	1129578496	1140856066	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5859dthy	5859	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis RW_1	53	6	136	1129578496	1140856322	−1	TYC61001
TYMD5860dthy	5860	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis RW_2	53	6	136	1129578496	1140856578	−1	TYC61001
TYMD5861dthy	5861	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis RW_3	53	6	136	1129578496	1140856834	−1	TYC61001
TYMD5862dthy	5862	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis STR	53	6	136	1129578496	1140854786	−1	TYC61001
TYMD5863dthy	5863	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis TTCN_RX	53	6	136	1129578496	1140852226	−1	TYC61001
TYMD5864dthy	5864	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis TTCN_TX	53	6	136	1129578496	1140852738	−1	TYC61001
TYMD5865dthy	5865	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis TTCR_RX	53	6	136	1129578496	1140852482	−1	TYC61001
TYMD5866dthy	5866	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis TTCR_TX	53	6	136	1129578496	1140852994	−1	TYC61001
TYMD5867dthy	5867	Srv (6,136) TCS Memory Dump: Delta Temp Hysteresis AIS	53	6	136	1129578496	1146097154	−1	TYC61001
TYMD5868dthf	5868	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff BATTERY_S0	53	6	136	1129578496	1129317379	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5869dthf	5869	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff BATTERY_S1	53	6	136	1129578496	1129317635	−1	TYC61001
TYMD5870dthf	5870	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff BATTERY_S2	53	6	136	1129578496	1129317891	−1	TYC61001
TYMD5871dthf	5871	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff DDS_TX_N	53	6	136	1129578496	1146095875	−1	TYC61001
TYMD5872dthf	5872	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff DDS_TX_R	53	6	136	1129578496	1146096131	−1	TYC61001
TYMD5873dthf	5873	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff FOG	53	6	136	1129578496	1140855811	−1	TYC61001
TYMD5874dthf	5874	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff GPS_0	53	6	136	1129578496	1140858627	−1	TYC61001
TYMD5875dthf	5875	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff GPS_1	53	6	136	1129578496	1140858883	−1	TYC61001
TYMD5876dthf	5876	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff GPS_2	53	6	136	1129578496	1140859139	−1	TYC61001
TYMD5877dthf	5877	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff MGM_0	53	6	136	1129578496	1140857347	−1	TYC61001
TYMD5878dthf	5878	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff MGM_1	53	6	136	1129578496	1140857603	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5879dthf	5879	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff MGT_N	53	6	136	1129578496	1140857859	−1	TYC61001
TYMD5880dthf	5880	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff MGT_R	53	6	136	1129578496	1140858115	−1	TYC61001
TYMD5881dthf	5881	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff MICS	53	6	136	1129578496	1146097667	−1	TYC61001
TYMD5882dthf	5882	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff OBC_0	53	6	136	1129578496	1140851203	−1	TYC61001
TYMD5883dthf	5883	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff OBC_1	53	6	136	1129578496	1140851459	−1	TYC61001
TYMD5884dthf	5884	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff OSIRIS	53	6	136	1129578496	1146096387	−1	TYC61001
TYMD5885dthf	5885	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff PAMCAM	53	6	136	1129578496	1146097411	−1	TYC61001
TYMD5886dthf	5886	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff PCDU	53	6	136	1129578496	1140863491	−1	TYC61001
TYMD5887dthf	5887	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff PLOC_N	53	6	136	1129578496	1146094595	−1	TYC61001
TYMD5888dthf	5888	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff PLOC_R	53	6	136	1129578496	1146094851	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5889dthf	5889	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff RW_0	53	6	136	1129578496	1140856067	−1	TYC61001
TYMD5890dthf	5890	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff RW_1	53	6	136	1129578496	1140856323	−1	TYC61001
TYMD5891dthf	5891	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff RW_2	53	6	136	1129578496	1140856579	−1	TYC61001
TYMD5892dthf	5892	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff RW_3	53	6	136	1129578496	1140856835	−1	TYC61001
TYMD5893dthf	5893	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff STR	53	6	136	1129578496	1140854787	−1	TYC61001
TYMD5894dthf	5894	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff TTCN_RX	53	6	136	1129578496	1140852227	−1	TYC61001
TYMD5895dthf	5895	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff TTCN_TX	53	6	136	1129578496	1140852739	−1	TYC61001
TYMD5896dthf	5896	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff TTCR_RX	53	6	136	1129578496	1140852483	−1	TYC61001
TYMD5897dthf	5897	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff TTCR_TX	53	6	136	1129578496	1140852995	−1	TYC61001
TYMD5898dthf	5898	Srv (6,136) TCS Memory Dump: Delta Temp HeaterOff AIS	53	6	136	1129578496	1146097155	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5899miet	5899	Srv (6,136) TCS Memory Dump: Min Temp Ever BATTERY_S0	53	6	136	1129578496	1129317380	−1	TYC61001
TYMD5900miet	5900	Srv (6,136) TCS Memory Dump: Min Temp Ever BATTERY_S1	53	6	136	1129578496	1129317636	−1	TYC61001
TYMD5901miet	5901	Srv (6,136) TCS Memory Dump: Min Temp Ever BATTERY_S2	53	6	136	1129578496	1129317892	−1	TYC61001
TYMD5902miet	5902	Srv (6,136) TCS Memory Dump: Min Temp Ever DDS_TX_N	53	6	136	1129578496	1146095876	−1	TYC61001
TYMD5903miet	5903	Srv (6,136) TCS Memory Dump: Min Temp Ever DDS_TX_R	53	6	136	1129578496	1146096132	−1	TYC61001
TYMD5904miet	5904	Srv (6,136) TCS Memory Dump: Min Temp Ever FOG	53	6	136	1129578496	1140855812	−1	TYC61001
TYMD5905miet	5905	Srv (6,136) TCS Memory Dump: Min Temp Ever GPS_0	53	6	136	1129578496	1140858628	−1	TYC61001
TYMD5906miet	5906	Srv (6,136) TCS Memory Dump: Min Temp Ever GPS_1	53	6	136	1129578496	1140858884	−1	TYC61001
TYMD5907miet	5907	Srv (6,136) TCS Memory Dump: Min Temp Ever GPS_2	53	6	136	1129578496	1140859140	−1	TYC61001
TYMD5908miet	5908	Srv (6,136) TCS Memory Dump: Min Temp Ever MGM_0	53	6	136	1129578496	1140857348	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5909miet	5909	Srv (6,136) TCS Memory Dump: Min Temp Ever MGM_1	53	6	136	1129578496	1140857604	−1	TYC61001
TYMD5910miet	5910	Srv (6,136) TCS Memory Dump: Min Temp Ever MGT_N	53	6	136	1129578496	1140857860	−1	TYC61001
TYMD5911miet	5911	Srv (6,136) TCS Memory Dump: Min Temp Ever MGT_R	53	6	136	1129578496	1140858116	−1	TYC61001
TYMD5912miet	5912	Srv (6,136) TCS Memory Dump: Min Temp Ever MICS	53	6	136	1129578496	1146097668	−1	TYC61001
TYMD5913miet	5913	Srv (6,136) TCS Memory Dump: Min Temp Ever OBC_0	53	6	136	1129578496	1140851204	−1	TYC61001
TYMD5914miet	5914	Srv (6,136) TCS Memory Dump: Min Temp Ever OBC_1	53	6	136	1129578496	1140851460	−1	TYC61001
TYMD5915miet	5915	Srv (6,136) TCS Memory Dump: Min Temp Ever OSIRIS	53	6	136	1129578496	1146096388	−1	TYC61001
TYMD5916miet	5916	Srv (6,136) TCS Memory Dump: Min Temp Ever PAMCAM	53	6	136	1129578496	1146097412	−1	TYC61001
TYMD5917miet	5917	Srv (6,136) TCS Memory Dump: Min Temp Ever PCDU	53	6	136	1129578496	1140863492	−1	TYC61001
TYMD5918miet	5918	Srv (6,136) TCS Memory Dump: Min Temp Ever PLOC_N	53	6	136	1129578496	1146094596	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5919miet	5919	Srv (6,136) TCS Memory Dump: Min Temp Ever PLOC_R	53	6	136	1129578496	1146094852	−1	TYC61001
TYMD5920miet	5920	Srv (6,136) TCS Memory Dump: Min Temp Ever RW_0	53	6	136	1129578496	1140856068	−1	TYC61001
TYMD5921miet	5921	Srv (6,136) TCS Memory Dump: Min Temp Ever RW_1	53	6	136	1129578496	1140856324	−1	TYC61001
TYMD5922miet	5922	Srv (6,136) TCS Memory Dump: Min Temp Ever RW_2	53	6	136	1129578496	1140856580	−1	TYC61001
TYMD5923miet	5923	Srv (6,136) TCS Memory Dump: Min Temp Ever RW_3	53	6	136	1129578496	1140856836	−1	TYC61001
TYMD5924miet	5924	Srv (6,136) TCS Memory Dump: Min Temp Ever STR	53	6	136	1129578496	1140854788	−1	TYC61001
TYMD5925miet	5925	Srv (6,136) TCS Memory Dump: Min Temp Ever TTCN_RX	53	6	136	1129578496	1140852228	−1	TYC61001
TYMD5926miet	5926	Srv (6,136) TCS Memory Dump: Min Temp Ever TTCN_TX	53	6	136	1129578496	1140852740	−1	TYC61001
TYMD5927miet	5927	Srv (6,136) TCS Memory Dump: Min Temp Ever TTCR_RX	53	6	136	1129578496	1140852484	−1	TYC61001
TYMD5928miet	5928	Srv (6,136) TCS Memory Dump: Min Temp Ever TTCR_TX	53	6	136	1129578496	1140852996	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5929miet	5929	Srv (6,136) TCS Memory Dump: Min Temp Ever AIS	53	6	136	1129578496	1146097156	−1	TYC61001
TYMD5930maet	5930	Srv (6,136) TCS Memory Dump: Max Temp Ever BATTERY_S0	53	6	136	1129578496	1129317381	−1	TYC61001
TYMD5931maet	5931	Srv (6,136) TCS Memory Dump: Max Temp Ever BATTERY_S1	53	6	136	1129578496	1129317637	−1	TYC61001
TYMD5932maet	5932	Srv (6,136) TCS Memory Dump: Max Temp Ever BATTERY_S2	53	6	136	1129578496	1129317893	−1	TYC61001
TYMD5933maet	5933	Srv (6,136) TCS Memory Dump: Max Temp Ever DDS_TX_N	53	6	136	1129578496	1146095877	−1	TYC61001
TYMD5934maet	5934	Srv (6,136) TCS Memory Dump: Max Temp Ever DDS_TX_R	53	6	136	1129578496	1146096133	−1	TYC61001
TYMD5935maet	5935	Srv (6,136) TCS Memory Dump: Max Temp Ever FOG	53	6	136	1129578496	1140855813	−1	TYC61001
TYMD5936maet	5936	Srv (6,136) TCS Memory Dump: Max Temp Ever GPS_0	53	6	136	1129578496	1140858629	−1	TYC61001
TYMD5937maet	5937	Srv (6,136) TCS Memory Dump: Max Temp Ever GPS_1	53	6	136	1129578496	1140858885	−1	TYC61001
TYMD5938maet	5938	Srv (6,136) TCS Memory Dump: Max Temp Ever GPS_2	53	6	136	1129578496	1140859141	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5939maet	5939	Srv (6,136) TCS Memory Dump: Max Temp Ever MGM_0	53	6	136	1129578496	1140857349	−1	TYC61001
TYMD5940maet	5940	Srv (6,136) TCS Memory Dump: Max Temp Ever MGM_1	53	6	136	1129578496	1140857605	−1	TYC61001
TYMD5941maet	5941	Srv (6,136) TCS Memory Dump: Max Temp Ever MGT_N	53	6	136	1129578496	1140857861	−1	TYC61001
TYMD5942maet	5942	Srv (6,136) TCS Memory Dump: Max Temp Ever MGT_R	53	6	136	1129578496	1140858117	−1	TYC61001
TYMD5943maet	5943	Srv (6,136) TCS Memory Dump: Max Temp Ever MICS	53	6	136	1129578496	1146097669	−1	TYC61001
TYMD5944maet	5944	Srv (6,136) TCS Memory Dump: Max Temp Ever OBC_0	53	6	136	1129578496	1140851205	−1	TYC61001
TYMD5945maet	5945	Srv (6,136) TCS Memory Dump: Max Temp Ever OBC_1	53	6	136	1129578496	1140851461	−1	TYC61001
TYMD5946maet	5946	Srv (6,136) TCS Memory Dump: Max Temp Ever OSIRIS	53	6	136	1129578496	1146096389	−1	TYC61001
TYMD5947maet	5947	Srv (6,136) TCS Memory Dump: Max Temp Ever PAMCAM	53	6	136	1129578496	1146097413	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5948maet	5948	Srv (6,136) TCS Memory Dump: Max Temp Ever PCDU	53	6	136	1129578496	1140863493	−1	TYC61001
TYMD5949maet	5949	Srv (6,136) TCS Memory Dump: Max Temp Ever PLOC_N	53	6	136	1129578496	1146094597	−1	TYC61001
TYMD5950maet	5950	Srv (6,136) TCS Memory Dump: Max Temp Ever PLOC_R	53	6	136	1129578496	1146094853	−1	TYC61001
TYMD5951maet	5951	Srv (6,136) TCS Memory Dump: Max Temp Ever RW_0	53	6	136	1129578496	1140856069	−1	TYC61001
TYMD5952maet	5952	Srv (6,136) TCS Memory Dump: Max Temp Ever RW_1	53	6	136	1129578496	1140856325	−1	TYC61001
TYMD5953maet	5953	Srv (6,136) TCS Memory Dump: Max Temp Ever RW_2	53	6	136	1129578496	1140856581	−1	TYC61001
TYMD5954maet	5954	Srv (6,136) TCS Memory Dump: Max Temp Ever RW_3	53	6	136	1129578496	1140856837	−1	TYC61001
TYMD5955maet	5955	Srv (6,136) TCS Memory Dump: Max Temp Ever STR	53	6	136	1129578496	1140854789	−1	TYC61001
TYMD5956maet	5956	Srv (6,136) TCS Memory Dump: Max Temp Ever TTCN_RX	53	6	136	1129578496	1140852229	−1	TYC61001

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

TYMD5957maet	5957	Srv (6,136) TCS Memory Dump: Max Temp Ever TTCN_TX	53	6	136	1129578496	1140852741	−1	TYC61001
TYMD5958maet	5958	Srv (6,136) TCS Memory Dump: Max Temp Ever TTCR_RX	53	6	136	1129578496	1140852485	−1	TYC61001
TYMD5959maet	5959	Srv (6,136) TCS Memory Dump: Max Temp Ever TTCR_TX	53	6	136	1129578496	1140852997	−1	TYC61001
TYMD5960maet	5960	Srv (6,136) TCS Memory Dump: Max Temp Ever AIS	53	6	136	1129578496	1146097157	−1	TYC61001
TYHK5800temp	35800	Srv (3,25) Component Temperatures Best Estimates	53	3	25	5800	0	−1	
TYHK5900tast	35900	Srv (3,25) TCS Component Temperature Target States	53	3	25	5900	0	−1	
TYHK5901cust	35901	Srv (3,25) TCS Component Temperature Current States	53	3	25	5901	0	−1	
TYHK5902herq	35902	Srv (3,25) TCS Components Heater Request Status	53	3	25	5902	0	−1	
Temperature Sensors TM									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
TSHK5200tbus	35200	Srv (3,25) TCS calibr Bus Measured Temperature Values	53	3	25	5200	0	−1	
TSHK5201payl	35201	Srv (3,25) TCS calibr Payload Temperature Values	53	3	25	5201	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

Payload Computer TM (FLP Generation 1)									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
LRDT3700vers	23700	Srv (2,135) PLOC VTX Direct TM: Version	53	2	135	1146093568	257	−1	LRC02010
LRDT3701ram1	23701	Srv (2,135) PLOC VTX Direct TM: SSRAM 1 Information	53	2	135	1146093568	45516	−1	LRC02B0C
LRDT3702ram2	23702	Srv (2,135) PLOC VTX Direct TM: SSRAM 2 Information	53	2	135	1146093568	45772	−1	LRC02B0C
LRDT3703ram3	23703	Srv (2,135) PLOC VTX Direct TM: SSRAM 3 Information	53	2	135	1146093568	46284	−1	LRC02B0C
LRDT3704ram4	23704	Srv (2,135) PLOC VTX Direct TM: SSRAM 4 Information	53	2	135	1146093568	47308	−1	LRC02B0C
LRDT3705dats	23705	Srv (2,135) PLOC VTX Direct TM: Data Source/Sink Allocation	53	2	135	1146093568	4352	−1	LRC02110
LRDT3706dbsp	23706	Srv (2,135) PLOC VTX Direct TM: Debug Speed	53	2	135	1146093568	1025	−1	LRC02401
LRDT3707dbso	23707	Srv (2,135) PLOC VTX Direct TM: Debug Pin Source	53	2	135	1146093568	1026	−1	LRC02402
LRDT3708dbin	23708	Srv (2,135) PLOC VTX Direct TM: Debug Info	53	2	135	1146093568	1228	−1	LRC024CC

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

LRDT3709dpts	23709	Srv (2,135) PLOC VTX Direct TM: DDS Protocol Transmit State	53	2	135	1146093568	4626	−1	LRC02212
LRDT3710dpps	23710	Srv (2,135) PLOC VTX Direct TM: DDS Protoc Pseudorand Bypa State	53	2	135	1146093568	4610	−1	LRC02202
LRDT3711lopb	23711	Srv (2,135) PLOC VTX Loopback Data Echo	53	2	135	1146093568	512	−1	LRC02020
LRDT3712clst	23712	Srv (2,135) PLOC VTX MMU Clock State	53	2	135	1146093568	40978	−1	LRC02A12
LRDT3713clsp	23713	Srv (2,135) PLOC VTX MMU Clock Speed	53	2	135	1146093568	40980	−1	LRC02A14
LRDT3714rest	23714	Srv (2,135) PLOC VTX MMU Reset State	53	2	135	1146093568	40983	−1	LRC02A17
LRDT3715l1md	23715	Srv (2,135) PLOC VTX MMU Lane 1 Monitoring Data	53	2	135	1146093568	41241	−1	LRC02A19
LRDT3716l2md	23716	Srv (2,135) PLOC VTX MMU Lane 2 Monitoring Data	53	2	135	1146093568	41497	−1	LRC02A19
LRDT3717l3md	23717	Srv (2,135) PLOC VTX MMU Lane 3 Monitoring Data	53	2	135	1146093568	42009	−1	LRC02A19

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

LRDT3718l4md	23718	Srv (2,135) PLOC VTX MMU Lane 4 Monitoring Data	53	2	135	1146093568	43033	−1	LRC02A19
LRDT3719l1rd	23719	Srv (2,135) PLOC VTX MMU Lane 1 Read Data	53	2	135	1146093568	41378	−1	LRC02AA2
LRDT3720l2rd	23720	Srv (2,135) PLOC VTX MMU Lane 2 Read Data	53	2	135	1146093568	41634	−1	LRC02AA2
LRDT3721l3rd	23721	Srv (2,135) PLOC VTX MMU Lane 3 Read Data	53	2	135	1146093568	42146	−1	LRC02AA2
LRDT3722l4rd	23722	Srv (2,135) PLOC VTX MMU Lane 4 Read Data	53	2	135	1146093568	43170	−1	LRC02AA2
LRDT3723l1pl	23723	Srv (2,135) PLOC VTX MMU Lane 1 Picture List	53	2	135	1146093568	41393	−1	LRC02AB1
LRDT3724l2pl	23724	Srv (2,135) PLOC VTX MMU Lane 2 Picture List	53	2	135	1146093568	41649	−1	LRC02AB1
LRDT3725l3pl	23725	Srv (2,135) PLOC VTX MMU Lane 3 Picture List	53	2	135	1146093568	42161	−1	LRC02AB1
LRDT3726l4pl	23726	Srv (2,135) PLOC VTX MMU Lane 4 Picture List	53	2	135	1146093568	43185	−1	LRC02AB1
LRDT3727l1bb	23727	Srv (2,135) PLOC VTX MMU Lane 1 Bad Block List	53	2	135	1146093568	41394	−1	LRC02AB2
LRDT3728l2bb	23728	Srv (2,135) PLOC VTX MMU Lane 2 Bad Block List	53	2	135	1146093568	41650	−1	LRC02AB2

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

LRDT3729l3bb	23729	Srv (2,135) PLOC VTX MMU Lane 3 Bad Block List	53	2	135	1146093568	42162	−1	LRC02AB2
LRDT3730l4bb	23730	Srv (2,135) PLOC VTX MMU Lane 4 Bad Block List	53	2	135	1146093568	43186	−1	LRC02AB2
LRDT3731l1pb	23731	Srv (2,135) PLOC VTX MMU Lane 1 Picture Block List	53	2	135	1146093568	41395	−1	LRC02AB3
LRDT3732l2pb	23732	Srv (2,135) PLOC VTX MMU Lane 2 Picture Block List	53	2	135	1146093568	41651	−1	LRC02AB3
LRDT3733l3pb	23733	Srv (2,135) PLOC VTX MMU Lane 3 Picture Block List	53	2	135	1146093568	42163	−1	LRC02AB3
LRDT3734l4pb	23734	Srv (2,135) PLOC VTX MMU Lane 4 Picture Block List	53	2	135	1146093568	43187	−1	LRC02AB3
LRDT3735l1fb	23735	Srv (2,135) PLOC VTX MMU Lane 1 Full Block List	53	2	135	1146093568	41396	−1	LRC02AB4
LRDT3736l2fb	23736	Srv (2,135) PLOC VTX MMU Lane 2 Full Block List	53	2	135	1146093568	41652	−1	LRC02AB4
LRDT3737l3fb	23737	Srv (2,135) PLOC VTX MMU Lane 3 Full Block List	53	2	135	1146093568	42164	−1	LRC02AB4

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

LRDT3738l4fb	23738	Srv (2,135) PLOC VTX MMU Lane 4 Full Block List	53	2	135	1146093568	43188	−1	LRC02AB4
LRDT3739patt	23739	Srv (2,135) PLOC VTX Direct TM: Pattern Generator Info	53	2	135	1146093568	4884	−1	LRC02134
LRDT3750nver	23750	Srv (2,135) PLOC CC N Direct TM: Firmware Version	53	2	135	1146094080	104	−1	LRC06800
LRDT3751rver	23751	Srv (2,135) PLOC CC R Direct TM: Firmware Version	53	2	135	1146094336	104	−1	LRC06801
LRDT3752nreg	23752	Srv (2,135) PLOC CC N Direct TM: Register Data	53	2	135	1146094080	100	−1	LRC06400
LRDT3753rreg	23753	Srv (2,135) PLOC CC R Direct TM: Register Data	53	2	135	1146094336	100	−1	LRC06401
LRHK3500pwdn	33500	Srv (3,25) PLOC Power-Board N Data	53	3	25	3500	0	−1	
LRHK3501pwdr	33501	Srv (3,25) PLOC Power-Board R Data	53	3	25	3501	0	−1	
LRHK3600ccnd	33600	Srv (3,25) PLOC CC N Data	53	3	25	3600	0	−1	
LRHK3601ccrd	33601	Srv (3,25) PLOC CC R Data	53	3	25	3601	0	−1	
LRHK3700plcn	33700	Srv (3,25) PLOC N (CC, PWR) Housekeeping	53	3	25	3700	0	−1	
LRHK3701plcr	33701	Srv (3,25) PLOC R (CC, PWR) Housekeeping TM	53	3	25	3701	0	−1	

(continued)

Table 17.3 S/C telemetry packet definitions (continued)

Science Data Downlink System TM (FLP Generation 1)									
Name	**SPID**	**Description**	**APID**	**Type**	**Subt.**	**Id 1**	**Id 2**	**TPSD**	**TC Ref**
LDDT3100rgva	23100	Srv (2,135) DDS N Direct TM: Register Value	53	2	135	1146095872	5392214	−1	LDCGRG00
LDDT3101mid1	23101	Srv (2,135) DDS N Direct TM: Module Identification 1	53	2	135	1146095872	5065028	−1	LDCGMI00, LDCGMI10
LDDT3102mid2	23102	Srv (2,135) DDS N Direct TM: Module Identification 2	53	2	135	1146095872	21842244	−1	LDCGMI00, LDCGMI10
LDDT3103mid3	23103	Srv (2,135) DDS N Direct TM: Module Identification 3	53	2	135	1146095872	38619460	−1	LDCGMI00, LDCGMI10
LDDT3106dcg3	23106	Srv (2,135) DDS N Direct TM: Default Configuration 3	53	2	135	1146095872	38028102	−1	LDCGDC00
LDDT3150rgva	23150	Srv (2,135) DDS R Direct TM: Register Value	53	2	135	1146096128	5392214	−1	LDCGRG01
LDDT3151mid1	23151	Srv (2,135) DDS R Direct TM: Module Identification 1	53	2	135	1146096128	5065028	−1	LDCGMI01, LDCGMI11
LDDT3152mid2	23152	Srv (2,135) DDS R Direct TM: Module Identification 2	53	2	135	1146096128	21842244	−1	LDCGMI01, LDCGMI11
LDDT3153mid3	23153	Srv (2,135) DDS R Direct TM: Module Identification 3	53	2	135	1146096128	38619460	−1	LDCGMI01, LDCGMI11
LDDT3156dcg3	23156	Srv (2,135) DDS R Direct TM: Default Configuration 3	53	2	135	1146096128	38028102	−1	LDCGDC01
LDHK3100powe	33100	Srv (3,25) DDS Output Power Data	53	3	25	3100	0	−1	

Table 17.4 S/C Event packet definitions

Event Name	Event ID	Description	Source Object ID	Object ID [hex]	Sev.	Parameter 1	Parameter 2
FOG_TEMP_WARNING	1000	FOG Temperature Warning triggered			L	0	0
FOG_ACL_ERROR	1001	FOG ACL Error triggered			L	0	0
FOG_HARDWARE_ERROR	1002	FOG Hardware Error triggered			L	0	0
FOG_OVERFLOW_ERROR	1003	FOG Overflow Error triggered			L	0	0
GPS_FIX	1100	GPS has achieved Fix			I	0	0
GPS_LOST_FIX	1101	GPS has lost Fix			I	0	0
RW_RESET	1400	A RW handler has detected that a RW has reset itself			I		
RW_SATURATION	1401	RW went into saturation			L	0	0
IO_HEALTHY_AGAIN	2100	Both Io Boards are faulty, so one will be set to healthy again	1090584576	41010000	I	Obj ID of the board set to healthy	0
RMAP_SWITCHED_PORT	2101	RMAP Switched Port			I	New Port Nr	Old Port Nr
IO_BOARD_SWITCHED	2102	Separate event indicating the primary I/O-Board was switched	1090584576	41010000	I	ObjID Primary Board	Obj ID Secondary Board
STORE_SEND_WRITE_FAILED	2200	TM Store: Initiating sending data to store failed			L	Failure Code	Debug Info
STORE_WRITE_FAILED	2201	TM Store: Data was sent, but writing failed			L	Failure Code	0

(continued)

Table 17.4 S/C Event packet definitions (continued)

Event Name	Event ID	Description	Source Object ID	Object ID [hex]	Sev.	Parameter 1	Parameter 2
STORE_SEND_READ_FAILED	2202	TM Store: Initiating reading data from store failed.			L	Failure Code	0
STORE_READ_FAILED	2203	TM Store: Data was requested, but access failed			L	Failure Code	0
UNEXPECTED_MSG	2204	TM Store: An unexpected TM packet or data message occurred			I	0	Debug Info
STORING_FAILED	2205	TM Store: Storing data failed. May simply be a full store			L	Failure Code	SSC of failed packet
TM_DUMP_FAILED	2206	TM Store: Dumping retrieved data failed			L	Failure Code	SSC of failed packet
STORE_INIT_FAILED	2207	TM Store: Corrupted init data or read error			L	Failure Code	Debug Info
STORE_INIT_EMPTY	2208	TM Store: Store was not initialized. Starts empty			I	0	0
STORE_CONTENT_CORRUPTED	2209	TM Store: Data was read out, but it is inconsistent			L	0	Debug Info
STORE_INITIALIZE	2210	Event indicating the store will be initialized, either at boot or after IOB switch			I	0	0
HARD_LIMIT_VIOLATION	2600	A controller detected a limit violation			L		
DEVICE_BUILDING_COMMAND_FAILED	2800	Building of a Device Command Failed			L	Failure Code	CommandId

(continued)

Table 17.4 S/C Event packet definitions (continued)

Event Name	Event ID	Description	Source Object ID	Object ID [hex]	Sev.	Parameter 1	Parameter 2
DEVICE_SENDING_COMMAND_FAILED	2801	Sending of a Device Command Failed			L	Failure Code	CommandId
DEVICE_REQUESTING_REPLY_FAILED	2802	Requesting of a Device Reply Failed			L	Failure Code	0
DEVICE_READING_REPLY_FAILED	2803	Reading of a Device Reply failed			L	Failure Code	Data Length
DEVICE_INTERPRETING_REPLY_FAILED	2804	Interpreting of a Device Reply failed			L	Failure Code	CommandId
DEVICE_MISSED_REPLY	2805	Missed Device Reply			L	CommandId	0
DEVICE_UNKNOWN_REPLY	2806	Received unknown reply from device			L	CommandId	0
DEVICE_UNREQUESTED_REPLY	2807	Received Unrequested Reply from Device			L	CommandId	0
DDS_SHUTDOWN_TIMER	3100	A DDS was switched off automatically by the timer in the PCDU			I	0	0
SOC_CALIBRATION	4000	Battery string SoC value was calibrated			I	0	0
CAP_DELTA_VIOLATION	4011	Capacity calculated by PCDU and PSS differs more than expected	1129316352	43500000	L	0	0
SOC_WARNING	4012	The Battery SoC has fallen under the Warning Limit	1129316352	43500000	L	0	0
SOC_CRITICAL	4013	The Battery SoC has fallen under the Critical Limit	1129316352	43500000	L	0	0

(continued)

Table 17.4 S/C Event packet definitions (continued)

Event Name	Event ID	Description	Source Object ID	Object ID [hex]	Sev.	Parameter 1	Parameter 2
SOC_SHUTDOWN	4014	The Battery SoC has fallen under the Shutdown Limit	1129316352	43500000	L	0	0
SOC_HIGH	4015	The Battery SoC is above 100 %. Should never happen	1129316352	43500000	L	0	0
BATTERY_CURRENT_LOW	4020	The current value of the string is below a physically unrealistic limit			L	0	0
BATTERY_CURRENT_HIGH	4023	The current value of the string is above a physically unrealistic limit			L	0	0
BATTERY_VOLTAGE_LOW	4026	The voltage value of the string is below a physically unrealistic limit			L	0	0
BATTERY_VOLTAGE_HIGH	4029	The voltage value of the string is above a physically unrealistic limit			L	0	0
BATTERY_CAPACITY_TOO_HIGH	4032	Capacity is above the max. capacity, but no calibration was triggered			L	0	0
FUSE_CURRENT_LOW	4200	The value is below a physically unrealistic limit			L	0	0
FUSE_CURRENT_HIGH	4201	The value is above a physically unrealistic limit			L	0	0

(continued)

Table 17.4 S/C Event packet definitions (continued)

Event Name	Event ID	Description	Source Object ID	Object ID [hex]	Sev.	Parameter 1	Parameter 2
FUSE_WENT_OFF	4300	A fuse went off			L	0	0
PANELS_NOT_DEPLOYED	4800	PSS detected that panels are not (yet) deployed	1129316352	43500000	L	Nr of first Sensor, which is not "deployed"	0
PANELS_DEPLOYED	4801	Panels have been successfully deployed	1129316352	43500000	I	0	0
PANEL_CURRENT_LOW	4820	The current value of the panel is below a physically unrealistic limit			L	0	0
PANEL_CURRENT_HIGH	4824	The current value of the panel is above a physically unrealistic limit			L	0	0
PANEL_VOLTAGE_LOW	4828	The voltage value of the panel is below a physically unrealistic limit			L	0	0
PANEL_VOLTAGE_HIGH	4832	The voltage value of the panel is above a physically unrealistic limit			L	0	0
FUSE_WENT_OFF	4900	PSS detected a fuse that went off	1129316352	43500000	L	0	0
PSS_TIMEOUT	4902	PSS commanded something that timed out	1129316352	43500000	L	PSS Internal State	0
PSS_COMMANDING_PCDU_FAILED	4903	The PSS Controller failed to command the PCDU	1129316352	43500000	L	PCDU Function ID	Return Value

(continued)

Table 17.4 S/C Event packet definitions (continued)

Event Name	Event ID	Description	Source Object ID	Object ID [hex]	Sev.	Parameter 1	Parameter 2
BUS_CURRENT_LOW	4904	The value is below a physically unrealistic limit	1129317120	43500300	L	0	0
BUS_CURRENT_HIGH	4905	The value is above a physically unrealistic limit	1129317120	43500300	L	0	0
BUS_VOLTAGE_LOW	4906	The value is below a physically unrealistic limit	1129317120	43500300	L	0	0
BUS_VOLTAGE_HIGH	4907	The value is above a physically unrealistic limit	1129317120	43500300	L	0	0
POWER_ABOVE_HIGH_LIMIT	4910	The value is above the expected power consumption			L	0	0
POWER_BELOW_LOW_LIMIT	4911	The value is below the expected power consumption			L	0	0
HEATER_ON	5000	A Heater has been switched on			I	0	0
HEATER_OFF	5001	A Heater has been switched off			I	0	0
HEATER_TIMEOUT	5002	A Heater was on for a very long time			L	0	0
HEATER_TIMEOUT_SURVIVAL	5003	A Heater was on for a very long time in Survival Mode			L	0	0
TEMP_SENSOR_HIGH	5200	Broken wire (150Â°C will be reported)	1129578496	43540000	L	0	0

(continued)

Table 17.4 S/C Event packet definitions (continued)

Event Name	Event ID	Description	Source Object ID	Object ID [hex]	Sev.	Parameter 1	Parameter 2
TEMP_SENSOR_LOW	5201	Short circuit (−50Â°C will be reported)	1129578496	43540000	L	0	0
TEMP_SENSOR_GRADIENT	5202	Difference between two consecutive values for ONE sensor is too high	1129578496	43540000	L	0	0
BATTERY_T_LOCAL_GRADIENT_HIGH	5207	The gradient between the two sensors of one battery string is too high	1129578496	43540000	L		
TCS_CONTROL_STRATEGY	5800	The TCS Controller announces its control strategy	1129578496	43540000	I	Control Strategy (TYX10001)	0
COMPONENT_TEMP_LOW	5801	A component reports that its temperature is out of its lower soft limit			L	0	0
COMPONENT_TEMP_HIGH	5802	A component reports that its temperature is out of its higher soft limit			L	0	0
COMPONENT_TEMP_OOL_LOW	5803	A component reports that its temperature is out of its lower hard limit			L	0	0
COMPONENT_TEMP_OOL_HIGH	5804	A component reports that its temperature is out of its higher hard limit			L	0	0
DEVICE_REBOOT_INITIATED	7001	FDIR reports that is rebooted a device	1128660992	43460000	M	Object ID of rebooted device	0

(continued)

Table 17.4 S/C Event packet definitions (continued)

Event Name	Event ID	Description	Source Object ID	Object ID [hex]	Sev.	Parameter 1	Parameter 2
DEVICE_MARKED_NON_HEALTY	7002	FDIR reports that it marked an object non-healthy (faulty)	1128660992	43460000	M	Object ID of device marked faulty	0
SAFE_MODE_TRANSITION	7003	FDIR reports that is triggered a transition to system safe mode	1128660992	43460000	M	0	0
CHANGING_MODE	7400	An Object has started to change its mode			I	Target Mode	Target Submode
MODE_INFO	7401	An Object announces its mode			I	Current Mode	Current Submode
FALLBACK_FAILED	7402	An Object reports that the transition to its fallback mode failed			L	Failure Code	Failed Fallback Sequence
MODE_TRANSITION_FAILED	7403	An Object reports that a mode transition failed			L	Current Mode	Current Submode
CANT_KEEP_MODE	7404	An Object reports that it cannot keept its mode			L	Old Mode	Old Submode
HEALTH_INFO	7406	An Object announces its health status			I	Current Health	Previous Health
CHILD_CHANGED_HEALTH	7407	An Object reports that one of its children changed its health status			I	0	0
OVERWRITING_HEALTH	7409	Assembly overwrites health of children to keep satellite alive			I		

(continued)

Table 17.4 S/C Event packet definitions (continued)

Event Name	Event ID	Description	Source Object ID	Object ID [hex]	Sev.	Parameter 1	Parameter 2
OPS_SCHEDULE_TC_LOST	7601	A time-tagged TC was lost before it entered the on-board queue	1342180096	50000B00	L	Failure Code	0
OPS_SCHEDULE_SUSPENDED	7605	Scheduler received time jump event and disabled schedule	1342180096	50000B00	I	0	0
STORING_OBT_FAILED	7700	Storing the OBT (for correct time at reboot) failed	1342179584	50000900	L	Failure Code	0
TIME_WAS_SET	7701	Time Manager got a set time request and performed it	1342179584	50000900	I	POSIX seconds	POSIX usecs
TIME_JUMP_DETECTED	7702	Time Manager detected a time jump	1342179584	50000900	L	Time delta in seconds	0
BOOT	7800	triggered when the OBSW boots	0	0	I		
INITIATING_CORE_SWITCH	7801	Core Switch to the other (N/R) side is initiated	1128464384	43430000	I		
PST_IS_NOT_STARTED	7802	Polling Sequence Table not started (after failed OBC Core switch)	0	0	I		
FATAL_ERROR_OCCURED	7803	A fatal error occurred, which caused the previous reboot	0	0	H	Failure Code	0

(continued)

Table 17.4 S/C Event packet definitions (continued)

Event Name	Event ID	Description	Source Object ID	Object ID [hex]	Sev.	Parameter 1	Parameter 2
RF_AVAILABLE	7900	The CCSDS Board detected a RF available signal			I		
RF_LOST	7901	The CCSDS Board lost a previously found RF available signal			I		
BIT_LOCK	7902	The CCSDS Board detected a Bit Lock signal			I		
BIT_LOCK_LOST	7903	The CCSDS Board lost a previously found Bit Lock signal			I		
RF_CHAIN_LOST	7904	The CCSDS Board detected that bit lock and/or RF available are lost			I		
FRAME_PROCESSING_FAILED	7905	The CCSDS board could not process a received TC frame (e.g. 40e5)			L	Failure Code	0

17.4 FLP Flight Procedures

The following types of Flight Procedure types are defined:

- MDE High Level Mode change procedures of system and subsystems
- PRC General Procedure General procedures.
- CTG Contingency Procedures, similar to PRCs, but for non-nominal processes.

For the detailed explanation please refer to Chap. 15. The "Flying Laptop" specific Nominal Observation Mode FPs are marked in grey and are provided for completeness only (Table 17.5).

Table 17.5 Flight procedures overview

System			
Name	**Detailed Description**	**Arguments**	**Type**
YYQ00000	System Mode Change from None to Safe without GS Visibility		MDE
YYQ00001	System Mode Change from Safe to Boot without GS Visibility		MDE
YYQ00002	System Mode Change from Safe to Coarse Nadir without GS Visibility		MDE
YYQ00003	System Mode Change from Coarse Nadir to Safe without GS Visibility		MDE
YYQ00004	System Mode Change from Safe to Idle without GS Visibility		MDE
YYQ00005	System Mode Change from Idle to Safe without GS Visibility		MDE
YYQ00006	System Mode Change from Idle to Coarse Nadir without GS Visibility		MDE
YYQ00007	System Mode Change from Idle to Target Pointing_GS without GS Visibility	Target Coordinates	MDE
YYQ00008	System Mode Change from Idle to NEO1 without GS Visibility	Inertial Coordinates	MDE
YYQ00009	System Mode Change from Idle to NEO2 without GS Visibility		MDE
YYQ00010	System Mode Change from Idle to GENIUS without GS Visibility		MDE
YYQ00011	System Mode Change from higher Idle (NEO/GENIUS/Target Pointing_GS) to Idle without GS Visibility		MDE
YYQ00014	System Mode Change from any Payload Mode to Idle without GS Visibility		MDE
YYQ00015	System Mode Change from any Payload Mode to Safe without GS Visibility		MDE

(continued)

Table 17.5 (continued)

System			
YYQ00016	System Mode Change from Idle to MICS_Calib without GS Visibility	Inertial Coordinates	MDE
YYQ00017	System Mode Change from Idle to MICS_EO_Nadir without GS Visibility		MDE
YYQ00018	System Mode Change from Idle to PAMCAM_EO_Nadir without GS Visibility		MDE
YYQ00019	System Mode Change from Idle to PAMCAM_EO_Target without GS Visibility	Target Coordinates	MDE
YYQ00020	System Mode Change from Idle to OSIRIS_HPLD without GS Visibility	Target Coordinates	MDE
YYQ00021	System Mode Change from Idle to OSIRIS_EDFA without GS Visibility	Target Coordinates	MDE
YYQ00022	System Mode Change from Idle to DDS without GS Visibility	Target Coordinates	MDE
YYQ00023	System Mode Change from Idle to AIS without GS Visibility		MDE
YYQ00024	System Mode Change from Idle to ShipObs_Nadir without GS Visibility		MDE
YYQ00025	System Mode Change from Idle to ShipObs_Target without GS Visibility	Target Coordinates	MDE
YYQ00026	System Mode Change from Idle to MICS_PC_Nadir without GS Visibility		MDE
YYQ00027	System Mode Change from Idle to MICS_PC_Target without GS Visibility	Target Coordinates	MDE
YYQ00028	System Mode Change from Idle to Live_EO without GS Visibility	Target Coordinates	MDE
YYQ00029	System Mode Change from Idle to PLOC_Test without GS Visibility	Target Coordinates	MDE
YYQ01103	Perform Measurements to determine the alignment between MICS and STR CHUs	Inertial Coordinates	PRC
YYQ01104	Perform Measurements to determine the alignment between PAMCAM and STR CHUs	Inertial Coordinates	PRC
YYQ10000	Mode Change from None to Safe with GS Visibility		MDE
YYQ10001	Mode Change from Safe to Boot with GS Visibility		MDE
YYQ10002	Mode Change from Safe to Coarse Nadir with GS Visibility		MDE

(continued)

Table 17.5 (continued)

System			
YYQ10003	Mode Change from Coarse Nadir to Safe with GS Visibility		MDE
YYQ10004	Mode Change from Safe to Idle with GS Visibility		MDE
YYQ10005	Mode Change from Idle to Safe with GS Visibility		MDE
YYQ10006	Mode Change from Idle to Coarse Nadir with GS Visibility		MDE
YYQ10007	Mode Change from Idle to Target Pointing_GS with GS Visibility	Target Coordinates	MDE
YYQ10008	Mode Change from Idle to NEO1 with GS Visibility	Inertial Coordinates	MDE
YYQ10009	Mode Change from Idle to NEO2 with GS Visibility		MDE
YYQ10010	Mode Change from Idle to GENIUS with GS Visibility		MDE
YYQ10011	Mode Change from higher Idle (NEO/GENIUS/Target Pointing_GS) to Idle with GS Visibility		MDE
YYQ10014	Mode Change from any Payload Mode to Idle with GS Visibility		MDE
YYQ10015	Mode Change from any Payload Mode to Safe with GS Visibility		MDE
YYQ10016	Mode Change from Idle to MICS_Calib with GS Visibility	Inertial Coordinates	MDE
YYQ10017	Mode Change from Idle to MICS_EO_Nadir with GS Visibility		MDE
YYQ10018	Mode Change from Idle to PAMCAM_EO_Nadir with GS Visibility		MDE
YYQ10019	Mode Change from Idle to PAMCAM_EO_Target with GS Visibility	Target Coordinates	MDE
YYQ10020	Mode Change from Idle to OSIRIS_HPLD with GS Visibility	Target Coordinates	MDE
YYQ10021	Mode Change from Idle to OSIRIS_EDFA with GS Visibility	Target Coordinates	MDE
YYQ10022	Mode Change from Idle to DDS with GS Visibility	Target Coordinates	MDE
YYQ10023	Mode Change from Idle to AIS with GS Visibility		MDE
YYQ10024	Mode Change from Idle to ShipObs_Nadir with GS Visibility		MDE
YYQ10025	Mode Change from Idle to ShipObs_Target with GS Visibility	Target Coordinates	MDE
YYQ10026	Mode Change from Idle to MICS_PC_Nadir with GS Visibility		MDE

(continued)

Table 17.5 (continued)

System			
YYQ10027	Mode Change from Idle to MICS_PC_Target with GS Visibility	Target Coordinates	MDE
YYQ10028	Mode Change from Idle to Live_EO with GS Visibility	Target Coordinates	MDE
YYQ10029	Mode Change from Idle to PLOC_Test with GS Visibility	Target Coordinates	MDE
YYQ11000	Establish contact to the satellite from ground		PRC
YYQ11001	Request live TM from Ground and Downlink it automatically		PRC
YYQ11002	Request event TM from Ground and Downlink it automatically		PRC
YYQ11003	Request old HK TM from Ground and Downlink it automatically		PRC
YYQ11005	Set TLE to a value, which is an argument of the function	Current TLE	PRC
YYQ11006	Deploy the DOM at EOL		PRC
YYQ11100	Commissioning Procedure to checkout the HPC functionality		PRC
YYQ11101	Commissioning Procedure to checkout the System Safe Mode		PRC
YYQ11102	Commissioning Procedure to checkout the System Idle Mode		PRC
YYQ16000	Procedure to handle the case in which the satellite cannot be contacted from ground		CTG
YYQ16001	Perform Reboot of the whole satellite system (by rebooting PCDU)		CTG
YYQ16002	Trigger the PCDU to perform a reconfiguration loop of the whole OBC		CTG
YYQ16003	Trigger the PCDU to perform a reconfiguration loop of the I/O-Boards		CTG
YYQ16004	Switch all Safe Mode equipment from the nominal to the redundant configuration		CTG
Core DHS			
Name	**Detailed Description**	**Arguments**	**Type**
DYQ11000	Uplink, check and boot a new OBSW image	OBSW Image Data	PRC
DYQ16000	In case of an error, reboot the OBSW		CTG
CCSDS-Board			
DCQ16000	Procedure to switch to the second CCSDS Board for the uplink (change VC)		CTG

(continued)

Table 17.5 (continued)

System			
DCQ16001	Procedure to switch to the second CCSDS Board for the downlink		CTG
Memory			
DMQ11000	Procedure to configure which packets are to be saved in which TM Store	Packet Type TM Store	PRC
DMQ11001	Enable storage of Event TM packets on-board		PRC
DMQ11002	Disable storage of Event TM packets on-board		PRC
DMQ11003	Delete old data from the on-board Event TM packet store		PRC
DMQ11011	Enable storage of Housekeeping TM packets on-board		PRC
DMQ11012	Disable storage of Housekeeping TM packets on-board		PRC
DMQ11013	Delete old data from the on-board Housekeeping TM packet store		PRC
DMQ11021	Enable storage of Miscellaneous TM packets on-board		PRC
DMQ11022	Disable storage of Miscellaneous TM packets on-board		PRC
DMQ11023	Delete old data from the on-board Miscellaneous TM packet store		PRC
I/O-Board			
DIQ16000	Procedure to switch to the other IO-Bard		CTG
Processor-Board			
DOQ16000	Procedure to switch to the other OBC Processor		CTG
Scheduler			
YSQ11000	Enable execution of time-tagged commands in the on-board schedule		PRC
YSQ11001	Disable execution of time-tagged commands in the on-board schedule		PRC
YSQ16002	Delete all time-tagged commands in the on-board schedule		CTG
Time Managment			
YTQ11000	Set the on-board time manually to the current time (UTC)	Current Time	PRC
PSS Subsystem			
Name	**Detailed Description**	**Arguments**	**Type**
PYQ11000	Set a parameter of the PSS Control Algorithm	Parameter Value	PRC

(continued)

Table 17.5 (continued)

System			
PYQ11001	Update battery SoC parameters (e.g. due to degradation) in the PSS Controller and the PCDU	Parameter Values	PRC
PCDU			
PPQ11000	Procedure to reset the DOM timer and check for success		PRC
PPQ16000	Procedure to switch to the other PCDU in case of an error		CTG
PPQ16001	Reboot PCDU		CTG
Solar Panels			
PSQ01000	Perform a Measurement of the solar panel test string		PRC
TTC			
Name	**Detailed Description**	**Arguments**	**Type**
XYQ00000	RX assembly on ? normal, comm controller off ? normal, ccsds board assembly passive ? active without GS		MDE
XYQ00001	TX assembly off ? on without GS		MDE
XYQ00002	TX assembly on ? off without GS		MDE
XYQ10000	RX assembly on ? normal, comm controller off ? normal, ccsds board assembly passive ? active with GS		MDE
XYQ10001	TX assembly off ? on with GS		MDE
XYQ10002	TX assembly on ? off with GS		MDE
XYQ11000	Commissioning procedure to checkout the TTC		PRC
XYQ16000	Procedure to switch to the other TTC Transmitter		CTG
ACS			
Name	**Detailed Description**	**Arguments**	**Type**
AYQ01000	Perform Near Earth Object Detection Celestial Mode 1	Inertial Coordinates	PRC
AYQ01001	Perform Near Earth Object Detection Celestial Mode 2	Nadir Offset	PRC
AYQ01100	Commissioning Procedure to checkout the Nadir Pointing Mode		PRC
AYQ01101	Commissioning Procedure to checkout the Target Pointing Mode	Target Coordinates	PRC
AYQ01102	Commissioning Procedure to checkout the Inertial Pointing Mode	Inertial Coordinates	PRC
AYQ01103	Perform Measurements to determine the data integrity between FOG and STR		PRC
AYQ01104	Perform Measurements to determine the data integrity between FOG, STR and RW		PRC

(continued)

Table 17.5 (continued)

System			
AYQ11000	Set a parameter of the ACS Control Algorithm	Parameter Value	PRC
AYQ11001	Switch the position source from GPS to a propagator		PRC
FOG			
AFQ01000	Record FOG data at a high frequency for test purposes	Duration Frequency	PRC
AFQ11100	Commissioning procedure to checkout the FOG		PRC
GPS			
AGQ01000	Perform Measurements for the GENIUS experiment		PRC
AGQ01001	Record GPS data at a high frequency for test purposes	Duration Frequency	PRC
AGQ11000	Upload an Almanach to help the GPS acquire a fix	Almanach Data	PRC
AGQ11101	Commissioning procedure to checkout the GPS		PRC
AGQ11102	Commissioning procedure to checkout the GPS XNS		PRC
AGQ16000	Reboot GPS in case of an error		CTG
MGM			
AMQ01000	Record MGM data at a high frequency for test purposes	Duration Frequency	PRC
RWL			
ARQ01000	Record RW data at a high frequency for test purposes	Duration Frequency	PRC
ARQ11100	Commissioning procedure to checkout the RW		PRC
STR			
ASQ01002	Record STR data at a high frequency for test purposes	Duration Frequency	PRC
ASQ11000	Downlink a picture taken by the STR		PRC
ASQ11001	Perform an Update of the STR star catalog	Star Catalog Data	PRC
ASQ11100	Commissioning procedure to checkout the STR		PRC
ASQ11101	Commissioning procedure to checkout the STR MIRU		PRC
ASQ16000	Reboot STR in case of an error		CTG
TCS			
Name	**Detailed Description**	**Arguments**	**Type**
TYQ01001	Heat an unused device to its operating temperature range before using it	Device	PRC

(continued)

Table 17.5 (continued)

System			
TYQ11002	Set a parameter of the TCS Control Algorithm	Parameter Value	PRC
Heater			
THQ11100	Commissioning Procedure to switch on the heaters and record their power consumption.		PRC
Payload			
Name	**Detailed Description**	**Arguments**	**Type**
PYQ01000	Perform EO with MICS and PAMCAM	Target Coordinates Picture Frequency	PRC
PYQ01001	Perform AIS Measurement and Ship Observations with the MICS	Target Coordinates Picture Frequency	PRC
PYQ11000	Perform Live Earth Observation	Target Coordinates	PRC

Note: *Only the Payload subsystem-level Flight Procedures are listed here—using as example the implementation from the "Flying Laptop" mission. The individual "Flying Laptop" specific payload equipment Flight Procedures are not included in this platform Flight Operations Manual.*

17.5 TTC Subsystem Data Sheets

17.5.1 Bit Error Rate Reference Data

The following figures show the BER references measured at BoL together with the DLR 15 m antenna. They are provided as reference data for detection of potential degradations (Figs. 17.1 and 17.2).

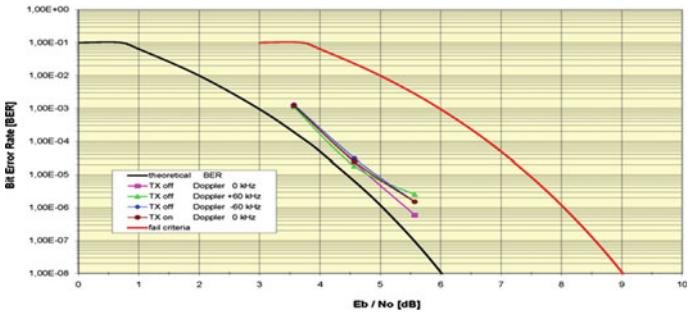


Fig. 17.1 Measured BER of RX-0. © IRS, University of Stuttgart

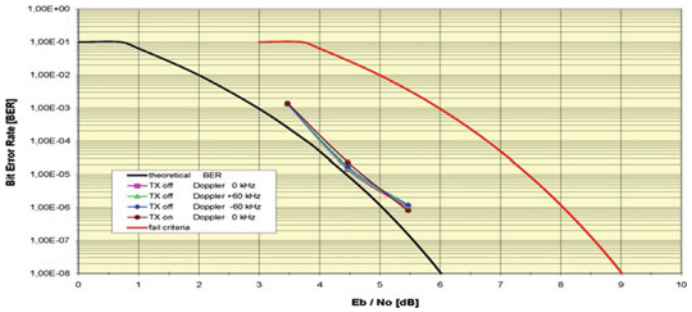


Fig. 17.2 Measured BER of RX-1. © IRS, University of Stuttgart

17.5.2 TTC Housekeeping Parameter Reference Data

The following reference diagrams are provided as measured BoL reference data for health monitoring of the receivers and transmitters—as far as available from [80]. *Note*: *No reference data for RX subcarrier loop stress have been provided* (See Figs. 17.3, 17.4, 17.5, 17.6, 17.7, 17.8, 17.9, 17.10, 17.11, 17.12, 17.13, 17.14, 17.15, 17.16, 17.17, 17.18, 17.19, 17.20, 17.21 and 17.22).

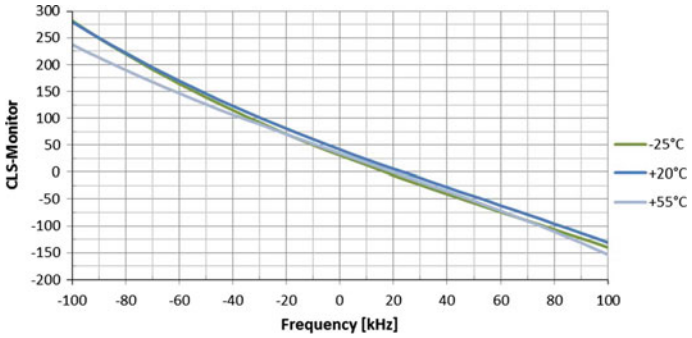


Fig. 17.3 Carrier loop stress monitor reference BoL for RX-0. © STT—from [71]

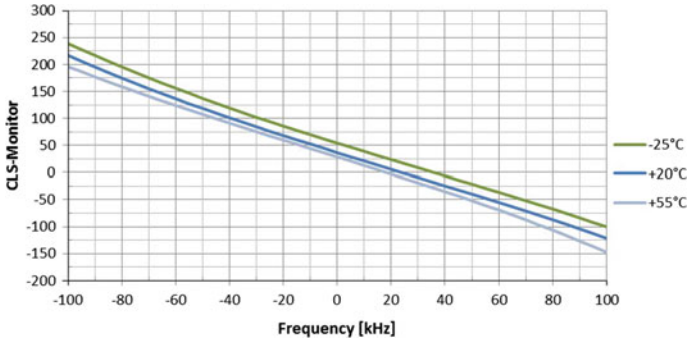


Fig. 17.4 Carrier loop stress monitor reference BoL for RX-1. © STT—from [71]

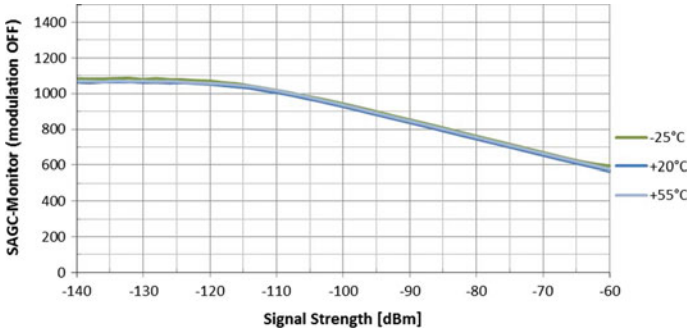


Fig. 17.5 Signal AGC monitor for RX-0. © STT—from [71]

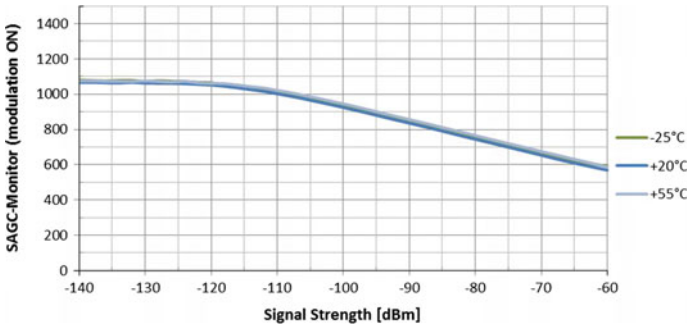


Fig. 17.6 Signal AGC monitor for RX-1. © STT—from [71]

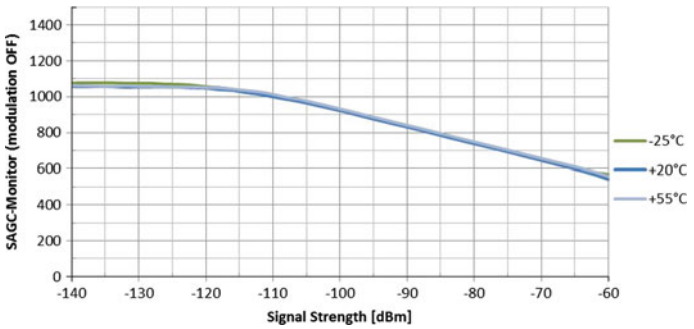


Fig. 17.7 Signal AGC monitor for RX-0 (modulation off). © STT—from [71]

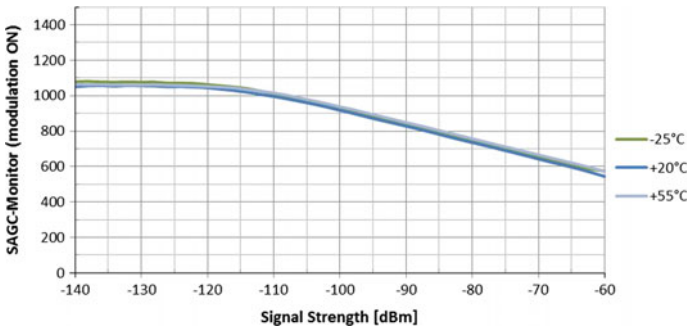


Fig. 17.8 Signal AGC Monitor for RX-1 (modulation off). © STT—from [71]

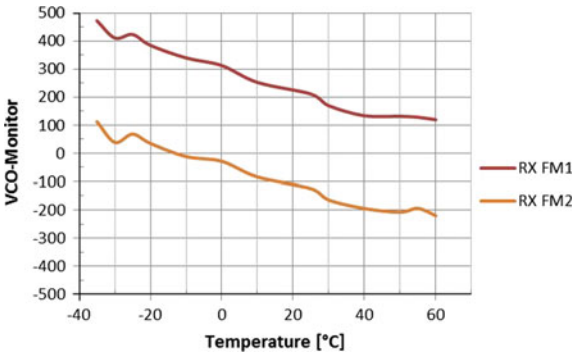


Fig. 17.9 Voltage CO control voltage over temperature. © STT—from [71]

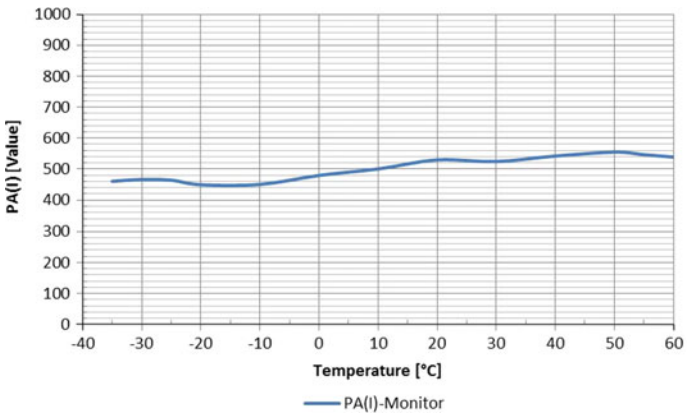


Fig. 17.10 Transmitter HPA current over temperature—TX-N. © STT—from [71]

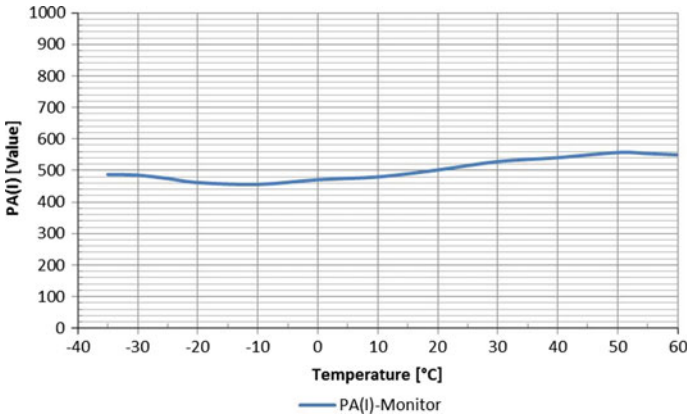


Fig. 17.11 Transmitter HPA current over temperature—TX-R. © STT—from [71]

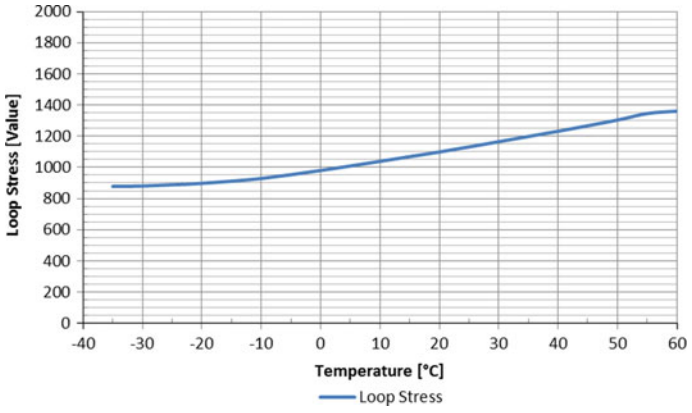


Fig. 17.12 Transmitter loop stress—TX-N. © STT—from [71]

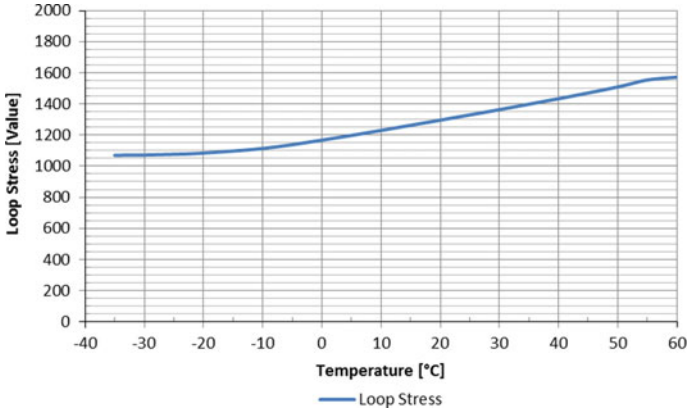


Fig. 17.13 Transmitter loop stress—TX-R. © STT—from [71]

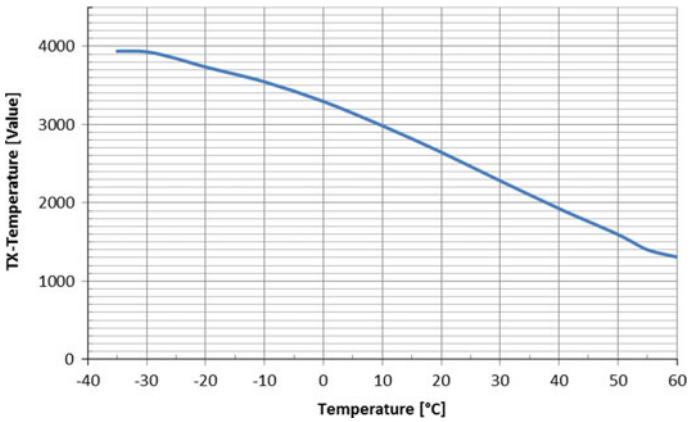


Fig. 17.14 TX temperature (telemetry vs. measured)—TX-N. © STT—from [71]

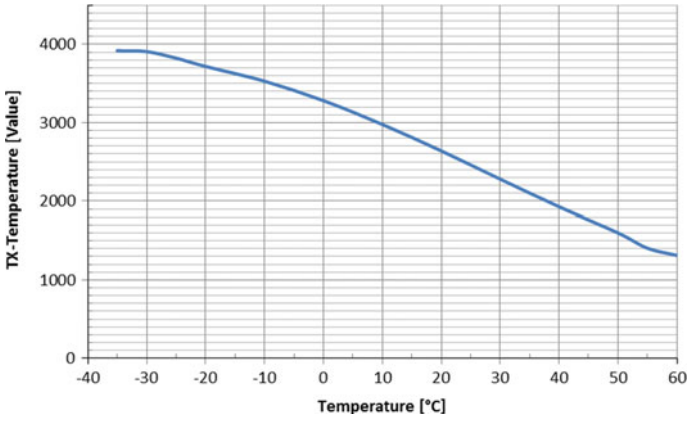


Fig. 17.15 TX temperature (telemetry vs. measured)—TX-R. © STT—from [71]

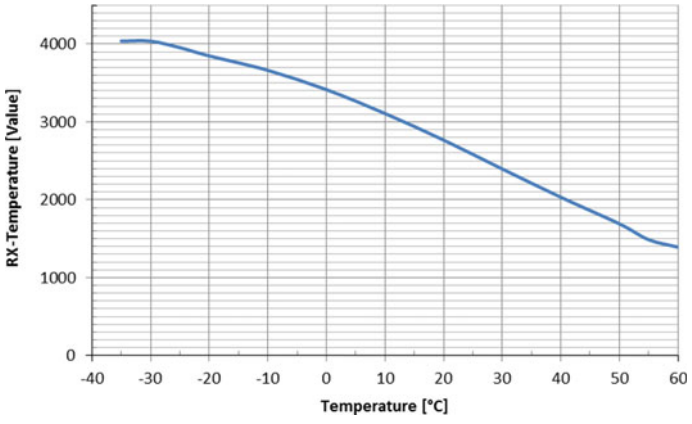


Fig. 17.16 RX temperature (telemetry vs. measured)—RX-0. © STT—from [71]

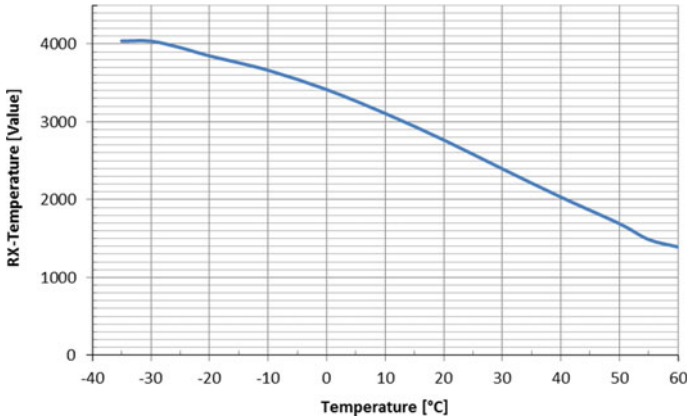


Fig. 17.17 RX temperature (telemetry vs. measured)—RX-1. © STT—from [71]

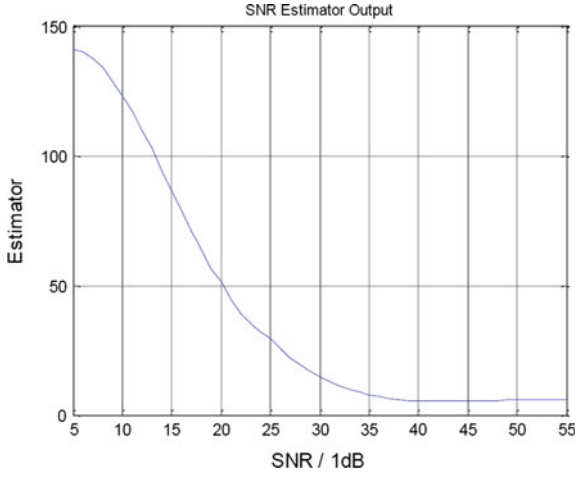


Fig. 17.18 SNR estimator versus SNR—RX-0/1. © STT—from [71]

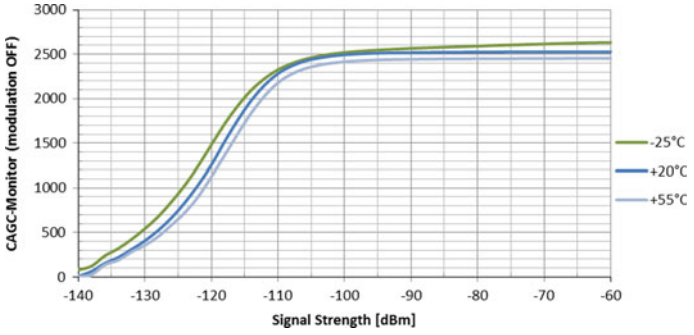


Fig. 17.19 Carrier ACG monitor (modulation off)—RX-0. © STT—from [71]

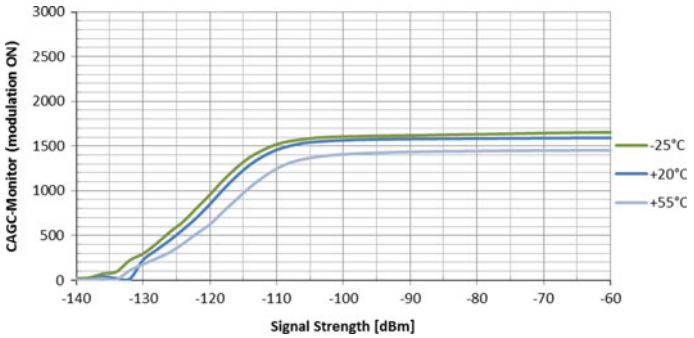


Fig. 17.20 Carrier ACG monitor (modulation on)—RX-0. © STT—from [71]

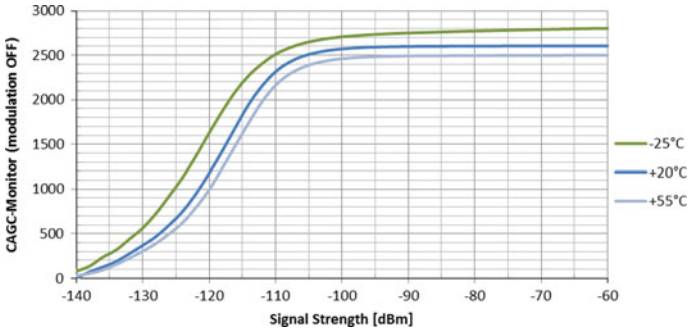


Fig. 17.21 Carrier ACG monitor (modulation off)—RX-1. © STT—from [71]

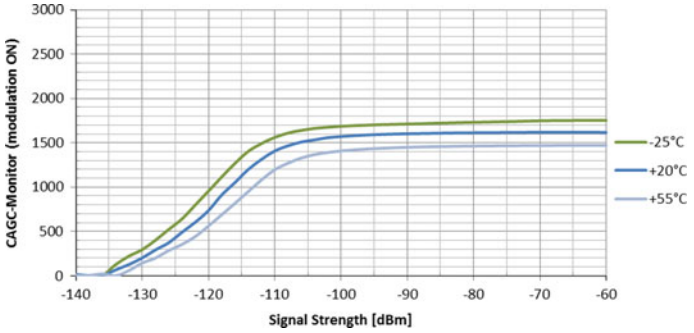


Fig. 17.22 Carrier ACG monitor (modulation on)—RX-1. © STT—from [71]

17.6 TC/TM Link Budgets for the IRS Antenna Station

Tables 17.6, 17.7, 17.8 and 17.9 depict the link budgets for operation of an FLP satellite via the antenna station of the IRS institute. The worst case is assumed for the satellite upcoming over the horizon (elevation angle 10°). The best case data refer to the satellite being exactly above the ground station (elevation angle 90°):

Table 17.6 Link Budget TC uplink (worst case)

Uplink Command Budget:		
Parameter:	**Value:**	**Units:**
Ground Station:		
Ground Station Transmitter Power Output:	5.0	W
In dBW:	7.0	dBW
In dBm:	37.0	dBm
Ground Stn. Total Transmission Line Losses:	6.6	dB
Antenna Gain:	31.9	dBi
Ground Station EIRP:	32.3	dBW
Uplink Path:		
Ground Station Antenna Pointing Loss:	0.3	dB
Gnd-to-S/C Antenna Polarization Losses:	0.2	dB
Path Loss:	164.7	dB
Atmospheric and Ionospheric Losses:	1.2	dB
Isotropic Signal Level at Spacecraft $RIP_{S/C}$:	−134.2	dBW
Spacecraft (Eb/No Method):		
Spacecraft Antenna Pointing Loss $L_{pointing\_S/C}$:	0.0	dB
Spacecraft Antenna Gain $G_{r\_S/C}$:	−8.0	dBi
Signal power after Antenna:	−142.2	dBW
Spacecraft Effective Noise Temperature Ts:	1581	K
Spacecraft Figure of Merrit (G/T):	−40.0	dB/K
S/C Signal-to-Noise Power Density (C/No):	54.4	dBHz
System Desired Data Rate:	4000	bps
In dBHz:	36.0	dBHz
Command System Eb/No:	18.4	dB
Demodulation Method:	**BPSK**	
Forward Error Correction Coding Used:	**None**	
System Allowed or Specified Bit-Error-Rate:	1.0E−05	
Eb/No Threshold:	13.5	dB
System Link Margin:	**4.9**	dB
Implementation Loss:	1	dB
Technical degradation	1.0	dB
System Link Margin after uncertain Losses:	**2.9**	dB

(continued)

Table 17.6 (continued)

Uplink Command Budget:		
Signal Power at FLP Receiver		
Min. premissable Signal Power at Receiver:	−148	dBW
Max. premissable Signal Power at Receiver:	−50	dBW
Signal power at Receiver input:	**−147.1**	dBW
	Within TTC Power Limit	

Table 17.7 Link Budget TM downlink (worst case)

Downlink Telemetry Budget:		
Parameter:	**Value:**	**Units:**
Spacecraft:		
Spacecraft Transmitter Power Output:	0.6	W
In dBW:	−2.0	dBW
In dBm:	28.0	dBm
Spacecraft Total Transmission Line Losses:	0.6	dB
Spacecraft Antenna Gain:	−8.0	dBi
Spacecraft EIRP:	−10.6	dBW
Downlink Path:		
Spacecraft Antenna Pointing Loss:	0.0	dB
S/C-to-Ground Antenna Polarization Loss:	0.3	dB
Path Loss:	165.5	dB
Atmospheric and Ionospheric Loss:	1.23	dB
Isotropic Signal Level at Ground Station RIP_{GS}:	−177.6	dBW
Ground Station (EbNo Method):		
Ground Station Antenna Pointing Loss $L_{pointingGS}$:	0.3	dB
Ground Station Antenna Gain G_{r_GS}:	33.1	dBi
Signal power after Antenna:	−144.8	dBW
Ground Station Effective Noise Temperature:	388	K
Ground Station Figure of Merrit (G/T):	7.2	dBi/K
G.S. Signal-to-Noise Power Density (C/No):	57.9	dBHz
System Desired Data Rate:	56000	bps
In dBHz:	47.5	dBHz
Telemetry System Eb/No for the Downlink:	10.4	dB
Demodulation Method Seleted:	**BPSK**	
Forward Error Correction Coding Used:	**Conv. R = 1/2,K = 7 & R.S. (255,223) IL = 5**	
System Allowed or Specified Bit-Error-Rate:	1.0E−05	
Demodulator Implementation Loss:	0	dB
Telemetry System Required Eb/No:	4.5	dB
Eb/No Threshold:	4.5	dB

(continued)

Table 17.7 (continued)

Downlink Telemetry Budget:		
System Link Margin:	**5.9**	dB
Implementation Loss:	1	dB
Technical degradation	1.0	dB
System Link Margin after uncertain Losses:	**3.9**	dB
Signal Power at GS Receiver		
Signal Power at the input of Downconverter:	−121.2	dBW
Max. Input Downconverter:	−60	dBW
Downconverter Gain:	15	dBW
Min. premissable Signal Power at IMBU:	−116	dBW
Max. premissable Signal Power at IMBU:	−60	dBW
Signal power at IMBU Receiver input:	**−106.2**	dBW
	Within IMBU Power Limit	

Table 17.8 Link Budget TC uplink (best case)

Uplink Command Budget:		
Parameter:	**Value:**	**Units:**
Ground Station:		
Ground Station Transmitter Power Output:	5.0	W
In dBW:	7.0	dBW
In dBm:	37.0	dBm
Ground Stn. Total Transmission Line Losses:	6.3	dB
Antenna Gain:	31.9	dBi
Ground Station EIRP:	32.6	dBW
Uplink Path:		
Ground Station Antenna Pointing Loss:	0.0	dB
Gnd-to-S/C Antenna Polarization Losses:	0.2	dB
Path Loss:	164.3	dB
Atmospheric and Ionospheric Losses:	0.0	dB
Isotropic Signal Level at Spacecraft $RIP_{S/C}$:	−132.0	dBW
Spacecraft (Eb/No Method):		
Spacecraft Antenna Pointing Loss $L_{pointing\_S/C}$:	0.0	dB
Spacecraft Antenna Gain $G_{r\_S/C}$:	4.0	dBi
Signal power after Antenna:	−128.0	dBW
Spacecraft Effective Noise Temperature Ts:	1474	K
Spacecraft Figure of Merrit (G/T):	−27.7	dB/K
S/C Signal-to-Noise Power Density (C/No):	68.9	dBHz
System Desired Data Rate:	4000	bps
In dBHz:	36.0	dBHz

(continued)

Table 17.8 (continued)

Uplink Command Budget:		
Command System Eb/No:	32.9	dB
Demodulation Method:	**BPSK**	
Forward Error Correction Coding Used:	**None**	
System Allowed or Specified Bit-Error-Rate:	1.0E−05	
Eb/No Threshold:	13.5	dB
System Link Margin:	**19.4**	dB
Implementation Loss:	0	dB
Technical degradation	0.0	dB
System Link Margin after uncertain Losses:	**19.4**	dB
Signal Power at FLP Receiver		
Min. premissable Signal Power at Receiver:	−147	dBW
Max. premissable Signal Power at Receiver:	−50	dBW
Signal power at Receiver input:	**−132.7**	dBW
	Within TTC Power Limit	

Table 17.9 Link Budget TM downlink (best case)

Downlink Telemetry Budget:		
Parameter:	**Value:**	**Units:**
Spacecraft:		
Spacecraft Transmitter Power Output:	1.0	W
In dBW:	−0.2	dBW
In dBm:	29.8	dBm
Spacecraft Total Transmission Line Losses:	0.6	dB
Spacecraft Antenna Gain:	4.0	dBi
Spacecraft EIRP:	3.3	dBW
Downlink Path:		
Spacecraft Antenna Pointing Loss:	0.0	dB
S/C-to-Ground Antenna Polarization Loss:	0.3	dB
Path Loss:	165.1	dB
Atmospheric and Ionospheric Loss:	0.00	dB
Isotropic Signal Level at Ground Station RIP_{GS}:	−162.1	dBW
Ground Station (EbNo Method):		
Ground Station Antenna Pointing Loss $L_{pointingGS}$:	0.0	dB
Ground Station Antenna Gain G_{r_GS}:	33.1	dBi
Signal power after Antenna:	−129.0	dBW
Ground Station Effective Noise Temperature:	145	K
Ground Station Figure of Merrit (G/T):	11.5	dBi/K
G.S. Signal-to-Noise Power Density (C/No):	78.0	dBHz
System Desired Data Rate:	56000	bps

(continued)

Table 17.9 (continued)

Downlink Telemetry Budget:		
In dBHz:	47.5	dBHz
Telemetry System Eb/No for the Downlink:	30.5	dB
Demodulation Method Seleted:	**BPSK**	
Forward Error Correction Coding Used:	**Conv. R = 1/2,K = 7 & R.S. (255,223) IL = 5**	
System Allowed or Specified Bit-Error-Rate:	1.0E − 05	
Demodulator Implementation Loss:	0	dB
Telemetry System Required Eb/No:	4.5	dB
Eb/No Threshold:	4.5	dB
System Link Margin:	**26.0**	dB
Implementation Loss:	0	dB
Technical degradation	0.0	dB
System Link Margin after uncertain Losses:	**26.0**	dB
Signal Power at GS Receiver		
Signal Power at the input of Downconverter:	−104.8	dBW
Max. Input Downconverter:	−60	dBW
Downconverter Gain:	15	dBW
Min. premissable Signal Power at IMBU:	−116	dBW
Max. premissable Signal Power at IMBU:	−60	dBW
Signal power at IMBU Receiver input:	**−89.8**	dBW
	Within IMBU Power Limit	

17.7 Power Subsystem Data Sheets

17.7.1 PCDU Switch and Fuse Allocation

The design of the PCDU includes a total of 27 fuses and 77 power switches plus 2 special switches for high-power consuming loads. The following table provides an overview of the assignments of the fuses and switches to equipment of the FLP target spacecraft (Table 17.10).

Table 17.10 Fuses and switches in the PCDU

No. of Fuse	No. of Switches	Component	Max. Current Value with margin [mA]
0	2	OBC Core N	480.0
1	2	OBC Core R	480.0
2	2	I/O Board N	150.0
3	2	I/O Board R	150.0
4	2	CCSDS 0	105.0
5	2	CCSDS 1	105.0
6	1	TC Receiver 0	280.0
7	1	TC Receiver 1	280.0
8	1	MICS G	439.5
	2	PLOC Power Channel N	1539.0
	1	DDS switch 0	435.0
9	1	MICS R	439.5
	2	PLOC Power Channel R	1539.0
	1	DDS switch 1	435.0
10	1	MICS NIR	439.5
	2	TM Transmitter N	2500.0
	1	STR N	435.0
11	1	RW 3	333.0
	1	STR R	435.0
	1	FOG 3	300.0
12	1	FOG 0	300.0
	1	RW 0	333.0
13	1	FOG 1	300.0
	1	RW 1	333.0
14	1	FOG 2	300.0
	1	RW 2	333.0
15	1	MGM 0	39.0
16	1	MGM 1	39.0
17	1	PAMCAM	153.0
	1	MGT Unit R	219.0
	1	GPS Electronics 1	141.0
18	1	GPS Electronics 0	141.0
	1	MGT Unit N	219.0
	1	GPS Electronics 2	141.0
19	2	Osiris channel 2	750.0
	2	TM Transmitter R	2500.0
20	2	Osiris channel 1	2250.0
	2	Data Transmitter R	1365.0
	1	Bimetal + Heater OBC Survival 1/ Bimetal + Heater TT&C Survival 1	2952.0

(continued)

Table 17.10 (continued)

No. of Fuse	No. of Switches	Component	Max. Current Value with margin [mA]
21	1	Bimetal + Heater OBC Survival 0/Bimetal + Heater TT&C Survival 0	2952.0
	2	Data Transmitter N	1365.0
22	2	Heater Payload Module 0	960.0
	2	Heater Core Module 0	960.0
	2	Heater Service Module 0	960.0
23	2	Heater Payload Module 1	960.0
	2	Heater Core Module 1	960.0
	2	Heater Service Module 1	960.0
24	2	Retaining Mechanism 0	2784.0
25	2	Retaining Mechanism 1	2784.0
26	2	DOM	450.0
	2	AIS Antenna	2304.0
	2	AIS Receiver	600.0

17.7.2 Power Consumption Versus Modes

Table 17.11 shows the power consumption of the spacecraft components versus different operational modes of the satellite.

Table 17.11 Power consumption versus spacecraft modes

Mode		**Launch/Shut Down Mode**		**Safe Mode**		**Idle Mode**		**Coarse Nadir Pointing Mode**		**None Mode**	
ACS Mode		**Off**		**Detumble/Safe**		**Idle**		**Target Pointing**		**Off**	
System	**max. power consumption [W]**	**duty cycle [%]**	**power consumption [W]**	**duty cycle [%]**	**power consumption [W]**	**duty cycle [%]**	**power consumption [W]**	**duty cycle [%]**	**power consumption [W]**	**duty cycle [%]**	**power consumption [W]**
"*FlyingLaptop*" *PLs*											
PLOC	11.50	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
PAMCAM	1.60	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
MICS Camera	16.40	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
AIS Receiver	6.20	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
OSIRIS LM + EDFA	26.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
OSIRIS HPLD	10.50	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
Data Downlink Sys.	15.95	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
ACS											
MGT	4.54	0	0.00	100	4.54	63	2.86	100	4.54	0	0.00
MGM	1.20	0	0.00	100	1.20	100	1.20	100	1.20	0	0.00
RW	13.54	0	0.00	0	0.00	50	6.77	0	0.00	0	0.00
STR	4.60	0	0.00	0	0.00	100	4.60	100	4.60	0	0.00
FOG	13.46	0	0.00	0	0.00	100	13.46	0	0.00	0	0.00
GPS	5.16	0	0.00	0	0.00	33	1.70	33	1.70	0	0.00
C&DH											
OBC	6.16	0	0.00	100	6.16	100	6.16	100	6.16	100	6.16

(continued)

Table 17.11 (continued)

Mode		Launch/Shut Down Mode		Safe Mode		Idle Mode		Coarse Nadir Pointing Mode		None Mode	
I/O-Board	2.20	0	0.00	100	2.20	100	2.20	100	2.20	100	2.20
CCSDS-Board	3.10	0	0.00	100	3.10	100	3.10	100	3.10	100	3.10
Power											
PCDU	24.50	2.5	0.61	33	8.09	40	9.80	33	8.09	33	8.09
Battery	1.00	80	0.80	80	0.80	100	1.00	100	1.00	80	0.80
Comms											
TTC Tx	12.00	0	0.00	10	1.20	10	1.20	10	1.20	10	1.20
TTC Rx	6.00	0	0.00	100	6.00	100	6.00	100	6.00	100	6.00
Misc.											
Srv. Module Heater	33.33	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
Core Module Heater	32.81	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
PL Module Heater	32.81	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
OBC Heater	53.42	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
TTC Heater	52.08	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
Battery Heater	8.00	0	0.00	100	8.00	0	0.00	0	0.00	100	8.00
PCDU Heater	3.50	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
Deploy Mech.	102.70	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
Sum	504.26		1.41		41.29		60.05		39.79		35.55
Total (margin 10 %)	**554.69**		**1.55**		**45.41**		**66.06**		**43.77**		**39.10**

(continued)

Table 17.11 (continued)

Mode		Target Pt for GS Contact Mode		Live EO Mode		MICS EO Mode		MICS/PAMCAM Mode		NEO Detection Mode	
ACS Mode		Target Pointing		Target Pointing		Nadir Pointing		Target Pointing		Nadir/Inertial Pointing	
System	max. power consumption [W]	duty cycle [%]	power consumption [W]	duty cycle [%]	power consumption [W]	duty cycle [%]	power consumption [W]	duty cycle [%]	power consumption [W]	duty cycle [%]	power consumption [W]
"FlyingLaptop" PLs											
PLOC	11.50	0	0.00	100	11.50	100	11.50	100	11.50	0	0.00
PAMCAM	1.60	0	0.00	100	1.60	100	1.60	100	1.60	0	0.00
MICS Camera	16.40	0	0.00	100	16.40	100	16.40	100	16.40	0	0.00
AIS Receiver	6.20	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
OSIRIS LM + EDFA	26.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
OSIRIS HPLD	10.50	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
Data Downlink Sys.	15.95	0	0.00	100	15.95	0	0.00	0	0.00	0	0.00
ACS											
MGT	4.54	63	2.86	63	2.86	63	2.86	63	2.86	63	2.86
MGM	1.20	100	1.20	100	1.20	100	1.20	100	1.20	100	1.20
RW	13.54	50	6.77	50	6.77	50	6.77	100	13.54	100	13.54
STR	4.60	100	4.60	100	4.60	100	4.60	100	4.60	100	4.60
FOG	13.46	100	13.46	100	13.46	100	13.46	100	13.46	100	13.46
GPS	5.16	33	1.70	33	1.70	33	1.70	33	1.70	33	1.70
C&DH											
OBC	6.16	100	6.16	100	6.16	100	6.16	100	6.16	100	6.16
I/O-Board	2.20	100	2.20	100	2.20	100	2.20	100	2.20	100	2.20
CCSDS-Board	3.10	100	3.10	100	3.10	100	3.10	100	3.10	100	3.10

(continued)

Table 17.11 (continued)

Mode		Target Pt for GS Contact Mode		Live EO Mode		MICS EO Mode		MICS/PAMCAM Mode		NEO Detection Mode	
Power											
PCDU	24.50	33	8.09	33	8.09	33	8.09	33	8.09	33	8.09
Battery	1.00	100	1.00	100	1.00	100	1.00	100	1.00	100	1.00
Comms											
TTC Tx	12.00	10	1.20	100	12.00	10	1.20	10	1.20	10	1.20
TTC Rx	6.00	100	6.00	100	6.00	100	6.00	100	6.00	100	6.00
Misc.											
Srv. Module Heater	33.33	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
Core Module Heater	32.81	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
PL Module Heater	32.81	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
OBC Heater	53.42	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
TTC Heater	52.08	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
Battery Heater	8.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
PCDU Heater	3.50	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
Deploy Mech.	102.70	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
Sum	504.26		58.34		114.59		87.84		94.61		65.11
Total (margin 10 %)	**554.69**		**64.17**		**126.05**		**96.62**		**104.07**		**71.62**

Table 17.12 Energy budget

Characteristic Parameters		
Standard Cells, Wing Panels		
I @ U_Batt	3.28	A
P @U_Batt	75.4	**W**
Standard Cells, Center Panel		
I @ U_Batt	1.94	A
P @U_Batt	44.59	**W**
Test String Center Panel		
I @ U_Batt	0.53	A
P @U_Batt	12.2	**W**
Max Total Power Generated	**207.59**	**W**
Losses		
General Loss (w/o degradation)	0.94	
degradation storage before launch	0.99	
Total Loss Factor	0.93	
Total Power Generated	**193.902**	**W**
Required Energy (@ 600 km)		
Average Power Consumption	66.06	W
Power Consumption Payload Ops worst	126.05	W
Duration Payload Ops Worst	15	min
Orbital Period	96.37	min
Eclipse Period	30.32	min
Sunlight Period	66.05	min
Loss Correction Factor	1.05	
Energy Required per Orbit	**127.15**	**Wh**
Generated Energy		
Energy Generated in Sunlight	164.98	Wh
Margin	**22.93**	**%**

17.7.3 Power Budget

Table 17.12 shows the energy budget worst case of the spacecraft.

When the battery is empty, the operating point of the solar panel will be approximately the same as the battery voltage, thus reducing the power from the panels with respect to the Maximum Power Point.

17.8 TCS Subsystem Data Sheets

Some additional tables concerning the thermal subsystem are provided on the next pages (Table 17.13):

Table 17.13 Sensor list with calibrations

Sensor ##	Name	Sensor Object ID [hex]	Uncalibrated Temperature Parameter Name	Uncali-brated Parameter Unit	Uncali-brated Temperature Parameter ID [hex]	Calibrated Temperature Parameter Name	Calibrated Temperature Parameter Unit	Calibrated Temperature Parameter ID [hex]
Sens.00	Battery S0 Bottom	43540001	PPTRTS00	Ohm	41111000	TSTTMP00	°C	52000100
Sens.01	Battery S1 Bottom	43540002	PPTRTS01	Ohm	41111001	TSTTMP01	°C	52000200
Sens.02	Battery S2 Top	43540003	PPTRTS02	Ohm	41111002	TSTTMP02	°C	52000300
Sens.03	Sun Sensor A	43540004	PPTRTS03	Ohm	41111003	TSTTMP03	°C	52000400
Sens.04	Battery S0 Top	43540005	PPTRTS04	Ohm	41111004	TSTTMP04	°C	52000500
Sens.05	Battery S1 Top	43540006	PPTRTS05	Ohm	41111005	TSTTMP05	°C	52000600
Sens.06	Battery S2 Bottom	43540007	PPTRTS06	Ohm	41111006	TSTTMP06	°C	52000700
Sens.07	Service Module	43540008	PPTRTS07	Ohm	41111007	TSTTMP07	°C	52000800
Sens.08	OBC 1	43540009	PPTRTS08	Ohm	41111008	TSTTMP08	°C	52000900
Sens.09	FOG 0	4354000A	PPTRTS09	Ohm	41111009	TSTTMP09	°C	52000A00
Sens.10	Core Module	4354000B	PPTRTS10	Ohm	4111100A	TSTTMP10	°C	52000B00
Sens.11	MGT Electronic	4354000C	PPTRTS11	Ohm	4111100B	TSTTMP11	°C	52000C00
Sens.12	OBC 0	4354000D	PPTRTS12	Ohm	4111100C	TSTTMP12	°C	52000D00
Sens.13	MGT Rod 1	4354000E	PPTRTS13	Ohm	4111100D	TSTTMP13	°C	52000E00
Sens.14	GPS	4354000F	PPTRTS14	Ohm	4111100E	TSTTMP14	°C	52000F00
Sens.15	MGT Rod 0/2	43540010	PPTRTS15	Ohm	4111100F	TSTTMP15	°C	52001000
Sens.16	Sun Sensor B	43540011	PPTRTS16	Ohm	41111010	TSTTMP16	°C	52001100
Sens.17	PL 1 (PAMCAM)	43540012	PPTRTS17	Ohm	41111011	TSTTMP17	°C	52001200
Sens.18	PL 4 (OSIRIS)	43540013	PPTRTS18	Ohm	41111012	TSTTMP18	°C	52001300
Sens.19	DDS Transmitter N	43540014	PPTRTS19	Ohm	41111013	TSTTMP19	°C	52001400
Sens.20	Sun Sensor C	43540015	PPTRTS20	Ohm	41111014	TSTTMP20	°C	52001500
Sens.21	Payload Module	43540016	PPTRTS21	Ohm	41111015	TSTTMP21	°C	52001600
Sens.22	DDS Transmitter R	43540017	PPTRTS22	Ohm	41111016	TSTTMP22	°C	52001700
Sens.23	FOG 1	43540018	PPTRTS23	Ohm	41111017	TSTTMP23	°C	52001800
Sens.24	Sun Sensor D	43540019	PPTRTS24	Ohm	41111018	TSTTMP24	°C	52001900
Sens.25	SA Teststring	4354001A	PPTRTS25	Ohm	41111019	TSTTMP25	°C	52001A00
Sens.26	Panel 1	4354001B	PPTRTS26	Ohm	4111101A	TSTTMP26	°C	52001B00
Sens.27	Panel 0	4354001C	PPTRTS27	Ohm	4111101B	TSTTMP27	°C	52001C00
Sens.28	Sun Sensor E	4354001D	PPTRTS28	Ohm	4111101C	TSTTMP28	°C	52001D00
Sens.29	Sun Sensor F	4354001E	PPTRTS29	Ohm	4111101D	TSTTMP29	°C	52001E00
Sens.30	Sun Sensor G	4354001F	PPTRTS30	Ohm	4111101E	TSTTMP30	°C	52001F00
Sens.31	Sun Sensor H	43540020	PPTRTS31	Ohm	4111101F	TSTTMP31	°C	52002000
Sens.32	PCDU internal P1.0	43540021	PPTTMI00	°C	41111200	TSTTMI00	°C	52010000
Sens.33	PCDU internal P1.1	43540022	PPTTMI01	°C	41111201	TSTTMI01	°C	52010100
Sens.34	PCDU internal P2	43540023	PPTTMI02	°C	41111202	TSTTMI02	°C	52010200
Sens.35	PCDU internal P3	43540024	PPTTMI03	°C	41111203	TSTTMI03	°C	52020300
Sens.36	PCDU internal P5	43540025	PPTTMI04	°C	41111204	TSTTMI04	°C	52020400

(continued)

Table 17.13 (continued)

Sensor ##	Name	Sensor Object ID [hex]	Uncalibrated Temperature Parameter Name	Uncali-brated Parameter Unit	Uncali-brated Temperature Parameter ID [hex]	Calibrated Temperature Parameter Name	Calibrated Temperature Parameter Unit	Calibrated Temperature Parameter ID [hex]
Sens.37	PL 2 (MICS Cam1)	43540026	LMTMCD00	raw	32020000	TSTMCD00	°C	53320000
Sens.38	PL 2 (MICS Cam1 LED 1)	43540027	LMTMLT00	raw	32020300	TSTMLT00	°C	53320300
Sens.39	PL 2 (MICS Cam1 LED 2)	43540028	LMTMLT10	raw	32020600	TSTMLT10	°C	53320600
Sens.40	PL 2 (MICS Cam2)	43540029	LMTMCD01	raw	32020100	TSTMCD01	°C	53320100
Sens.41	PL 2 (MICS Cam2 LED 1)	4354002A	LMTMLT01	raw	32020400	TSTMLT01	°C	53320400
Sens.42	PL 2 (MICS Cam2 LED 2)	4354002B	LMTMLT11	raw	32020700	TSTMLT11	°C	53320700
Sens.43	PL 2 (MICS Cam3)	4354002C	LMTMCD02	raw	32020200	TSTMCD02	°C	53320200
Sens.44	PL 2 (MICS Cam3 LED 1)	4354002D	LMTMLT02	raw	32020500	TSTMLT02	°C	53320500
Sens.45	PL 2 (MICS Cam3 LED 2)	4354002E	LMTMLT12	raw	32020800	TSTMLT12	°C	53320800
Sens.46	PLOC 1 PCB N	4354002F	LRTPTM00	°C	36000E00	TSTPTM00	°C	53360000
Sens.47	PLOC 2 PCB R	43540030	LRTPTM01	°C	36000F00	TSTPTM01	°C	53360100
Sens.48	PLOC 3 Xilinx N	43540031	LRTXTM00	°C	36001000	TSTXTM00	°C	53360200
Sens.49	PLOC 4 Xilinx R	43540032	LRTXTM01	°C	36001100	TSTXTM01	°C	53360300
Sens.50	TT&C 1 Power Amp Tx	43540033	XNTTTM00	V	60001500	TSTTTN00	°C	53620100
Sens.51	TT&C 2 Power Amp Tx	43540034	XRTTTM00	V	62001500	TSTTTR00	°C	53620300
Sens.52	TT&C 3 Rx	43540035	XNTRTM00	V	60001600	TSTRTN00	°C	53620000
Sens.53	TT&C 4 Rx	43540036	XRTRTM00	V	62001600	TSTRTR00	°C	53620200
Sens.54	RW 0	43540037	ARTRW000	K	14000000	TSTRW000	°C	53001000
Sens.55	RW 1	43540038	ARTRW100	K	14010000	TSTRW100	°C	53001100
Sens.56	RW 2	43540039	ARTRW200	K	14020000	TSTRW200	°C	53001200
Sens.57	RW 3	4354003A	ARTRW300	K	14030000	TSTRW300	°C	53001300
Sens.58	Magnetometer 0	4354003B	AMTMGM00	°C	12000100	TSTMGM00	°C	53000000
Sens.59	Magnetometer 1	4354003C	AMTMGM01	°C	12010100	TSTMGM01	°C	53000100
Sens.60	Star Tracker CHU A	4354003D	ASTSTR02	°C	15040200	TSTSTR02	°C	53180200
Sens.61	Star Tracker CHU B	4354003E	ASTSTR03	°C	15040300	TSTSTR03	°C	53180300
Sens.62	Star Tracker DPU CPU	4354003F	ASTSTR00	°C	15040000	TSTSTR00	°C	53180000
Sens.63	Star Tracker DPU SMPS	43540040	ASTSTR01	°C	15040100	TSTSTR01	°C	53180100
Sens.64	PL 4 (OSIRIS Laser)	43540041	LOTOLT00	°C	33001100	TSTOLT00	°C	53330100
Sens.65	PL 4 (OSIRIS Internal)	43540042	LOTOIT00	°C	33001000	TSTOIT00	°C	53330000

(continued)

Table 17.13 (continued)

Sensor ##	module heater assignment	mode component	priority	redundancy			calibration function			
							x^3	x^2	x	1
Sens.00	x	!	0	4	-1		0	1,00560799E-05	0,23555589	-247,71544649
Sens.01	x	!	0	5	-1		0	1,00560799E-05	0,23551165	-248,23362078
Sens.02	x	!	0	$	-1		0	1,00560799E-05	0,23556112	-247,65420127
Sens.03	x	!	0	-1	-1		0	1,00560799E-05	0,23558807	-247,33853131
Sens.04	x	!	0	0	-1		0	1,00560799E-05	0,23555710	-247,70131310
Sens.05	x	!	0	1	-1		0	1,00560799E-05	0,23557017	-247,54819673
Sens.06	x	!	0		-1		0	1,00560799E-05	0,23555710	-247,70131310
Sens.07	SM	!	0	-1	-1	Module average	0	-1,57819464E-05	0,29765232	-283,73674853
Sens.08	SM	Safe	3	12	-1	SM	0	1,60336018E-05	0,21905131	-236,81427929
Sens.09	CM	Idle		23	-1	CM	0	1,37438524E-05	0,22850086	-244,47983242
Sens.10	CM	!	0	-1	-1	Module average	0	7,83404400E-06	0,24223344	-252,87470222
Sens.11	CM	Safe	3	14	-1	CM	0	7,37428410E-07	0,25734162	-259,93401511
Sens.12	SM	Safe	3	(	-1	SM	0	1,60285246E-05	0,22151618	-239,20742334
Sens.13	CM	Safe	3	-1	-1	CM	0	8,14748522E-06	0,24038422	-250,59160698
Sens.14	CM	Idle		11	-1	CM	0	-4,17650021E-06	0,26980094	-267,36999678
Sens.15	CM	Safe	3	-1	-1	CM	0	4,24001783E-06	0,24951750	-255,31199744
Sens.16	x	!	0	-1	-1		0	1,00560799E-05	0,23557762	-247,46103439
Sens.17	PM	Payload	1	-1	-1	PM	0	7,44012761E-06	0,24157369	-250,55222697
Sens.18	PM	Payload	1	-1	-1	PM	0	1,23260678E-05	0,23021267	-244,03272187
Sens.19	CM	Payload	1		-1	CM	0	1,00560799E-05	0,23552110	-248,12292808
Sens.20	x	!	0	-1	-1		0	1,00560799E-05	0,23549958	-248,37492415
Sens.21	PM	!	0	-1	-1	Module average	0	1,29229755E-05	0,22917807	-243,85774049
Sens.22	CM	Payload	1	19	-1	CM	0	1,00560799E-05	0,23553458	-247,96512443
Sens.23	CM	Idle		&	-1	CM	0	1,36745654E-05	0,22741728	-243,41071473
Sens.24	x	!	0	-1	-1		0	1,00560799E-05	0,23560778	-247,10764534
Sens.25	x	!	0	-1	-1		0	1,00560799E-05	0,23550018	-248,36785915
Sens.26	x	!	0	-1	-1		0	1,00560799E-05	0,23554302	-247,86619814
Sens.27	x	!	0	-1	-1		0	1,00560799E-05	0,23552311	-248,09937587
Sens.28	x	!	0	-1	-1		0	1,00560799E-05	0,23555610	-247,71309093
Sens.29	x	!	0	-1	-1		0	1,00560799E-05	0,23551708	-248,17003190
Sens.30	x	!	0	-1	-1		0	1,00560799E-05	0,23555610	-247,71309093
Sens.31	x	!	0	-1	-1		0	1,00560799E-05	0,23555107	-247,77197932
Sens.32	SM	Safe	3	33	34	SM	0	0	1	0
Sens.33	SM	Safe	3	32	34	SM	0	0	1	0
Sens.34	SM	Safe	3	32	33	SM	0	0	1	0
Sens.35	SM	Safe	3	32	33	SM	0	0	1	0
Sens.36	SM	Safe	3	32	33	SM	0	0	1	0

(continued)

Table 17.13 (continued)

Sensor ##	module heater assignment	mode component	priority	redundancy			calibration function			
							x^3	x^2	x	1
Sens.37	PM	Payload	1	40	43	PM	0	0	&# (	-11,65
Sens.38	x	Payload	1	-1	-1		0	0	0,08	-34,8
Sens.39	x	Payload	1	-1	-1		0	0	0,08	-34,8
Sens.40	PM	Payload	1	43	37	PM	0	0	&# (	-11,65
Sens.41	x	Payload	1	-1	-1		0	0	0,08	-34,8
Sens.42	x	Payload	1	-1	-1		0	0	0,08	-34,8
Sens.43	PM	Payload	1	37	40	PM	0	0	&# (	-11,65
Sens.44	x	Payload	1	-1	-1		0	0	0,08	-34,8
Sens.45	x	Payload	1	-1	-1		0	0	0,08	-34,8
Sens.46	CM	Payload	1	47	(	CM	0	0	1	0
Sens.47	CM	Payload	1	48	(	CM	0	0	1	0
Sens.48	CM	Payload	1	49	(	CM	0	0	1	0
Sens.49	CM	Payload	1	46	(	CM	0	0	1	0
Sens.50	SM	Safe	3	51	52	SM	-3,04	25,7	-90,2	135,6
Sens.51	SM	Safe	3	50	53	SM	-3,04	25,7	-90,2	135,6
Sens.52	SM	Safe	3	53	50	SM	-3,04	25,7	-90,2	135,6
Sens.53	SM	Safe	3	52	51	SM	-3,04	25,7	-90,2	135,6
Sens.54	CM	Idle		55	46	CM	0	0	1	-273,15
Sens.55	CM	Idle		56	57	CM	0	0	1	-273,15
Sens.56	CM	Idle		57	54	CM	0	0	1	-273,15
Sens.57	CM	Idle		54	55	CM	0	0	1	-273,15
Sens.58	PM	Safe	3	-1	-1	PM	0	0	1	0
Sens.59	PM	Safe	3	-1	-1	PM	0	0	1	0
Sens.60	x	!	0	61	-1		0	0	1	0
Sens.61	x	!	0	60	-1		0	0	1	0
Sens.62	CM	Idle		63	-1	CM	0	0	1	0
Sens.63	CM	Idle		$	-1	CM	0	0	1	0
Sens.64	CM	Idle		$	-1	CM	0	0	1	0
Sens.65	CM	Idle		$	-1	CM	0	0	1	0

17.9 Mass Budget

Table 17.14 depicts the mass budget of the FLP satellite platform—excluding payloads.

Table 17.15 provides the mass budget overview on the entire "Flying Laptop" satellite—including payloads and optical bench of the cameras. The percentage distribution is visualized in addition in Fig. 17.23.

Table 17.14 FLP Generation 1 platform mass budget

Platform Component	Mass [kg]
RW	5.00
MGT	1.27
MGM	0.36
SUS	0.26
GPS	1.09
STR	1.68
FOG	2.01
OBC	4.64
PCDU	4.19
Solar Panels	6.87
Battery	10.07
Harness	9.88
Structure	26.64
Deployment Mechanism	1.86
DOM	1.66
TCS	3.48
TT&C	5.42
sum	**86.36**

Table 17.15 "Flying Laptop"—mass budget

Subsystem	Mass [kg]	Percentage
Payload	21.91	20.23 %
ACS	11.66	10.77 %
OBC	4.64	4.28 %
Harness	9.88	9.12 %
Power	21.13	19.52 %
S&M	30.16	27.85 %
TCS	3.48	3.21 %
TTC	5.42	5.00 %
	108.26	**100 %**

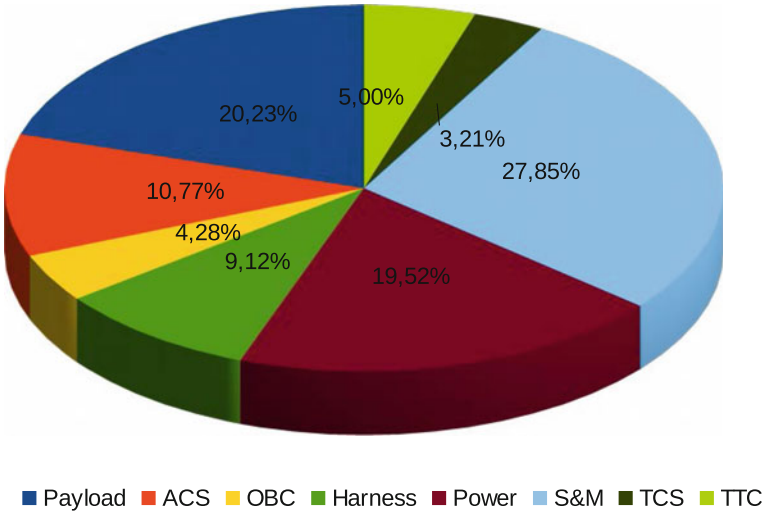


Fig. 17.23 "Flying Laptop"—mass budget distribution per subsystem. © IRS University of Stuttgart

17.10 Orbit Analysis

The "Flying Laptop" as first FLP-based mission spacecraft is intended to be launched as a piggy-back payload on an Indian PSLV rocket. The exact orbital parameters still are to be agreed with the launch provider and partly depend on the requirements of the launcher's primary passenger. Foreseen is a sun-synchronous orbit with an altitude of 500–650 km and a Local Time of Descending Node (LTDN) between 09:30 and 11:00 h.

The orbit analysis information provided in this chapter shall serve as an approximation to illustrate the platform performance.

17.10.1 Orbit Environment

A variety of orbits needs to be considered for the thermal analysis. The orbit analysis has defined 7 possible orbits for the FLP. These orbits were analyzed in [125] in order to define the parameters of both, the hot and the cold case. The results of this analysis are shown in Table 17.16. After availability of detailed input data from the launch supplier a more detailed analysis will be performed, considering orbital drift data and orbital thermal constraints.

Table 17.16 Thermal worst case scenarios of the "Flying Laptop" mission [125]

	Hot Case	Cold Case	Source
Orbit number	Orbit 04	Orbit 07	IRS-06-S04
Orbit altitude [km]	500	900	IRS-06-S04
Inclination [°]	97.4	99.03	IRS-06-S04
LTDN [hh:mm]	09:30	11:00	IRS-06-S04
Solar constant S_0 [W/m^2]	1412.11	1321.89	I-DEAS NX
Albedo [–]	0.4	0.2	Airbus DS
IR from Earth [W/m^2]	261	189	Airbus DS

17.10.2 Orbit Definition for the FLP Mission Flying Laptop

A set of slightly varying orbits is defined in order to simulate and compare the impact of different orbital conditions. Two parameters mainly define the differences of an SSO, namely the LTDN and the orbit altitude. The LTDN is the key factor for

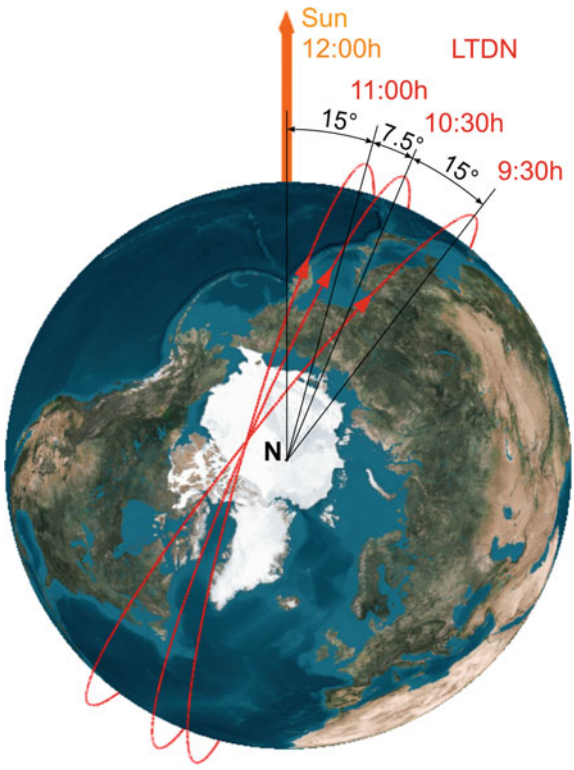


Fig. 17.24 LTDN illustration. © IRS University of Stuttgart

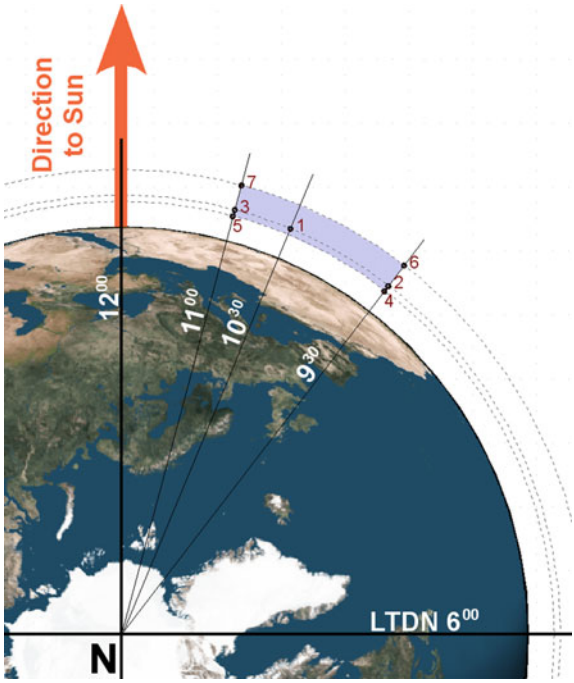


Fig. 17.25 Visualization of orbit range. © IRS University of Stuttgart

the lighting conditions for the Earth observation with the camera payloads while the orbit altitude is important for the optical resolution of the payload cameras and the slew rates during imaging maneuvers.

Table 17.21 later shows a short description of each of the seven analyzed orbits while Table 17.17 contains the corresponding definitions. Figures 17.24 and 17.25 illustrate the different LTDNs. Orbit 01 (see Table 17.17) is defined as the nominal orbit with an altitude of 600 km and an LTDN of 10:30 h. Considering former PSLV launches these values seem realistic.

Note: *Although orbits 6 and 7 have been analyzed in [125] for the "Flying Laptop" FLP Generation 1 mission, their orbit altitude is too high to be able to deorbit the spacecraft via the DOM sail in the required timeframe after mission end. The mission in such case would violate the European Code of Conduct on Space Debris Mitigation [1]. These Orbits are only relevant for missions based on the FLP Generation 2 with a propulsion system for orbit corrections and with a sufficient amount of fuel for orbit altitude reduction after nominal mission end.*

Table 17.17 Orbit definition for the Flying Laptop

Parameter	a	h*	e	i	LTAN/LTDN*	T_{Orbit}*		Revolutions*	
	[m]	[km]	[–]	[°]	[hh:mm]	[s]	[min]	[day^{-1}]	[$year^{-1}$]
Orbit 01	6 978 137	600	0	97.79	22:30/10:30	5801.23	96.69	14.89	5440
Orbit 02	6 978 137	600	0	97.79	21:30/09:30	5801.23	96.69		
Orbit 03	6 978 137	600	0	97.79	23:00/11:00	5801.23	96.69		
Orbit 04	6 878 137	500	0	97.40	21:30/09:30	5677.98	94.63	15.22	5558
Orbit 05	6 878 137	500	0	97.40	23:00/11:00	5677.98	94.63		
Orbit 06	7 278 137	900	0	99.03	21:30/09:30	6179.33	102.99	13.98	5107
Orbit 07	7 278 137	900	0	99.03	23:00/11:00	6179.33	102.99		

Orbit Epoch: 1 Jan 2012 00:00:00.000 UTCG (MJD 55927.0)
(*.additional Information; not used for definition)

17.10.3 *Longitudinal Displacement per Pass and Orbit Repeat Cycle*

The variation of the longitude of two subsequent passes at a certain latitude is of interest for Earth observation in particular. Figure 17.26 shows the ascending nodes of the satellite in Orbit 01 and illustrates the displacement. In opposition to the displacement in km, the displacement of the angular longitude depends on the given latitude (Table 17.18).

A property of the sun-synchronous orbit is that the orbit altitude may be chosen in a way to be synchronous to the rotation of the Earth. This way the ground track is repeated after a certain number of days. This orbit repeat cycle is depicted in Table 17.19.

As it is unlikely that the satellite will ever repeat the exact orbit 100 % so the repeat cycles are associated with a certain deviation given in degrees of longitude.

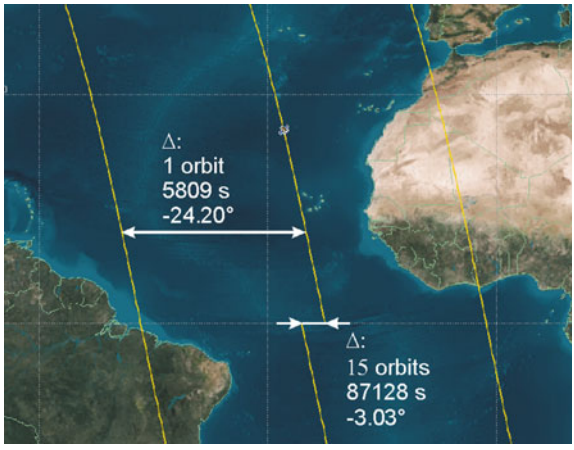


Fig. 17.26 Longitudinal displacement (Orbit 01). © IRS University of Stuttgart

Table 17.18 Displacement per pass

	Displacement in longitude						
	[deg]	[km]					
Latitude	overall	0°	20°	40°	48.674° (IRS)	60°	80°
Orbit 01–03	−24.202	2694.15	2529.44	2057.44	1771.56	1339.54	464.46
Orbit 04–05	−23.685	2636.60	2475.50	2013.76	1734.03	1311.24	454.68
Orbit 06–07	−25.776	2869.37	2693.62	2190.32	1885.69	1425.58	494.19

Table 17.19 Orbit repeat cycles

	Δ longitude [deg]	Δ time			Δ revolution number
		[s]	[h]	[day]	
Orbit 01–03	−3.0330	87128	24.20	1.008	15
	−0.0630	691215	192.00	8.000	119
Orbit 04–05	+4.7260	85266	23.68	0.987	15
	−0.0555	432014	120.00	5.000	76
Orbit 06–07	−0.8700	86608	24.06	1.002	14
	−0.2100	2592051	720.01	30.001	419

In Table 17.19 the first two typical repeat intervals are given for each orbit together with its deviation from the original orbit.

17.10.4 Contact to IRS and Other Ground Stations

A minimal elevation angle of 7° is required for contact to a ground station. 10° are deemed more realistic due to expected W-LAN noise on the campus. 7° is based on experiences of present ground stations and will be used for the accessibility prediction of the satellite. The LTAN/LTDN does not have any influence on this analysis.

Figure 17.27 shows the region where access to the IRS ground station is possible for the nominal altitude of 600 km taking the minimum elevation of 7° into account. It also shows ground tracks of three consecutive passes over the ground station. It can be identified that in the best case depicted here the satellite can be accessed from the ground station in three consecutive orbits and in any case in two consecutive orbits. This is also the case for the lower altitude of 500 km, as can be seen in Fig. 17.28.

Fig. 17.27 Contact area for nominal orbit altitude and ground track of three consecutive passes. © IRS University of Stuttgart

Fig. 17.28 Contact area for orbit altitude of 500 km and ground track of three consecutive passes. © IRS University of Stuttgart

Values for the number of contacts to be expected per day and their durations to IRS and to a partner's ground station are listed in Table 17.20. Figure 17.29 shows the orbits during which the ground stations in Stuttgart and Waco (Texas, USA) can be accessed from a satellite in the nominal orbit altitude. On the ordinate of the depicted diagram, the visibility windows are given in the range [0 … 1], where 0 and 1 denote the passes of the ascending node for each orbit. As both ground stations are located on the northern hemisphere, there is no ground station access during the second half of each orbit and thus the values greater than 0.5 are omitted. For the circular orbits considered here, the fraction of the orbit can be directly converted to time using the orbital period provided in Table 17.17 and true anomaly for further analysis. The number of orbits analyzed is limited to 119 by the second orbit repeat cycle, given in Table 17.19.

From this diagram, the maximum gap between two access intervals to the ground station at IRS can be estimated to be about 7 orbits or 11.5 h. This value can

Table 17.20 Number and duration of daily contact (mean values)

Orbit	Ground station	Mean number of contacts [day^{-1}]	Mean contact duration [min/day]	Mean contact duration [min/contact]	Max. contact duration [min]
01–03	Stuttgart (GER), IRS	4.83	36.37	7.53	9.64
04–05		4.39	29.14	6.64	8.48
06–07		5.91	59.13	10.00	12.88
01–03	Waco, TX (USA), Baylor University	3.61	27.17	7.53	9.57
04–05		3.30	21.79	6.61	8.41
06–07		4.32	43.55	10.08	12.81

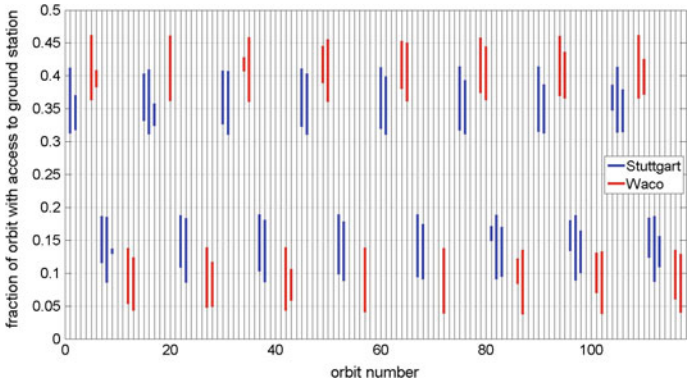


Fig. 17.29 Fraction of orbits with possible access to ground stations in Stuttgart and Waco for the nominal orbit altitude. © IRS University of Stuttgart

also be used as worst case for the LEOP analysis. If the ground station in Waco can be used, the maximum gap time could easily be reduced by half.

Due to its location closer to the equator, this ground station can only be accessed in a maximum of two consecutive passes and might only be accessible in one orbit in the worst case (see red lines in Fig. 17.29).

Figure 17.30 shows a similar diagram for an orbit altitude of 500 km. For this orbit the second repeat cycle is only 76.

At this altitude, the maximum gap time between two access intervals to Stuttgart is approximately 11.3 h. In general, similar considerations as for the nominal altitude apply with access intervals being shorter at this lower altitude.

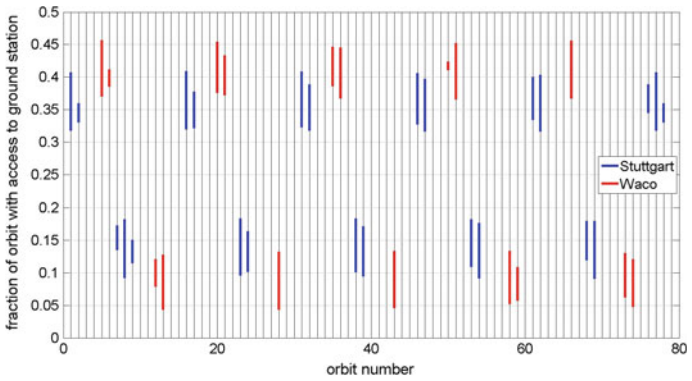


Fig. 17.30 Fraction of orbits with possible access to ground stations in Stuttgart and Waco for an orbit altitude of 500 km. © IRS University of Stuttgart

17.10.5 Sun, Umbra and Penumbra Phases

The proportion of the phases where the satellite travels in the sunlight, in the Earth's shadow or in the transition of one to another is not constant over the year, but performs a slight change. Figure 17.31 shows the evolution over 1 year exemplary for Orbit 01. The corresponding mean values are displayed in Fig. 17.32. More detailed information is provided in Table 17.21.

Depending on the orbit the sun proportion varies between 62.7 and 70.2 %. The phase where the satellite is situated in the penumbra is negligible for most of the

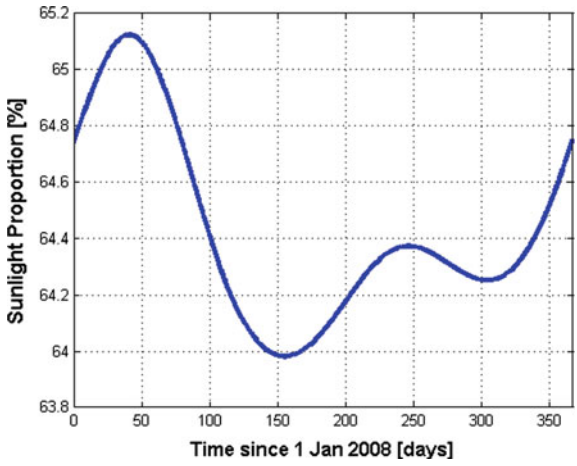


Fig. 17.31 Evolution of the sunlight proportion over 1 year (Orbit 01). © IRS University of Stuttgart

Table 17.21 Sun, umbra and penumbra times over 1 year

Orbit	Mean Values						Min Values			Max Values		
	Sun		Umbra		Penumbra		Sun	Umbra	Pen.	Sun	Umbra	Pen.
	[s]	[%]	[s]	[%]	[s]	[%]	[s]	[s]	[s]	[s]	[s]	[s]
01	3742.4	64.43	2047.3	35.25	9.4	0.32	3716.4	2005.3	8.9	3782.7	2074.1	10.1
02	3873.3	66.68	1912.9	32.93	11.2	0.38	3806.3	1814.6	10.0	3968.1	1981.8	12.7
03	3707.5	63.83	2083.1	35.86	8.9	0.31	3692.6	2056.9	8.6	3732.5	2098.5	9.4
04	3712.7	65.31	1950.1	34.31	10.8	0.38	3654.6	1864.4	9.8	3795.2	2009.9	12.2
05	3565.5	62.72	2101.4	36.97	8.7	0.31	3552.3	2078.3	8.4	3587.4	2115.0	9.2
06	4340.7	70.17	1821.0	29.44	12.3	0.40	4243.9	1680.6	10.8	4476.6	1920.5	14.3
07	4120.1	66.6	2047.1	33.09	9.6	0.31	4099.0	2010.6	9.2	4155.1	2068.7	10.1

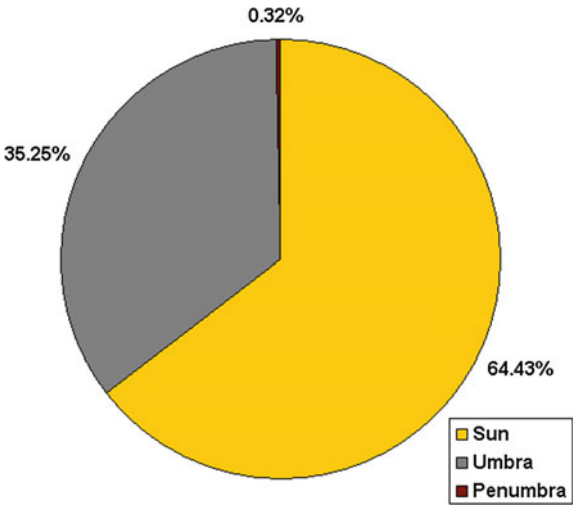


Fig. 17.32 Mean proportion of sun, umbra and penumbra (Orbit 01). © IRS University of Stuttgart

simulations as its percentage always stays below 0.5. For a conservative approach it can be added to the umbra time. Moon eclipses are not considered.

17.10.6 Orbital Drift

In an ideal SSO the orbital plane drifts 360° per solar year to maintain a constant orientation to the sun. However, in reality every orbit is affected by disturbances. For a SSO the two main interferences are the gravity of the Sun, which causes a change in the orbit inclination, and the influence of the atmosphere, which leads to a lowering of the orbit altitude. Hence, the orbit will not remain as intended for too long. This is especially the case for the FLP Generation 1 missions as this platform version is not yet equipped with an orbit control system to compensate any orbital disturbances.

Figure 17.33 shows the change of different orbital parameters over time for some relevant satellites launched into a sun synchronous orbit with a PSLV. The data in the graphs was obtained from archived TLEs at www.space-track.org. Unlike the other satellites IRS-P4 and IRS-P6 (ISRO) are equipped with an orbit control system to compensate any orbital drift, which explains the nearly optimal behavior

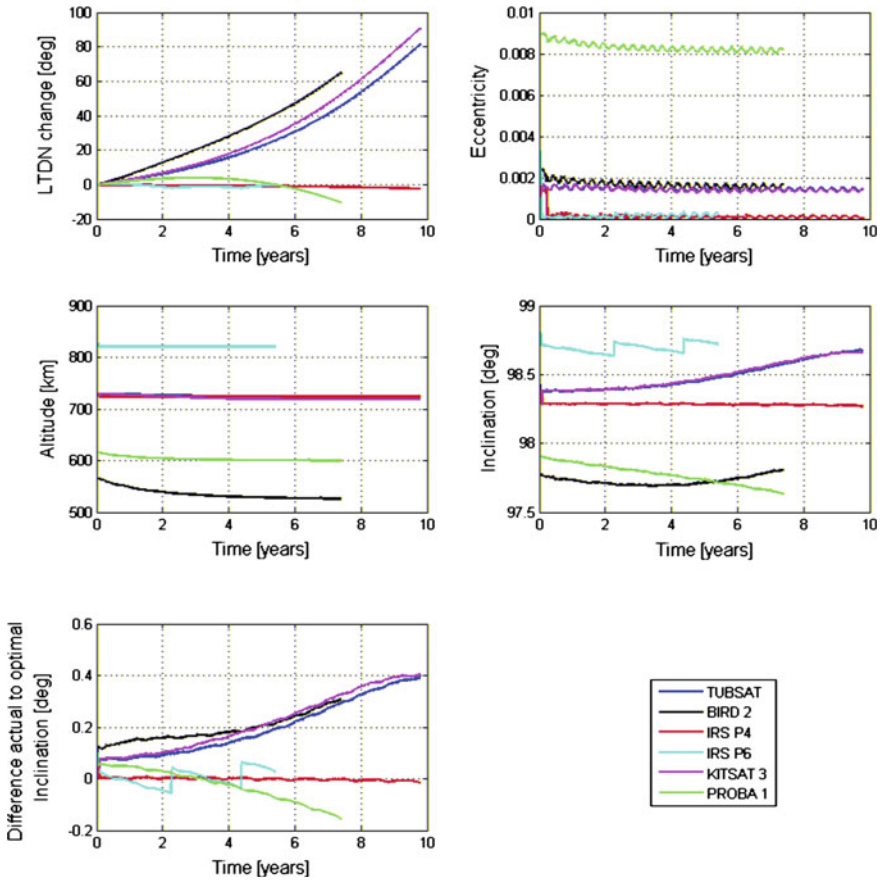


Fig. 17.33 Long-term behavior of exemplary satellites in a SSO. © IRS University of Stuttgart

of the satellite. TUBSAT, KITSAT 3 are both launched on PSLV C3 and show a reasonably similar drifting behavior, while Bird 2 and Proba 1 are launched on PSLV C2 but differ significantly in their drift characteristics. The distinction is apparently caused by the different injection conditions of the two satellites. The inclinations and especially the orbit altitudes are different and cause an unequal long time behavior.

For all satellites not equipped with an orbit control system the crucial factor for the order of magnitude of the deviations from an optimal SSO is thus the accuracy of the orbit injection. Since the final orbit and the drag properties of the satellite is not known, it is nearly impossible to conduct meaningful simulations of the orbit

drift. The examples given here (especially Bird 2) provide a rough estimate of the orbit drift behavior of FLP Generation 1 missions like the “Flying Laptop”.

17.10.7 Multi-angle Earth Observation

The FLP provides the feature of a target pointing mode for observation of celestial bodies, Moon or Earth targets. Optical multi-angle Earth observation is for example of interest for so-called Bidirectional Reflectance Distribution Function (BRDF) measurements and vegetation analysis based thereupon.

The “Flying Laptop” can be taken as representative example of an Earth observation mission with a sensor/camera arrangement and Star Tracker/Sun sensor arrangement. For such multi-angle Earth observations pointing to a dedicated target, the following operational constraints have to be taken into account:

- Sun–Zenith angle at test sites is less than 60°,
- satellite pitch angle within ±60°,
- satellite roll angle within ±5°,
- at least one star tracker is not blinded by the:
  - Sun (Sun exclusion angle = 55°),
  - Earth (Earth exclusion angle = 93°),
  - Moon (Moon exclusion angle = 8°).

The target visibility windows have to be selected accordingly and have to be considered in the mission planning and mission timeline generation.

17.11 Spacecraft Skin Connector Panel Layout

The spacecraft platform provides a skin connector panel which comprises an external power supply connector, a debug interface connector for the platform OBC and two debug connectors for the nominal and the redundant FLP Generation 1 payload computer PLOC. The overall skin connector panel and connector arrangement is depicted in Fig. 17.34. The Tables 17.22, 17.23, 17.24 and 17.25 list the connector pin allocations. For the specifics of the OBC debug connector which provides the interface to the Debug Support Unit (DSU), please refer to the CDPI reference documentation [47].

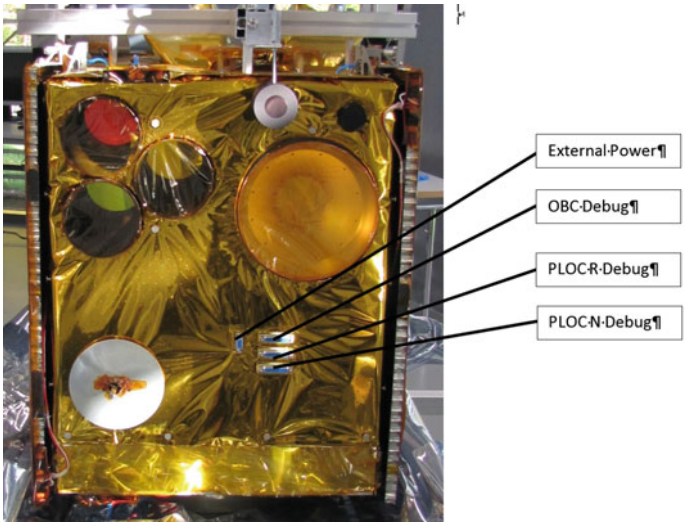


Fig. 17.34 Spacecraft skin connector panel. © IRS University of Stuttgart

Table 17.22 External Power Supply (D-Sub HD 44 S)

Pin	Pin Description	Mating Connector		Mating Connector Pin
1	PWR Charge Battery 0 (BCR0)	PCDU	J1	3
2	PWR Charge Battery 1 (BCR1)	PCDU	J2	3
3	not connected			
4	GND Charge Battery 0 (BCR0)	PCDU	J1	16
5	GND Charge Battery 1 (BCR1)	PCDU	J2	16
6	PWR Charge Battery 2 (BCR2)	PCDU	J3	3
7	not connected			
8	not connected			
9	GND Charge Battery 2 (BCR2)	PCDU	J3	16

Table 17.23 OBC Debug (D-Sub HD 44 S)

Pin	Pin Description	Mating Connector		Mating Connector Pin
1	N—DSUTMS	OBC_Power	J1	1
2	N—DSUTCK			2
3	N—DSUTDI			3
4	N—DSUTDO			4
5	N—EMDC (DSU)			5
6	N—DSUACT			6
7	N—DSUBRK			7
8	N—DSUEN			8

(continued)

Table 17.23 (continued)

Pin	Pin Description	Mating Connector		Mating Connector Pin
9	not connected			
10	not connected			
11	N—Ext. Power	OBC_Power	J1	10
12	R—Ext. Power	OBC_Power	J7	10
13	PWR EGSE	PCDU	J3	11
14	PWR EGSE			12
15	PWR EGSE			13
16	N—DSURSTN	OBC_Power	J1	9
17	N—(RS422) Data+			10
18	N—(RS422) Data-			11
19	not connected			
20	R—DSURSTN	OBC_Power	J7	9
21	R—(RS422) Data+			10
22	R—(RS422) Data-			11
23	not connected			
24	not connected			
25	not connected			
26	not connected			
27	not connected			
28	not connected			
29	not connected			
30	not connected			
31	R—DSUTMS	OBC_Power	J7	1
32	R—DSUTCK			2
33	R—DSUTDI			3
34	R—DSUTDO			4
35	R—EMDC (DSU)			5
36	R—DSUACT			6
37	R—DSUBRK			7
38	R—DSUEN			8
39	not connected			
40	N—Ext. GND	OBC_Power	J1	13
41	R—Ext. GND	OBC_Power	J7	13
42	GND EGSE	PCDU	J3	23
43	GND EGSE			24
44	GND EGSE			25

Table 17.24 PLOC R Debug (D-Sub HD 44 S)

Pin	Pin Description	Mating Connector		Mating Connector Pin
1	MMU_Debug_00	PLOC	J6	1
2	MMU_Debug_02			2
3	MMU_Debug_04			3
4	MMU_Debug_06			4
5	MMU_Debug_08			5
6	MMU_Debug_10			6
7	MMU_Debug_12			7
8	MMU_Debug_14			8
9	MMU_Debug_16			9
10	MMU_Debug_18			10
11	MMU_Debug_20			11
12	GND			12
13	not connected			
14	not connected			
15	GND			15
16	MMU_Debug_01			16
17	MMU_Debug_03			17
18	MMU_Debug_05			18
19	MMU_Debug_07			19
20	MMU_Debug_09			20
21	MMU_Debug_11			21
22	MMU_Debug_13			22
23	MMU_Debug_15			23
24	MMU_Debug_17			24
25	MMU_Debug_19			25
26	MMU_Debug_21			26
27	GND			27
28	not connected			
29	not connected			
30	GND			30
31	MMU_JTAG_TCK			31
32	MMU_JTAG_TMS			32
33	MMU_JTAG_TDO			33
34	MMU_JTAG_VJTAG			34
35	MMU_JTAG_TDI			35
36	MMU_JTAG_VPUMP			36
37	MMU_JTAG_TRST			37
38	not connected			
39	not connected			

(continued)

Table 17.24 (continued)

Pin	Pin Description	Mating Connector		Mating Connector Pin
40	GND (signal)			40
41	CPN_Debug_Rx−			41
42	CPN_Debug_Rx+			42
43	CPN_Debug_Tx−			43
44	CPN_Debug_Tx+			44

Table 17.25 PLOC N Debug (D-Sub HD 44 S)

Pin	Pin Description	Mating Connector		Mating Connector Pin
1	MMU_Debug_00	PLOC	J7	1
2	MMU_Debug_02			2
3	MMU_Debug_04			3
4	MMU_Debug_06			4
5	MMU_Debug_08			5
6	MMU_Debug_10			6
7	MMU_Debug_12			7
8	MMU_Debug_14			8
9	MMU_Debug_16			9
10	MMU_Debug_18			10
11	MMU_Debug_20			11
12	GND			12
13	not connected			
14	not connected			
15	GND			15
16	MMU_Debug_01			16
17	MMU_Debug_03			17
18	MMU_Debug_05			18
19	MMU_Debug_07			19
20	MMU_Debug_09			20
21	MMU_Debug_11			21
22	MMU_Debug_13			22
23	MMU_Debug_15			23
24	MMU_Debug_17			24
25	MMU_Debug_19			25
26	MMU_Debug_21			26
27	GND			27
28	not connected			
29	not connected			

(continued)

Table 17.25 (continued)

Pin	Pin Description	Mating Connector		Mating Connector Pin
30	GND			30
31	MMU_JTAG_TCK			31
32	MMU_JTAG_TMS			32
33	MMU_JTAG_TDO			33
34	MMU_JTAG_VJTAG			34
35	MMU_JTAG_TDI			35
36	MMU_JTAG_VPUMP			36
37	MMU_JTAG_TRST			37
38	not connected			
39	not connected			
40	GND (signal)			40
41	CPN_Debug_Rx–			41
42	CPN_Debug_Rx+			42
43	CPN_Debug_Tx–			43
44	CPN_Debug_Tx+			44

17.12 Red-Tagged Items

There are the following Red-Tagged Items to be removed before launch concerning the FLP platform.

- Debug interface connector for OBC JTAG interface board.

Further items may exist for the payload module which are mission specific such as for the "Flying Laptop" mission the:

- Camera protection covers
- OSIRIS Laser Beam Protection

17.13 Green-Tagged Items

The list of Green-Tagged Items to be mounted before launch concerning the FLP platform comprises:

- SA deployment protection connectors (4 items):
- Each of the four retaining mechanism of the solar panel deployment system are equipped with an activation connector (green-tagged item) closing both electric heater circuits on one mechanism for heating the melting braid (Fig. 5.5). These

connectors—when missing—will prevent activation, which is intended during the integration, test and transport phase. They are not mounted during AIT and transport.
- Spacecraft separation circuit (see Fig. 11.3 for PSLV and Fig. 11.4 for Soyuz/Fregat).
- Safety caps for the spacecraft skin connector panel.

There may be further items which are mission and payload specific.

17.14 Two Line Element Orbit Data Definitions

The TLE data sets provide orbit information in a format which is going back to times where computer information still was coded in punch-cards. The two lines provide the orbit information for a spacecraft in two lines—formerly representing two computer punch-cards.

The following example is taken from Wikipedia and explains a two line elements set using the data from the ISS main module ZARYA the diverse parameters of the data set are explained in the tables below:

TLE set:

ISS	(ZARYA)
1	25544U 98067A 08264.51782528 -.00002182 00000-0 -11606-4 0 2927
2	25544 51.6416 247.4627 0006703 130.5360 325.0288 15.72125391563537

Title line:

Field	Columns	Content	Example
1	01–24	Satellite name	ISS (ZARYA)

LINE 1:

Field	Columns	Content	Example
1	01–01	Line number	1
2	03–07	Satellite number	25544
3	08–08	Classification (U = Unclassified)	U
4	10–11	International Designator (Last two digits of launch year)	98
5	12–14	International Designator (Launch number of the year)	067
6	15–17	International Designator (Piece of the launch)	A
7	19–20	Epoch Year (Last two digits of year)	08
8	21–32	Epoch (Day of the year and fractional portion of the day)	264.51782528

(continued)

(continued)

Field	Columns	Content	Example
9	34–43	First Time Derivative of the Mean Motion divided by two [4]	−.00002182
10	45–52	Second Time Derivative of Mean Motion divided by six (decimal point assumed)	00000-0
11	54–61	BSTAR drag term (decimal point assumed) [4]	-11606-4
12	63–63	The number 0 (Originally this should have been "Ephemeris type")	0
13	65–68	Element set number. incremented when a new TLE is generated for this object. [4]	292
14	69–69	Checksum (Modulo 10)	7

LINE 2:

Field	Columns	Content	Example
1	01–01	Line number	2
2	03–07	Satellite number	25544
3	09–16	Inclination [Degrees]	51.6416
4	18–25	Right Ascension of the Ascending Node [Degrees]	247.4627
5	27–33	Eccentricity (decimal point assumed)	0006703
6	35–42	Argument of Perigee [Degrees]	130.5360
7	44–51	Mean Anomaly [Degrees]	325.0288
8	53–63	Mean Motion [Revs per day]	15.72125391
9	64–68	Revolution number at epoch [Revs]	56353
10	69–69	Checksum (Modulo 10)	7

References

Applicable Standards

1. European Code of Conduct on Space Debris Mitigation, Issue 1
2. ECSS-S-ST-00-01C ECSS System Glossary of Terms, Issue 3, (01 October 2012)
3. ECSS-E-ST-50-12C (31 July 2008) SpaceWire - Links, nodes, routers and networks
4. ECSS-E-ST-50-51C (5 February 2010) SpaceWire protocol Identification
5. ECSS-E-ST-50-52C (5 February 2010) SpaceWire - Remote memory access protocol
6. ECSS-E-50-12C SpaceWire cabling
7. ECSS-E-ST-50C Communications
8. ECSS-E-ST-50-01C Space data links - Telemetry synchronization and channel coding
9. ECSS-E-ST-50-02C Ranging and Doppler tracking
10. ECSS-E-ST-50-03C Space data links - Telemetry Transfer Frame protocol
11. ECSS-E-ST-50-04C Space data links - Telecommand rotocols, synchronization and channel coding
12. ECSS-E-ST-50-05C Radio frequency and modulation
13. ECSS-E-70-41A Ground systems and operations - Telemetry and telecommand packet utilization
14. ECSS-E-70-11C Space Engineering - Space Segment Operability
15. CCSDS standard, Consultative Committee for Space Data Systems: *CCSDS Recommended Standards,* Blue Books, available online at http://public.ccsds.org/publications/BlueBooks.aspx
16. CCSDS 130.0-G-2 CCSDS layer conversions
17. CCSDS-131.0-B-1 TM Synchronization and Channel Coding
18. CCSDS-132.0-B-1 TM Space Data Link Protocol
19. CCSDS 133.0-B-1 Space Packet Protocol
20. CCSDS-133.0-B-1-C1 Encapsulation Service Technical Corrigendum 1
21. CCSDS-135.0-B-4 Space Link Identifiers
22. CCSDS-201.0 Telecommand - Part 1 - Channel Service, CCSDS 201.0-B-3, June 2000
23. CCSDS-202.0 Telecommand - Part 2 - Data Routing Service, CCSDS 202.0-B-3, June 2001

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5

24. CCSDS-202.1 Telecommand - Part 2.1 - Command Operation Procedures, CCSDS 202.1-B-2, June 2001
25. CCSDS-203.0 Telecommand - Part 3 - Data Management Service, CCSDS 203.0-B-2, June 2001
26. CCSDS-231.0-B-2 TC Synchronization and Channel Coding
27. CCSDS 232.0-B-2 Telecommand Space Data Link Protocol
28. CCSDS-232.1-B-2 Communications Operation Procedure-1
29. CCSDS-401.0-B Radio Frequency and Modulation Systems
30. CCSDS-732.0-B-2 AOS Space Data Link Protocol
31. ESA PSS-04-0 Space data communications
32. ESA PSS-04-103 Telemetry channel coding standard, Issue 1
33. ESA PSS-04-105 Radio frequency and modulation standard
34. ESA PSS-04-106 Packet Telemetry Standard, Issue 1
35. ESA PSS-04-107 Packet Telecommand Standard, Issue 2
36. ESA PSS-04-151 Telecommand Decoder Standard, Issue 1
37. ECSS TM/TC Packet Utilization Standard, 30. January 2003
38. ECSS-E-10-03A Space Engineering - Testing, February 2002
39. Airbus Patent Affiliation P700377-DE-NP: Multifunktionaler Kontroller für einen Satelliten Deutsches Patent- und Markenamt DPMA 10 2012 009 513.9
40. Airbus Patent Affiliation 107973.65556US Multifunctional Controller for a Satellite US Patent Office 2015

References on “Flying Laptop” Target Satellite Documents

41. FLP-SP-220-001-IRS FLP Orbit Definition and Simulations University of Stuttgart, Stuttgart, Institute of Space Systems, Germany
42. Fist, Timothy: FLP-TN-519-006-IRS Coarse Nadir Pointing Mode - Performance Characterization Iss. 2, 16.08.2014 University of Stuttgart, Stuttgart, Institute of Space Systems, Germany
43. Axmann, R.; Gude, J.; Muehlbauer, P.: TET-1 Satellite Operations Lessons Learned Preparation of Mission, LEOP and routine Operations of 11 different Experiments, 62nd International Astronautical Congress, Cape Town, SA
44. TET-1, DLR Blog: 1. Arbeitstag im Orbit http://www.dlr.de/dlr/presse/desktopdefault.aspx/tabid-10308/471_read-4318/year-all/#gallery/6725
45. TET-1, DLR Blog: Deutscher Kleinsatellit TET-1 erfolgreich gestartet http://www.dlr.de/dlr/presse/desktopdefault.aspx/tabid-10308/471_read-4318/year-all/#gallery/6725
46. http://sanaregistry.org/r/spacecraftid/spacecraftid.html

References on FLP Satellite Platform Equipment

47. Eickhoff, J. (Ed): A Combined Data and Power Management Infrastructure, Springer, Berlin, Heidelberg, 2013 ISBN 978-3-642-35556-1
48. Eickhoff, Jens; Stratton, Sam; Butz, Pius; Cook, Barry; Walker, Paul; Uryu, Alexander; Lengowski, Michael; Röser, Hans-Peter: Flight Model of the FLP Satellite OBC and Reconfiguration Unit Data Systems in Aerospace, DASIA 2012 Conference, 14. - 16. May, 2012, Dubrovnik, Croatia
49. Leon3FT Processor: http://www.aeroflex.com/ams/pagesproduct/prods-hirel-leon.cfm
50. Stratton, Sam: Fault Tolerant LEON Processing, Devices and Circuit Cards MAPLD 2009 - Greenbelt, Maryland, August 31 - September 3, 2009

51. Aeroflex Gaisler Leon3FT IP Core: http://www.gaisler.com/cms/index.php?option=com_content&task=view&id=13&Itemid=53
52. Aeroflex Gaisler AB: CCSDS TM/TC and SpaceWire FPGA Data Sheet and User's Manual GR-TMTC-0004 July 2012, Version 1.2
53. Cobham Gaisler AB: UT699E LEON 3FT/SPARC™ V8 Microprocessor Functional Manual, Version 1.0.0, November 2014
54. GRLIB IP Core User's Manual, Aeroflex Gaisler http://www.aeroflex.com/gaisler
55. Spacecraft Data Handling IP Core User's Manual, Aeroflex Gaisler http://www.aeroflex.com/gaisler
56. PCDU Microcontroller data sheet: RENESAS Electronics, Renesas 32-Bit RISC Microcomputer, SuperH RISC engine Family/SH7040 Series, hardware manual, Issue 6.0, 2003
57. VECTRONIC Aerospace GmbH: Interface Control Document & Operation Manual for Power Control and Distribution Unit Type VPCDU-1, Project IRS-FLP TD-VAS-PCDU-FLP-ICD18.doc, Date 28.04.2014
58. Battery data sheet: A123 Systems, Inc.: Nanophosphate High Power Lithium Ion Cell ANR26650M1B, MD100113-01, 2011
59. Solar cells data sheet: AZUR SPACE Solar Power GmbH, GAGET1-ID/160-8040 Data Sheet, 2005
60. Solar array test string data sheet: RWE Space Solar Power GmbH, RWE3G-ID2*/150-8040 Data Sheet, 2005
61. Micro Advanced Stellar Compass TM/TC Interface Control Document ASC-DTU-ICD-3004, Issue 3.1, 31.01.2011
62. Micro Advanced Stellar Compass, User's Manual, ASC-DTU-MA-3001, Issue 2.0, 07.06.2006
63. Montenbruck, O.; Markgraf, M.: User's Manual for the Phoenix GPS Receiver, Tech. Rep. GTN-MAN-0120, Rev. 1.9, German Space Operations Center, 14. July .2008
64. FLP-TN-518-002-IRS FOG-Electrical_ICD Issue Draft, 26.10.2011 University of Stuttgart, Stuttgart, Institute of Space Systems, Germany
65. FLP-ICD-512-002-IRS Reaction_Wheel_Assembly-Electrical_ICD Issue 1, 19.07.2012 University of Stuttgart, Stuttgart, Institute of Space Systems, Germany
66. FLP-ICD-513-001-IRS MGT_Electronic_ICD Issue 3, 03.01.2013 University of Stuttgart, Stuttgart, Institute of Space Systems, Germany
67. FLP-ICD-514-001-IRS Magnetometer Electrical ICD Issue Draft, 11.09.2011 University of Stuttgart, Stuttgart, Institute of Space Systems, Germany
68. Reaction Wheel Assembly RSI 01-5/28 Technical Description Rockwell Collins, 2007.
69. http://www.astrofein.com/astro-und-feinwerktechnik-adlershof/produkte/raumfahrt-eng/10/rw-90/
70. Taubenreuther, P.: HF-System Electrical Interfaces, FLP-STT-TRX-DWG-102, Issue 1A, 2012 STT-SystemTechnik GmbH, Gneisenaustr. 15, D-80992 Munich, Germany
71. Serbe, P; Taubenreuther, P.: TMTC-System Technical Description - Housekeeping Interface FLP-STT-TRX-RPT-101, Iss. 1C, 04.11.2014 STT-SystemTechnik GmbH, Gneisenaustr. 15, D-80992 Munich, Germany
72. SpaceQuest: S-Band Turnstile Antenna Datasheet, SpaceQuest, Ltd., Fairfax, VA, USA, 2009
73. NASA radiation PD-ED-1258: Space radiation Effects on Electronic Components in Low-Earth Orbit, April 1996, NASA - Johnson Space Center (JSC)
74. Dual-Core LEON3-FT SPARC V8 Processor GR712RC User's Manual, April 2015, Issue 2.5
75. LEON Board Hardware Interface Document iOBC-D1.3-01-HWICD01 SSBV Space & Ground Systems

References on FLP Subsystems

76. FLP-TP-532-003-IRS Battery Life Cycle Test Procedure, University of Stuttgart, Stuttgart, Institute of Space Systems, Germany
77. Dittmar, Marek; Bucher, Nico: FLP-ICD-566-001-IRS FLP Space Link ICD Issue 1.6, 04.12.2014, University of Stuttgart, Stuttgart, Institute of Space Systems, Germany

References on Test Plans, Reports and Performance Data

78. Wiedemann, Klaus: RF Compatibility Test Plan and Procedure FLP-DLR-TP-1001, Iss. 1 Rev. 0 DLR Raumflugbetrieb und Astronautentraining, Weilheim, 01.12.2014
79. Wiedemann, Klaus: RF Compatibility Test Report FLP-DLR-TR-1001, Iss. 0 Rev. 1 DLR Raumflugbetrieb und Astronautentraining, Weilheim, 15.12.2014
80. Taubenreuther, P.: TMTC-System- Acceptance Test Report, FLP-STT-TRX-TR-101, Issue 1C, 05.12.2014 STT-SystemTechnik GmbH, Gneisenaustr. 15, D-80992 Munich, Germany

References on FLP Software

81. OAR Corporation: http://www.rtems.com
82. RTEMS C User's Guide http://docs.rtems.org/doc-current/share/rtems/pdf/c_user.pdf
83. Aeroflex Gaisler RTEMS: http://www.aeroflex.com/gaisler
84. FLP On-Board Software Architectural Design Documentation FLP-DDD-1221204-001-IRS, Issue 4, 2014

References on Mission Control Systems and Infrastructure

85. http://www.egos.esa.int/portal/egos-web/products/MCS/SCOS2000/
86. EGOS-MCS-S2K-ICD-0001 SCOS-2000 Database Import ICD Iss. 6.2, Aug. 2006
87. EGOS-MCS-S2K-SRD-0002 SCOS-2000 Commanding Software Requirements Document Issue 2.6
88. EGOS-MCS-S2K-SUM-2210 SCOS-2000 Telemetry Desktop Operator User Manual
89. EGOS-MCS-S2K-SUM-2230 SCOS-2000 Variable Packet Display Operator User Manual
90. EGOS-MCS-S2K-SUM-2360 SCOS-2000 Telecommand History Display Operator User Manual Issue 1.8
91. EGOS-MCS-S2K-SUM-2220 SCOS-2000 Out Of Limits Display Operator User Manual
92. CCS-TER-MAN-001 CCS User Manual Issue 1.24 Terma B.V., Leiden, The Netherlands
93. CCS-TER-DD-001 CCS5 Design Description Issue 2.1 Terma B.V., Leiden, The Netherlands
94. European Ground Systems Common Core http://www.egscc.esa.int/system.html http://www.egscc.esa.int/architecture.html#conceptual
95. Database System Software User Manual – EEMCS EEMCS-LOG-SUM-0001 Issue 6.2 GMV, 08/09/06
96. J. Houser SCOS-2000 Database Editing System User's Manual Issue 2.1
97. EGOS-NIS-NCTR-ICD-0002 NIS/NCTRS Volume 2- Detailed interface Definition: MCS
98. CCSDS 910.4-B-2 Cross Support Reference Model—Part 1: Space Link Extension Services Issue 2, October 2005

99. CCSDS 911.1-B-3 Space Link Extension—Return All Frames Service Specification Issue 3, January 2010
100. CCSDS 912.1-B-3 Space Link Extension—Forward CLTU Service Specification Issue 3, July 2010
101. CCSDS 913.1-B-1 Space Link Extension —Internet Protocol for Transfer Services
102. Satellite Services BV: Design Description & Operation and Maintenance Manual SU-SSBV-TTCIMBU-MA-0001
103. Satellite Services BV: Command and Data ICD SU-SSBV-TTCIMBU-ICD-0002
104. Satellite Services BV: SpacelinkNGT TM/TC Baseband Processor Operation and Maintenance Manual SLNGT-SAT-MA-0308
105. www.work-microwave.de
106. http://en.wikipedia.org/wiki/Tcl
107. ECSS-E-ST-70-32C Space Engineering – Test and operations procedure language
108. Bucher, N: FLP-TN-630-003-IRS Flying Laptop Space Link Extension Installation Report, Issue 1.2, April 2014
109. Systems Toolkit http://www.agi.com/products/
110. http://en.wikipedia.org/wiki/Two-line_elements
111. http://de.wikipedia.org/wiki/Satellitenbahnelement
112. http://www.satobs.org/tletools.html
113. CCSDS 502.0-B-2 Orbit Data Messages Recommended Standard Blue Book November 2009
114. DLR-GSO-GDS-ICD-001 GSOC/GDS Multi-Mission Support Interface Control Document, Iss. 0.6, March 2014
115. LSE Space GmbH: Scheduling Software Interface ICD, Issue 1, November 2008
116. Gully, S.: GDS Scheduling Interface Description Document DLR/GSOC, November 2010
117. http://www.zds-fr.com/upload/docs/030004-ftp62-ed7-crt.pdf
118. FLP-TN-630-002-IRS MOIS Operation and Administration Guide University of Stuttgart, Stuttgart, Institute of Space Systems, Germany
119. MOIS Scheduler ICD GEN-MOIS-M6-ICD-001 Issue 1.1, October 2013 RHEA System S. A., Wavre, Belgium
120. Scheduler-User Manual ESO-MOIS-UM-RHEA-0002 Issue 1.0, July 2008 RHEA System S. A., Wavre, Belgium
121. FLP-SP-245-001-IRS FLP Procedure Guidelines Iss. 1.6, 13.05.2015 University of Stuttgart, Stuttgart, Institute of Space Systems, Germany
122. FLP-TN-630-003-IRS Technical Note on FLP Mission Planning Issue 1.7, May 2015

IRS Thesis Documents

123. Uryu, A. N.: Development of a Multifunctional Power Supply System and an Adapted Qualification Approach for a University Small Satellite, PhD Thesis, University of Stuttgart, Stuttgart, Institute of Space Systems, Germany, Dr. Hut, München, 2013, ISBN 978-3-8439-0852-8
124. Grillmayer, Georg: An FPGA based Attitude Control System for the Micro-Satellite Flying Laptop. PhD thesis, Institute of Space Systems. Dr. Hut, München, 2009, ISBN 978-3-86853-654-6
125. Zeile, Oliver: Entwicklung einer Simulationsumgebung und robuster Algorithmen für das Lage- und Orbitkontrollsystem der Kleinsatelliten Flying Laptop und PERSEUS. PhD thesis, Institute of Space Systems. Dr. Hut, München, 2012, ISBN 978-3-8439-0316-5
126. Franz, Julian: Evaluation verschiedener operationeller Prozeduren zur Optimierung der Bodenkontaktzeit für den Kleinsatelliten “Flying Laptop” Bachelor Thesis, IRS, Uni Stuttgart, IRS-13-S06, 2013

127. Bätz, Bastian: Development of scheduling design and equipment handling algorithms for the software of the small satellite Flying Laptop Diploma Thesis, IRS, Uni Stuttgart, IRS-11-S21, July 2011
128. Bucher, Nico: Integration of attitude control functions into the on-board software for the small satellite Flying Laptop Diploma Thesis, IRS, Uni Stuttgart, IRS-12-S50, 2012
129. Steinmetz, Fabian: Realisierung des Thermalkontrollsystems für einen universitären Kleinsatelliten PhD Thesis, IRS, Uni Stuttgart, 2014
130. Keil, Sebastian: Design of a control algorithm for thermal control and calibration of the temperature sensors for the small satellite Flying Laptop Diploma Thesis, IRS, Uni Stuttgart, IRS-14-S057, 2014
131. Lengowski, Michael: Entwicklung mechanisch/thermischer Architekturen und innovativer Strukturelemente im Rahmen zweier Satellitenmissionen des Stuttgarter Kleinsatellitenprogramms, Dr. Hut, München, 2014, ISBN 978-3-8439-1427-7
132. Witt, Rouven: Failure Detection, Isolation and Recovery Capabilities for the university Small Satellite Flying Laptop, PhD thesis, Institute of Space Systems, 2015
33. Bucher, Nico: Funktionaler Verifikationsansatz für Onboard Software Komponenten eines Kleinsatelliten PhD Thesis, IRS, Uni Stuttgart, under Development, 2016
134. Bätz, Bastian: Entwicklung, Implementierung und Verifikation einer konfigurier- und portierbaren Steuerungssoftware für teilautonome Echtzeitsysteme PhD Thesis, IRS, Uni Stuttgart, under Development, 2016
135. Hagel, Philipp: Modular Software Architecture for FPGA based Payload Module Computers PhD Thesis, IRS, Uni Stuttgart, under Development, 2016
136. Salzburger, Stefan: Design, Implementation and Tests of ACS Failure Detection Methods for the Small Satellite "Flying Laptop" Diploma Thesis, IRS Uni Stuttgart, under Development, 2015

References on Launch Vehicles

137. Ramakrishnan. S.: PSLV - Auxiliary Satellite - User's Manual Issue 1 Indian Space Research Organisation, 1999
138. Perez, Edouard: Soyuz User's Manual Issue 2 Ariane Space, March 2012 http://www.arianespace.com/launch-services-soyuz/Soyuz-Users-Manual-March-2012.pdf

References on S/C Engineering

139. Eickhoff, Jens: Simulating Spacecraft Systems, Springer, 2009, ISBN: 978-3-642-01275-4
140. Eickhoff, Jens: Onboard Computers, Onboard Software and Satellite Operations - An Introduction, Springer, 2011, ISBN 978-3-642-25169-6
141. Wertz, J. R.; Larson, W. J.: Space Mission Analysis and Design, 3rd ed., Microcosm Press, 1999, ISBN 978-1881883104
142. Ley: Handbuch der Raumfahrttechnik Hanser Verlag, 2008
143. Gilmore, David G.: Spacecraft Thermal Control Handbook, Volume 1 - Fundamental Technologies (2nd Edition), American Institute of Aeronautics and Astronautics, Reston, VA, 2007, ISBN 978-1-884989-11-7

Referenced on other Missions and Studies

144. Assessment Study for Space Based Space Surveillance Demonstration Mission (Phase A) ESA Contract No.: 4000106392/12/NL/AF Assessment of FLP as Platform for the SBSS Demonstrator AirbusDS-FLP-SBSS, Iss. 1, 12.12.2014
145. Reichel, R.: Auslegung von `Fly-by-Wire' Systemen. Lecture Notes, ILS University of Stuttgart, 2012

Referenced Papers and Conference Contributions

146. Eickhoff, Jens; Herpel, Hans-Jürgen; Willich, Georg; Pastena, Max; Walker, Paul: A Consistent Avionics Computing Platform for Spacecraft Control and Data Processing Data Systems in Aerospace, DASIA 2015 Conference, 14.-16. May, 2015, Barcelona
147. Eickhoff, Jens; Fritz, Michael; Witt, Rouven; Bucher, Nico; Röser, Hans-Peter: Test Program of the "Combined Data and Power Management Infrastructure" Data Systems in Aerospace, DASIA 2013 Conference, 14.-16. May, 2013, Porto
148. Herpel, H.-J.; Willich, G.; Vogel, T.: OMAC4S - Open Modular Avionics for Space Applications, DASIA 2013 Conference, 14 - 16 May, 2013, Porto, Portugal
149. http://www.fokus.fraunhofer.de/go/obcsa
150. Eickhoff, Jens; Stratton, Sam; Butz, Pius; Cook, Barry; Walker, Paul; Uryu, Alexander; Lengowski, Michael; Röser, Hans-Peter: Flight Model of the FLP Satellite OBC and Reconfiguration Unit Data Systems in Aerospace, DASIA 2012 Conference, 14. - 16. May, 2012, Dubrovnik, Croatia
151. Eickhoff, Jens; Cook, Barry; Walker, Paul; Habinc, Sandi A.; Witt, Rouven; Röser, Hans-Peter: Common board design for the OBC I/O unit and the OBC CCSDS unit of the Stuttgart University Satellite `Flying Laptop' Data Systems in Aerospace, DASIA 2011 Conference, 17 - 20 May, 2011, San Anton, Malta
152. Eickhoff, Jens; Stevenson, Dave; Habinc, Sandi; Röser Hans-Peter: University Satellite featuring latest OBC Core & Payload Data Processing Technologies, Data Systems in Aerospace, DASIA 2010 Conference, Budapest, Hungary, June, 2010
153. Witt, Rouven.; Kennedy, A.; Baetz, Bastian; Mohr, Ulrich; Bucher, Nico; Eickhoff, Jens: Implementation of Fault Management Capabilities for the Flying Laptop Small Satellite Project through a Failure-Aware System Model AIAA Infotech@Aerospace 2013 Conference, August 2013, Boston, USA
154. Pollard, Mark: The Design, Build, Test and In-orbit Performance of Low cost Xenon Warm Gas Propulsion Systems, Space Propulsion Conference, May 03 – 06, 2010, San Sebastian, Spain
155. Heinen, W.; Blake, R.; Fortuno, J.; Camino, O.: Smart-1 Ground Operations Automation SpaceOps, Rome, Italy, 2006
156. Cook, Barry M.; Walker, C. Paul H.: SpaceWire Network Topologies, 1st International SpaceWire Conference September 17-19, 200, Dundee, UK
157. Klemich, K.; Lengowski, M.; Röser, H.-P.; Speichermann, J.: A robust and cost-effective approach for a battery system based on off-the-shelf Lithiumironphosphate cells for the small satellite Flying Laptop, 9th IAA Symposium on Small Satellites for Earth Observation, Berlin, Germany, April 09-12, 2013

158. Gessner, R.: Kösters, B.; Hefler, A.; Eilenbeger, R.; Hartmann, J.; and Schmidt, M. (Eds.): Hierarchical FDIR Concepts in S/C Systems, SpaceOps Montreal, Quebec, Canada, May 17 – 21, 2004
159. Lange, M.; Holzwarth, M.; Schulte, G.; Peukert, M.; Feindt, O.: Feasibility Study and Performance Assessment of a Myriade Propulsion Module with an ADN based Green Monopropellant 46th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit, 25 - 28 July 2010, Nashville, TN, USA

Index

J. Eickhoff (ed.), *The FLP Microsatellite Platform*,
Springer Aerospace Technology, DOI 10.1007/978-3-319-23503-5