ASTRO APS STAR TRACKER FOR SENTINEL 2 AND EARTHCARE

Title:	**STR PUS ICD**		
Doc. No.:	GS2.ICD.JOP.STR.02402		
Issue/Revision	5/e	Date:	07.03.2013
Date of first issue:	05.09.2009		

prepared by	Behrens V. Behrens	Date:	07.03.2013
checked by	F. Ahlendorf F. Ahlendorf	Date:	08.03.2013
approved by PA	A Knop A. Knop	Date:	08.03.2013
approved by PM	J. Mehlhorn J. Mehlhorn	Date:	08.03.2013

Distribution Record

No.	Name	Dept./Org.	No.	Name	Dept./Org.
1	Thomas Zinke	Astrium			
2	Jürgen Mehlhorn	JOP			

Change Record

Issue	Date	Description of Change
Draft	05.09.2009	Initial issue
1/a	22.01.2010	• issue numbers of AD1 and AD3 updated • references to several AD1 sections updated (acc. to new AD1 issue) COM_SW-OPS-12_AI-02, COM_SW-OPS-19_AI-02, COM_SW-OPS-20_AI-01, COM_SW-OPS-22_AI-01: • TC summary table modified: service and subservice values added; command data field length values replaced by TC packet length values; TC_CHECK_MEMORY and a lot of other TC packets added COM_SW-OPS-12_AI-02, COM_SW-OPS-19_AI-02, COM_SW-OPS-69_AI-01: • TM summary table modified: PCAT, service and subservice, and SID values added; TM structure length values replaced by TM packet length values; several TM packets added COM_SW-OPS-9_AI-01, COM_SW-OPS-9_AI-02: • Timestamp format corrected (3 bytes subseconds will be used in PUS protocol structures; reduced format with 6 bytes only will be used in STR application data structures) • PRID numbers changed (acc. to new AD1 issue) • service numbers for STR mode management, STR parameter management and STR table management changed (acc. to new AD1 issue) • STR modes: diagram and description completely revised • the flag enablePrecessionCorrection is now right aligned in its byte COM_SW-OPS-25_AI-01: • typo in CRC polynomial corrected

Issue	Date	Description of Change
		COM_SW-OPS-14_AI-02: • TM_EDB and TM_PDB now treated as TM(3,26) packets (diag) • any (3,x) services which relate to the definition of new HK/DIAG SIDs and to the reporting of SID definitions are omitted (not supported by the STR) COM_SW-OPS-16_AI-01: • Service 6 described in more detail COM_SW-OPS-23_AI-01: • allocation of SIDs to Hk and Diag packets (see TM summary table and detailed descriptions of TM(3,25), TM(3,26) packets) • table of all used SIDs added COM_SW-OPS-15_AI-01: • preliminary list of candidate events given in section 9 (to be discussed) • Block protocol: descriptions completely revised and reset request handling in both directions described • Service 3: several additional subservices now supported • Service 5 now supported • Service 8 now supported (for SELFTEST execution and long duration command management) • Service 9 described in more detail • Service 223 added (for STR specific memory handling TCs which do not fit in service 6) • the STR specific parameter setting service 224 is still used instead of service 140 because the introduction and handling of ParamIDs (necessary to use service 140) would require too many changes in the existing APSW. • the detailed TC packet descriptions have been completely revised and dedicated subsections for all formerly missing TC packets have been added • the detailed TM packet descriptions have been completely revised and dedicated subsections for all formerly missing TM packets have been added • Memory Layout has been detailed and allocation of memory IDs has been added
1/b	11.05.2010	§1.2.1: updated the issue numbers of AD1 and AD3 COM_SW-OP-86-AI01: • document chapters 6 (STR command descriptions) and 7 (STR TM descriptions) have been reordered acc. to ascending PUS service and subservice numbers • §10: added the list and description of all FIDs used by the STR COM_SW-OP-45-AI01: • Table 9.1 and §9.31: event EV_CYCLE_TOO_LONG added • Table 9.1 and §9.32: event EV_CYCLE_OVERRUN added COM_SW-OP-63-AI01: • Table 2.1 and §6.4.2: changed subtype of TC(8,2) to 220 COM_SW-OP-69-AI01 • §§6.7.2, 6.10.3, 7.2.1: reference frame and scaling for the velocity vector described in more detail COM_SW-OP-72-AI01, COM_SW-OP-87-AI01 • Table 9.1 and §§9.9, 9.10, 9.11, 9.12, 9.13, 9.14: event EV_NEW_MODE removed and added the following specific evts each identifying the reached mode by its dedicated EID value:

Issue	Date	Description of Change
		– EV_OPMODE_BOOT – EV_OPMODE_STANDBY – EV_OPMODE_PHOTO – EV_OPMODE_AADF – EV_OPMODE_AADW – EV_OPMODE_NAT COM_SW-OP-75-AI01 • Table 2.1, Table 2.2, and §§6.1.6, 6.1.7, 7.2.6: TC(3,130), TC(3,131), and TM(3,129) lengths and structures adapted to the new packet rate period format given in S2/EC PUS issue 7 COM_SW-OP-77-AI01 • Table 9.1 and §9: dedicated TM(5,1) and TM(5,3) event packets added for the successful and failed time sync verification reports COM_SW-OP-80-AI01 • §9.15: added a note to explain the term ‘streak processing mode’ as a submode in AADf COM_SW-OP-76-AI01 • Table 9.1 and §9.3: to report the currently active selftest step a dedicated parameter has been added to the event EV_SELFTEST_INPROGRESS (packet length increased by 1 byte) COM_SW-OP-94-AI01, COM_SW-OP-109-AI01 • §13: the MemIDs used by the STR are no longer dependent on the STR instance (all STRs use the same MemID numbers) • §13: the MemIDs have been corrected acc. to the S2/EC PUS, annex B2.10 (hex-dec misinterpretation corrected) • §13: a MUL column has been added to specify the address increment to be applied for the corresponding memory areas COM_SW-OP-106-AI02: • §7.2.4: the pixel value parameters of the detector window delivered with a TM_PDB packet are now numbered in ascending order from 0 to 224 regardless of the window size; several notes have been added which describe the field usage COM_SW-OP-107-AI01: • §2.3.1, Table 2.2: added the default enable state and default and max selectable packet rate for all periodic HK/Status/Diag packets COM_SW-OP-108-AI01: • §6.7.4: added a note which indicates that the actual reset execution will be delayed by 0.5s §2.2.1, Table 2.1: • packet length of the TC_PHOTO command corrected • the TC_PRECESSION, TC_ABERRATION, and TC_RATE_LIMIT commands are now also accepted in STANDBY and PHOTO mode • packet lengths of TC_LOAD_CALIBRATION1, TC_LOAD_CALIBRATION2 and TC_LOAD_ALGORITHM commands changed • the TC_ENABLE_TEMPERATURE command has been added

Issue	Date	Description of Change
		§2.3.1, Table 2.2: • packet lengths of TM_CALIBRATION_DUMP and TM_ALGORITHM_DUMP packets changed §4.2: current memory read/write enable states added to the STR's health status message §5.9: subtype of the TC_RESET command corrected (must be 4) §6.7.3: application data length of TC_PHOTO command corrected §6.8.1: application data length of TC_LOAD_CALIBRATION1 command increased (8 float params added) §6.8.2: application data length of TC_LOAD_CALIBRATION2 command decreased (1 float param removed) §6.8.8: application data length of TC_LOAD_ALGORITHM command increased (several params have been added to alg. structure) §6.8.11: upper limits for the command parameters of TC_DUMP_CALIBRATION increased because the calibration data structure length has been increased §6.9.1, note 1 modified: for security reasons the meaning of the specific timeout value 0 has been changed from 'no limit' to 'disable memory reads' §6.9.2, note 4 modified: for security reasons the meaning of the specific timeout value 0 has been changed from 'no limit' to 'disable memory writes' §§6.9.2 and 6.9.4: added a note that the H/W related write protection mechanism of the EEPROM is not affected by these commands §6.10.1: added the TC_ENABLE_TEMPERATURE command §§6.10.2, 6.10.3, and 6.10.4: the TC_PRECESSION, TC_ABERRATION, and TC_RATE_LIMIT commands are now also accepted in STANDBY and PHOTO mode §6.10.5: noted that the command is only accepted when enabled §6.10.6: typo in word "watdogError" corrected §7.6.1: upper limit of the application source data length of the TM_CALIBRATION_DUMP packet increased §7.6.3: application source data length of the TM_ALGORITHM_DUMP packet increased §9: defined and described all EID values used by the STR §12.2: modified the calibration data structure (8 params added, 1 param removed) and updated the param offsets and the structure length accordingly §12.3: modified the algorithm parameter structure and updated the param offsets and the structure length accordingly §7.2.1, §2.3.1 (Table 2.2): added attitude quality index §§2.2.1 (Table 2.1), 6.8.2, 6.8.11, 7.6.1, 12.2: added sigmaX and sigmaY
1/c	28.05.2010	§4.2: corrected the description part of 3 entries of the health status message table §§6.3.2, 6.8.14 and 6.8.15: replaced the *nPerPkt* parameter by the *maxDumpSizeInBytes* parameter §5.1: STR role names corrected (A➔1, B➔2, C➔3) acc. AD1,§B2.1 §10, FID list: • changed the names, descriptions and parameter lists of the following 5 FIDs: FID_INVALID_NPERPKT ➔ FID_INVALID_MXDMPSZ FID_INV_NPERPKT_IN_TCDMPGSC ➔ FID_INV_MXDMPSZ_IN_TCDMPGSC FID_INV_NPERPKT_IN_TCDMPGSCIDX ➔ FID_INV_MXDMPSZ_IN_TCDMPGSCIDX FID_ILL_XMIN_IN_TCPHOTO➔FID_ILL_XRANGE_IN_TCPHOTO FID_ILL_YMIN_IN_TCPHOTO➔FID_ILL_YRANGE_IN_TCPHOTO • changed the description of FID_ILL_XWIDTH_IN_TCPHOTO and FID_ILL_YHEIGHT_IN_TCPHOTO and removed the 2nd parameter of these FIDs • added (223,10) and (223,11) to the "generating services" list of FID_FAILED_MEM_ACCESS • value of FID_CMD_NOT_ALLOWED changed to 0x180 (the previous value 0x200 was already in use)

Issue	Date	Description of Change
		• FID_LONGDUR_QUEUE_FULL added with value 0x181 • for FID_LONGDUR_TC_ABORTED and FID_TC_NOT_ENABLED now the service number will be delivered as Param1 and the subservice number as Param2 • corrected the description of FID_ILL_QUAT_IN_TCACQ • FID_INERNAL_STR_ERROR added with value 0xAFFF • removed the following 8 FIDs: FID_ILL_PARAM_IN_TCLDCAL1, FID_ILL_PARAM_IN_TCLDCAL2, FID_ILL_PARAM_IN_TCLDCAL3, FID_ILL_PARAM_IN_TCLDCAL4, FID_ILL_PARAM_IN_TCLDCAL5, FID_ILL_PARAM_IN_TCLDCAL6, FID_ILL_PARAM_IN_TCLDPREP, FID_ILL_PARAM_IN_TCLDALG §§6.8.9, 7.6.4, 12.4: description of the mag-Parameters corrected
1/d	07.09.2010	§1.2.1: • Update of AD1 and AD2 §10, FID list: • corrected decimal representation of FIDs 601(hex), 602(hex), and 603(hex) • corrected Report Type for FID 181(hex) to TM(1,8) • added the common FID value FID_HK_ACTIVE to the list, which is reported on TC(3,130) and TC(3,131) if the specified SID is currently enabled §13: • Updated address ranges in the memory map §4.3.2: • Increased ATR/ATC timeout to 500ms §§4.4, 4.5: • New sections on supported mode commands and legal messages §§6.3.2, 6.8.14, 6.8.15, 7.4.1, 7.6.4, 7.6.5, 10: • Updated limits for maxDumpSizeInBytes parameters §2.2.1: • Clarification of TC packet length in Table 2-2 §4.2: • Clarification of “realTimeExecutionFailure” §§10, 11.14: • Added definition of memory access error codes §§2.2.1, 6.5.2, 9.25, 10: • Clarified conditions for EV_TIME_SHIFT • Added threshold parameter for allowed time deviation to TC_SELECT_SYNC_SOURCE • New error code FID_INVALID_SHIFT_THRESHOLD • Updated Telecommand Summary §12.3: • Replaced unused maxAttPrediction parameter by allowedAcquisitionDurationSeconds • Clarification of allowedProcessingTimeInMs and minimumSpareTimeInMs §4.2: • Added “accumulatedNumberOfTcErrors” information §§13, 13.1, 4.2: • Added “Communication failure buffer” • Added associated “writeEnaState_MemID_91” flag §§2.2.1, 10: • Distinction of four classes of TCs rather than just “short duration” or “long duration” • Added “FID_CMD_TEMPORARILY_INHIBITED”
1/e	31.01.2011	§4.4: • Corrected mode code value of “Transmit last command”

Issue	Date	Description of Change
		§4.5: • OBT reception is legal on the broadcast address, too §2.2.1: • Corrected classification of long duration commands in Table 2-2 §2.3: • Added information on priorization of TM_ADB §4.2: • Removed “TBC”s from descriptions of “internalIfFailure” and “temporaryFailureOnData” §4.3.2: • Removed “TBC” from maximum delay of next ATR §12.3: • Fixed unit of maximumRate • Added minStarsSsva • Fixed description of enableSmallestWindowSize §6.8.5: • Corrected the description of x_{192}, y_{192}, and $mode_{192}$ §§6.7.2, 6.7.3, 7.1.1, 7.1.2, 7.1.3, 7.1.4, 9.3, 9.26, 9.33, 9.34: • Fixed corrupted text §5.1: • Changed fall-back PRID to 0x29
2/a	02.03.2011	§1.2.1: • update version of Applicable Documents §12.3: • Fixed unit of reservedTimeForReadOut §4.3 (in response to GS2.CN.JOP.STR.00002 resp EC.CN.JOP.STR.00002): • Increased the ATC timeout to 600ms • Changed reset handling to protect against repetitive resets
2/b	20.04.2011	§4.3 : • Corrected typo §5.1 : • Corrected document reference §12.3: • Clarified meaning of offsetMin, offsetWin, offsetStart, and offsetDelta
3/a	30.05.2011	§9.2: • EV_STR_ALIVE now contains the health message (COM_SW-OPS-34) • EV_STR_ALIVE is now disabled by default in the ASW (COM_SW-OPS-34) §4.3.3, §4.3.4 • Corrected description of MIL Bus protocol reset §7.2.2 • Corrected SID of TM_SDB
4/a	19.09.2011	§6.4.1: • Corrected allowed range of the fid parameter §13.2: • New section defining the location of the numeric software version identifiers §10: • Corrected references to “Common Fault Identifiers (FID)” section from AD1 §§9.1, 9.35: • Added new EV_INVALID_GSC §10: • Added new FID_INVALID_GSC §13 (Table 13-1): • Added information on the location of the GSC

Issue	Date	Description of Change
		§§1.2.1, 9.1: • Added information on deviations with respect to EID allocation §12.3: • Added “enableFpnForNonGuideStars” and “enableTrackingAtFovBorder” switches §§6.7.2, 6.10.4, 12.3: • Updated interpretation of the special value 0 for rate limits
5/a	24.02.2012	Update for Boot Mode and Application Software 1.5 In response to GS2.NCR.JOP.STR.00013 / EC.NCR.JOP.STR.00013: §§2.2.1, 6.5.2: • Corrected command TC_SELECT_SYNC_SOURCE §§2.2.1, 5.13, 6.10.7: • Added new command TC_TIME_SHIFT_THRESHOLD §10: • FID 2311(dec) is now generated by TC_TIME_SHIFT_THRESHOLD rather than TC_SELECT_SYNC_SOURCE §4.2: • Added timeShiftThreshold to health and status message In response to GS2.NCR.JOP.STR.00014 / EC.NCR.JOP.STR.00014: §§2.2.1, 5.11, 6.8: • Replaced TC_LOAD_CALIBRATION2..TC_LOAD_CALIBRATION6 by TC_LOAD_CALIBRATION3..TC_LOAD_CALIBRATION7 • Split TC_LOAD_CALIBRATION1 into TC_LOAD_CALIBRATION1 and TC_LOAD_CALIBRATION2 §§2.2.1, 5.11, 6.2, 6.3, 6.8: • Corrected size restrictions for TCs with variable length §6.2: • Corrected maximum number of EID in command according size restriction for TC §6.3: • Corrected maximum size of memory to upload §6.8: • Corrected maximum number of catalog entries / GSC indices to upload In response to GS2.NCR.JOP.STR.00015 / EC.NCR.JOP.STR.00015: §§2.3.1, 7.3.2: • Corrected size of parameter n in TM_DISEVENTS_DUMP and resulting TM length In response to GS2.NCR.JOP.STR.00016 / EC.NCR.JOP.STR.00016: §§2.2.1, 6.3.2: • Corrected command TC_DUMP_MEMORY §§2.2.1, 5.13, 6.10.8: • Added new command TC_MAX_DUMPSIZE §2.3.1: • Clarified source for count parameter in TM §7.4.1: • Added reference to TC_MAX_DUMPSIZE §10: • FID 1543(dec) is now generated by TC_MAX_DUMPSIZE rather than TC_DUMP_MEMORY

Issue	Date	Description of Change
		• Added FID 273(dec) FID_MTU_TOO_SMALL §4.2: • Added maxDumpSizeInBytes to health and status message Furthermore: §§2.3,6.8.10,6.8.11,6.8.15,7.2.3,7.6.4,10,12.4,12.5: • Corrected supported range of catalog numbers §§6.9.5, 6.9.6: • Clarified description §7.2.6: • Corrected the length of period_SDB, period_ADB, and period_TDB • Removed units §§1.2.1, 2.2.1, 2.3.1: • Added information on deviations with respect to generic PUS service allocation §1.2.1: • Added information on deviation with respect LOBT synchronisation §§5.2, 5.8 • Added section describing the synchronization of LOBT • Updated section about PUS service 9 §2.2: • Up to 5 commands per cycle §§6.7.3, 7.2.4 • Clarified window size used in Photo mode §§9.1, 9.2 • Corrected source data length §13.1.1 • Added alignment bytes to Communication Failure Buffer
5/b	07.06.2012	§9.1 • Added information on the time of event generation • EV_STR_ALIVE is disabled by default in some modes • Added note that EV_TRAP_MINOR is currently not implemented • Corrected modes for EV_TRAP_MAJOR, EV_CYCLE_TOO_LONG, and EV_CYCLE_OVERRUN §§2.2.1, 6.8.15, 6.8.16, 7.6.4, 7.6.5 • Correction and clarification of update rates for long duration commands
5/c	28.08.2012	In response to ASW_QR_PANEL-34 and -38: §§2.2.1, 2.3.1 • new sections specifying impact of AADf mode on telecommand processing and telemetry update rates §§2.3.2, 4.3.1, 6, 7, 9 • added references to §2.2.1 resp. §2.3.1 In response to ASW_QR_PANEL-52: §2.3.2 • corrected modes for TM_MEM_DUMP and TM_MEM_CHECK §9.1 • corrected modes for EV_EEREAD_INPROGRESS Furthermore: §§5.6, 5.12, 6.3.1, 6.3.2, 6.3.3,6.9.1, 6.9.2, 6.9.3, 6.9.4, 6.10.1, 6.10.5: • clarification of ENABLE/DISABLE command behavior
5/d	14.12.2012	in response to e-mail from Nils Harmsen, 06.12.2012

Issue	Date	Description of Change
		§6.8.2 .. §6.8.7 • correct sub service for TC_LOAD_CALIBRATION*n* in response to e-mail from Nils Harmsen, 04.12.2012 §6.8.15, §6.8.16 • clarification of *maxDumpSizeInBytes* in response to Sentinel2&Earthcare - Observation #679 §6.4.1 • update note regarding parameter *reqTests* for command TC_SELFTEST • add examples for parameter *reqTests* for command TC_SELFTEST §9.3 • update note regarding parameter *reqTests* for event EV_SELFTEST_INPROGRESS • add examples for parameter *reqTests* for event EV_SELFTEST_INPROGRESS §9.4, §9.5 • update note regarding parameter *reqTests* for events EV_SELFTEST_OK, EV_SELFTEST_FAILED
5/e	07.03.2013	§§6.9.5, 6.9.6, 6.10.5 • clarification of ENABLE/DISABLE command behavior §3.3.6 • removed incorrect note in response to NCTS-report GS2-67 §7.2.1 • correct note regarding velocity vector in case of disabled aberration correction in response to NCTS-report GS2-69 §7.2.1 • add note regarding julian data in case of disabled precession correction §6.7.2 • add note regarding a-priori time information

Table of Contents

List of Tables

1 INTRODUCTION

1.1 SCOPE

This document is the PUS Interface Control Document of the APS star sensor for Sentinel-2 and EarthCARE and covers the mapping between the PUS services defined in the Sentinel-2 Packet Utilization Standard and:

- the STR command input (contents and format)
- the STR output data (contents and format)

of the ASTRO APS star sensor.

To minimize the changes to the existing STR S/W the PUS protocol layer to be implemented for the STR shall not affect the internal data structures already in use by the STR S/W (as far as possible). The PUS protocol layer shall be implemented merely as a translating layer. Its purpose is to translate incoming TC packets to internal command structures and translate periodically and on-request generated internal TM structures to outgoing TM packets. This translation layer covers the packetization activities for outgoing TM data as well as the necessary checks for incoming TC packets including the reporting on TC acceptance and TC execution completion.

The STR for Sentinel-2 and EarthCare will be fully compliant with the MIL bus protocol specifications contained in AD3 for PUS packet terminals. Especially the block data transfer protocol specified in AD3 will be used. This is specified in detail in chapter 4 of this document.

1.2 DOCUMENTS

1.2.1 APPLICABLE DOCUMENTS

AD1	Sentinel-2 Packet Utilization Standard	GS2.STD.ASD.SY.00001, issue 9
AD2	Sentinel-2 Star Tracker Assembly Requirement Specification; Section “4.3.5 PUS”	GS2.RS.ASD.STR.00003, issue 3
AD3	MIL BUS Protocol Specification	GS2.RS.ASD.SY.00005, issue 10
AD4	STR Event ID Allocation	GS2.RFD.JOP.STR.00043, issue 1
AD5	STR-PUS implementation in comparison to S2-PUS-ICD Volume B	GS2.RFD.JOP.STR.00047, issue 1
AD6	LOBT Synchronisation	GS2.RFD.JOP.STR.00045, issue 3

1.2.2 REFERENCED DOCUMENTS

1.3 ACRONYMS

AAD	Autonomous attitude determination
ADB	Attitude Data Block
ADC	Analog to Digital Converter
ASCII	American Standard Code for Information Interchange
ATF	Adaptive threshold filter
BC	Bus Controller
BIT	Built-in Test
BRF	Boresight reference frame
CCITT	Comité Consultatif International Télégraphique et Téléphonique
CMD	Command
COBT	Central On-Board Time
CRC	Cyclic Redundancy Check
EDAC	Error Detection And Correction
EDB	Engineering Data Block
EEPROM	Electrically Erasable PROM

FDIR	Failure Detection, Isolation, and Recovery
FIFO	First In, First Out
FOV	Field of view
GSC	Guide star catalog
HAS	High accuracy star tracker
ICD	Interface control document
IEEE	Institute of Electrical and Electronics Engineers
JD	Julian day
JOP	Jena-Optronik GmbH
LOBT	Local On-Board Time
LOS	Line of sight (boresight or z-axis)
LSB	Least Significant Bit
MSB	Most Significant Bit
N/A	Not applicable
NAN	Not a number
NAT	Nominal attitude tracking
OBT	On-Board Time
PDB	Pixel Data Block
PDF	Packet Data Field
PRID	Process Identifier
PROM	Programmable ROM
PWM	Pulse Width Modulation
RAM	Random access memory
ROM	Read only memory
SAU	Smallest Addressable Unit
SCET	Spacecraft Elapsed Time
SDB	Status Data Block
SGM	Safe-guard memory
STR	Star tracker
S/W	Software

TBC	To be confirmed
TBD	To be defined
TBW	To be written
TC	Telecommand
TDB	Tracker Data Block
TEC	Thermo-Electrical Cooler
TM	Telemetry

2 GENERAL

2.1 STR MODES

Figure 2-1 below shows the mode switching diagram of the ASTRO APS autonomous star tracker.

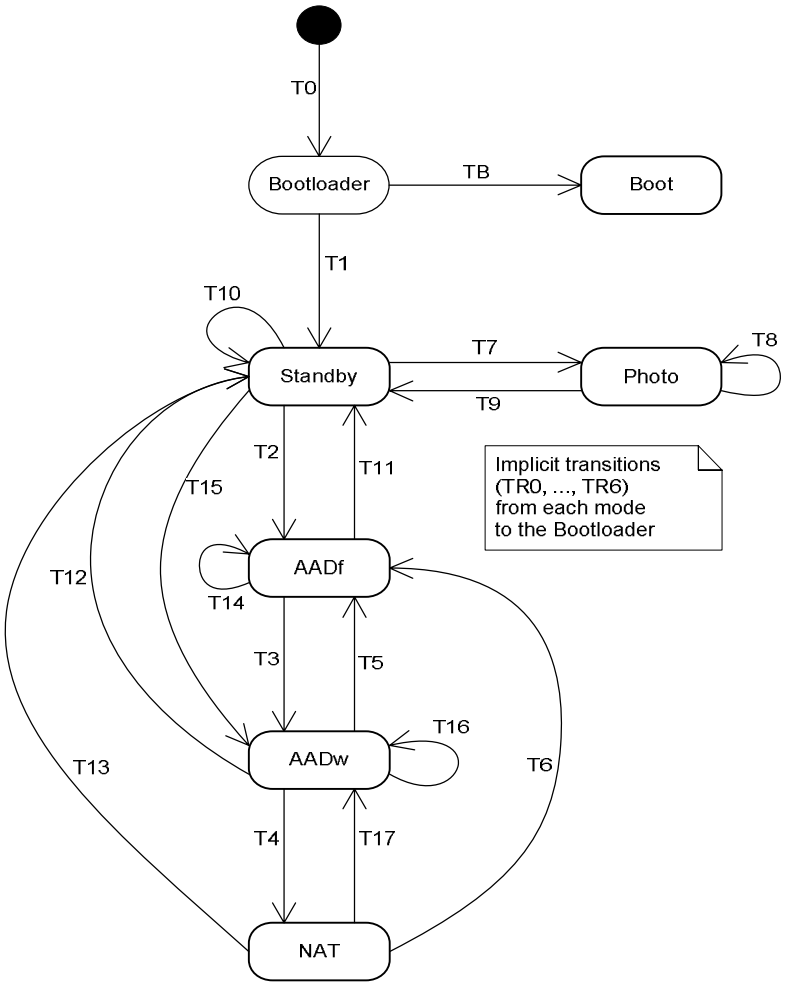

Figure 2-1 ASTRO APS Mode Switching Diagram

A complete and detailed description of the STR modes and transitions can be found in the STR Application SW and STR Boot mode S/W design documents (GS2.DD.JOP.STR.03401, GS2.DD.JOP.STR.03501). However, to understand the mode dependencies of the STR TC and TM data described by this PUS ICD a short overview is given by the following paragraphs.

After power-on (T0) or system reset (either commanded or caused by a fatal error), the STR executes the Boot Loader. The Boot Loader checks the PROM, the EEPROM and the RAM and

then, depending on the check results and on the state of the `startMode` variable (located in the SGM area in RAM) the Boot Loader performs one of the following actions:

- load the BOOT mode handler from ROM into the runtime area in RAM and start it
- load the STR application software (incl. the calibration data) from EEPROM into the runtime area in RAM and start it
- (not shown in the mode diagram above) copy the test application from the reserved RAM area to the runtime area in RAM and start it (note that for this, to make sense, a test application must have been uploaded into the reserved RAM)

After a Power-On Reset the SGM area in RAM is still undefined. In this case but also in the case that the SGM is corrupt the Boot Loader tries to load the APSW from EEPROM. In all other cases the Boot Loader in the first instance uses the `startMode` value to decide which way to go. If in the case `startMode==`**`startApsw`** the Boot Loader detects that the EEPROM content is corrupt or if in the case `startMode==`**`startTestapp`** the Boot Loader detects that the test application in the reserved RAM area is corrupt the Boot Loader will act the same way as if `startMode==`**`startBootHdl`** was set and will load and start the BOOT mode handler.

The BOOT mode handler covers the following functionality, which is completely implemented in the BOOT-ROM and is **not** in-flight modifiable:

- Initialization of the MIL bus I/F
- **Minimal TC handler**: Accept a subset of commands which is appropriate for memory checkout and software recovery. These commands include: direct loading of the APSW via operational I/F to RAM, reprogramming of the EEPROM, checkout of all memory, modification of RAM
- **Minimal TM handler:** Delivery of housekeeping information referring to the memory check results, consistency checks and memory uploads.

To exit the BOOT mode handler a TC_RESET command has to be used. Especially there is no way to pass to STANDBY mode directly from the BOOT mode handler.

The APSW, after initialization, begins in the STANDBY mode and then automatically passes to the AAD mode, i.e. if the STANDBY mode was entered via reboot it will be treated as an intermediate mode. However, in all other cases, i.e. if the STANDBY mode was entered either via TC_STANDBY command (from AAD, NAT, or PHOTO mode) or automatically (from TC_PHOTO mode when the commanded photo operation is complete) the STANDBY mode will only be left via a TC_RESET, TC_ACQUIRE or a TC_PHOTO command.

In STANDBY mode as well as in all other APSW modes the nominal TC/TM handling via the operational interface will be performed.

As seen from the mode switching diagram, the unit can be commanded either in the operational branch, or in the PHOTO Mode. The PHOTO mode provides a sequentially scanned image via the

operational interface, i.e. the PHOTO mode can be used e.g. as camera mode or for maintenance if necessary.

The nominal operational branch consists of two modes: the AAD (autonomous attitude determination, with two submodes AADf and AADw) mode and the NAT (nominal attitude tracking) mode.

The AAD performs autonomously the full field search operation for stars, the star tracking, centroiding and initial star identification. The switch between the APS detector full frame read out and the windowing read out is controlled by the AAD. After successful star identification, attitude determination and verification an automatic switch to the NAT mode is performed. NAT is the nominal operating mode of the star tracker providing valid attitude quaternion and 3-axis rate measurements.

So, normally it needs no command from the S/C control computer after power-on to activate the STR for continuously providing attitude quaternion.

If tracking is lost in NAT mode for any reason the STR automatically falls back to AAD mode to recover.

2.2 STR COMMANDS

The STR supports three sorts of commands:

- commands without parameters (e.g. TC_STANDBY)
- commands with parameters having a fix length (e.g. TC_ACQUIRE)
- commands having a variable length (e.g. TC_LOAD_MEMORY)

The STR will process **up to 5 commands per 100ms STR cycle**. If more commands are received within the same STR cycle, the processing of these commands will be postponed. Processing here has the meaning that the next TC packet is extracted from the last received TC block, then checked for consistency and then either executed or, if it is a command of “long duration” type (e.g. memory dump), stored in a queue. The queue for long duration commands may hold up to 5 commands. Any further long duration command which exceeds this capacity will be rejected. No parallel execution of long duration commands is supported, i.e. the queued long duration commands will be executed in the order of their reception, one after the other. The abortion of long duration commands is supported as well as the flushing of the queue. Commands which are not of “long duration” type are executed in parallel to the execution of the long duration commands.

Each command understood by the STR is identified by a service and a subservice number contained in the data field header of the TC packet structure. For details of the TC packet structure, especially for the TC packet header and the TC Data Field Header (format, field descriptions, allowed values, a.s.o.) see AD1, section 1.5.

2.2.1 IMPACT OF STR MODE ON TELECOMMAND PROCESSING

In general, the STR works in internal cycles of 100ms each. However, in AADf mode the STR cycle duration is about 240 to 250ms (default, when black-level correction is activated). AADf mode STR cycles are not aligned with the internal 10Hz basic cycles (which are aligned with the external 1Hz pulses).

In consequence, telecommand processing is delayed correspondingly when the STR is in AADf mode. This means that the limit of “5 commands per STR cycle” mentioned in section 2.2 means “5 commands per 250ms” when the STR is in AADf mode. Furthermore, processing of long duration commands (like for instance dump and check commands) is slowed down in AADf mode.

In addition, the period for daba block reception (see section 4.3.1) is increased to 250ms in AADf mode: The STR will be able to receive and acknowledge up to 2 TC blocks within 250ms (rather than 100ms in other modes).

2.2.2 STR TELECOMMAND SUMMARY

Table 2-2 gives a summary of the telecommands understood by the STR. The table shows for each TC its command mnemonic, its class (long, short duration, …, see Table 2-1), the service and subservice values, the TC packet length and the STR modes in which the command is allowed. The TC packet length covers the 6-byte packet header, the 4-byte data field header, the application data field (may be empty) and the 2-byte error control field.

The length of the application data field of each TC packet may be obtained by subtracting 12 from the given packet length value.

For each variable length command (e.g. TC_LOAD_MEMORY) the TC packet length value given in the table shows a counter *n* to denote the length variability. For each such command the actual value of *n* is given by a dedicated command parameter preceding the variable length part. To fulfill the size restriction of max. 206 bytes defined for telecommand source packets by AD1, section 1.7, an upper limit which satisfies this restriction is given for each *n* in the 'TC packet length' column of the table. However, these upper length limit values given in the table are only recommendations for the S2/EC user. The STR itself does neither require nor check against these limit values and will accept TC packets of larger sizes as well; actually a maximum TC packet length of 2044 bytes is accepted by the STR (➔two 1kB data transfer blocks).

The detailed format of the individual STR commands incl. the true length limitations applied by the STR are given in chapter 6 of this document.

Each TC is associated to one of four classes as defined in Table 2-1. This class defines the execution duration type and the interaction of the TC with other TCs.

Abbreviation	Class	Description
S	Normal short duration	No impact on long duration TCs
SM	Mode change (short duration)	This is a short duration command which performs a mode change. If a long duration command of class L is pending (i.e. either currently executing or within the waiting queue of already received TCs), the SM command is be rejected with a TM(1,8) / FID_CMD_TEMPORARILY_INHIBITED error report (see section 10). If no class L TC is pending, any pending class LA command is aborted with a TM(1,8) / FID_LONGDUR_TC_ABORTED error report (see section 10) and the execution of the SM command is started.

Abbreviation	Class	Description
L	Long duration, blocking commands of class SM)	A pending TC of this class (i.e. either currently executing or within the waiting queue) inhibits the execution of short duration commands of class SM.
LA	Long duration, aborted by commands of class SM	Any pending TC of this class (i.e. either currently executing or within the waiting queue) is aborted in case of an uninhibited TC of class SM.

Table 2-1 TC Classes

The duration type “long” (i.e. class L or LA) does not necessarily mean that the execution of a command of this type might take a long time as, for example, with the TC_SELFTEST command. For instance, the “long” type is also used for those commands which allow the user to request that the total TM output generated by the command is delivered in small pieces.

TC name (mnemonic)	Class	(Service, Subservice)	TC packet Length [bytes]	BOOT	STANDBY	AAD	NAT	PHOTO
TC_ENABLE_HKPKT	S	(3,5)	13	-	X	X	X	X
TC_DISABLE_HKPKT	S	(3,6)	13	-	X	X	X	X
TC_ENABLE_DIAGPKT	S	(3,7)	13	-	X	X	X	X
TC_DISABLE_DIAGPKT	S	(3,8)	13	-	X	X	X	X
TC_SET_HKPKTRATE	S	(3,130)	15	-	X	X	X	X
TC_SET_DIAGPKTRATE	S	(3,131)	15	-	X	X	X	X
TC_DUMP_SIDSTATES	S	(3,128)	12	-	X	X	X	X
TC_REPORT_HKPKT	S	(3,136)	13	-	X	X	X	X
TC_ENABLE_EVENTS	S	(5,5)	13+n*2 (n=1..96[1])	X	X	X	X	X
TC_DISABLE_EVENTS	S	(5,6)	13+n*2 (n=1..96[1])	X	X	X	X	X
TC_DUMP_DISEVENTS	S	(5,133)	12	X	X	X	X	X
TC_LOAD_MEMORY	S	(6,2)	22+n (n=1..184[1])	X	X	-	-	-
TC_DUMP_MEMORY	LA	(6,5)	22	X	X	X	X	X
TC_CHECK_MEMORY	LA	(6,9)	22	X	X	X	X	X
TC_SELFTEST	LA	(8,1)	14	X	X	-	-	-
TC_MANAGE_LONGDURCMDS	S	(8,220)	14	X	X	X	X	X
TC_VERIFY_TIME_SYNC	S	(9,135)	12	X	X	X	X	X
TC_SELECT_SYNC_SOURCE	S	(9,136)	13	X	X	X	X	X
TC_PING	S	(17,1)	12	X	X	X	X	X
TC_STANDBY	SM	(220,1)	12	-	X	X	X	X
TC_ACQUIRE	SM	(220,2)	50	-	X	X	X	-
TC_PHOTO	SM	(220,3)	22	-	X	-	-	X
TC_RESET	SM	(220,4)	13	X	X	X	X	X
TC_LOAD_CALIBRATION1	S	(221,1)	116	-	X	-	-	-
TC_LOAD_CALIBRATION2	S	(221,2)	108	-	X	-	-	-
TC_LOAD_CALIBRATION3	S	(221,3)	192	-	X	-	-	-
TC_LOAD_CALIBRATION4	S	(221,4)	204	-	X	-	-	-
TC_LOAD_CALIBRATION5	S	(221,5)	204	-	X	-	-	-

[1] This is the maximum resulting from the limitations in AD1; the actual maximum is larger (see description of the service in section 6)

TC name (mnemonic)	Class	(Service, Subservice)	TC packet Length [bytes]	BOOT	STANDBY	AAD	NAT	PHOTO
TC_LOAD_CALIBRATION6	S	(221,6)	204	-	X	-	-	-
TC_LOAD_CALIBRATION7	S	(221,7)	204	-	X	-	-	-
TC_LOAD_PREPROCESSOR	S	(221,10)	28	-	X	-	-	-
TC_LOAD_ALGORITHM	S	(221,11)	128	-	X	-	-	-
TC_LOAD_GSC	S	(221,12)	14+n*16 (n=1..12[1])	-	X	-	-	-
TC_LOAD_GSC_INDICES	S	(221,13)	14+n*2 (n=1..96[1])	-	X	-	-	-
TC_DUMP_CALIBRATION	S	(221,20)	16	-	X	X	X	X
TC_DUMP_PREPROCESSOR	S	(221,21)	12	-	X	X	X	X
TC_DUMP_ALGORITHM	S	(221,22)	12	-	X	X	X	X
TC_DUMP_GSC	LA	(221,23)	18	-	X	X	X	X
TC_DUMP_GSC_INDICES	LA	(221,24)	15	-	X	X	X	X
TC_ENABLE_MEMREAD	S	(223,1)	16	X	X	X	X	X
TC_ENABLE_MEMWRITE	S	(223,2)	16	X	X	-	-	-
TC_DISABLE_MEMREAD	S	(223,3)	14	X	X	X	X	X
TC_DISABLE_MEMWRITE	S	(223,4)	14	X	X	-	-	-
TC_COPY_EE2TAPP	LA	(223,10)	12	X	X	-	-	-
TC_COPY_TAPP2EE	L	(223,11)	12	X	X	-	-	-
TC_ENABLE_TEMPERATURE	S	(224,1)	14	-	X	X	X	X
TC_PRECESSION	S	(224,4)	15	-	X[2]	X	X	X[2]
TC_ABERRATION	S	(224,5)	19	-	X[2]	X	X	X[2]
TC_RATE_LIMIT	S	(224,6)	13	-	X[2]	X	X	X[2]
TC_TEMPERATURE	S	(224,7)	14	-	X	X	X	X
TC_ZERO_COUNTERS	S	(224,8)	13	X	X	X	X	X
TC_TIME_SHIFT_THRESHOLD	S	(224,9)	13	X	X	X	X	X
TC_MAX_DUMPSIZE	S	(224,10)	14	X	X	X	X	X

Table 2-2 STR Telecommand Summary

Note: The set of services/subservices implemented in the STR deviates from the “PUS Generic Service/Subservice Allocation” specified in section B3 of AD1. See AD5 for details.

[2] Command is accepted in this mode but without immediate effect

2.3 STR TELEMETRY

The STR supports two types of telemetry: periodic and non-periodic. Periodic telemetry covers all output data which are generated / updated periodically by the STR. Non-periodic telemetry covers all output data which are only generated on request, i.e. as a response on a TC, or to indicate progress and error events.

The stream of TM packets generated by the STR is inserted into Acquisition Data Blocks which are delivered to the OBC using the MIL bus data block transfer protocol which is described in chapter 4.3. TM_ADB is always inserted in front of all other periodic telemetry of the same cycle (TM_SDB, TM_TDB, TM_EDB, and the various TM_PDBs).

2.3.1 IMPACT OF STR MODE ON TELEMETRY UPDATE RATES

In general, telemetry update rates specified in this document (for instance in Table 2-3 or in chapter 7) reflect those in operational modes other than AADf, i.e. in modes with a nominal STR cycle duration of 100ms.

In AADf mode, the STR cycle duration is about 240 to 250ms (default, when black-level correction is activated). AADf mode STR cycles are not aligned with the internal 10Hz basic cycles (which are aligned with the external 1Hz pulses).

Actual update rates of telemetry packets will typically be significantly lower than specified in this document when the STR is in AADf mode. Especially when 10Hz update rates are specified, this rate decreases to some 4Hz when the STR is in AADf mode. This also applies to the rate of Acquisition Data Block generation (see also section 4.3.2).

2.3.2 STR TELEMETRY SUMMARY

The following table gives a summary of all TM packets maintained/provided by the STR. The table shows for each TM packet type its name (mnemonic), its packet category, service and subservice, associated structure ID value (where applicable), the TM packet length and the STR modes in which the TM packet may be delivered by the STR. The TM packet length value given in the table below covers the 6-byte packet header, the 12-byte data field header, the packet source data field and the 2-byte error control field. For details of the TM packet structure, especially for the TM source packet header and the TM Data Field Header (format, field descriptions, allowed values, a.s.o.) see AD1, section 2.3.

For variable length TM packets (e.g. TM_MEM_DUMP) the TM packet length value given in the table shows a counter *n* which denotes the length variability.

The counter *n* which is used in the length column of the table

- is taken or derived from a corresponding count parameter provided with the associated dump TC mentioned in the last column of the table (e.g. count parameter for TM_GSC_DUMP is derived from count parameter in TC_DUMP_GSC) or
- is taken from a default value which can be overwritten by the corresponding parameter set command (e.g. count parameter for TM_MEM_DUMP can be overwritten by TC_MAX_DUMPSIZE).

Thus the output amount produced by each of these commands is under control of the command generation side, i.e. if bandwidth is a concern, the count parameter provided in the command has to be set appropriately.

The detailed source data format of the individual TM packets may be found in chapter 7 of this document.

TM name (mnemonic)	PCAT	(Service/ Subservice)	SID (dec)	TM packet Length [bytes]	BOOT	STBY	AAD	NAT	PHOT	Trigger[1]
TM_ACK_VERISUCC	1	(1,1)	-	24	X	X	X	X	X	any TC
TM_ACK_VERIFAIL	1	(1,2)	-	26+k (k depends on FID)	X	X	X	X	X	any TC (for FIDs see §10)
TM_ACK_EXECSUCC	1	(1,7)	-	24	X	X	X	X	X	any TC
TM_ACK_EXECFAIL	1	(1,8)	-	26+k (k depends on FID)	X	X	X	X	X	any TC (for FIDs see §10)
TM_SDB	4	(3,25)	1	65	-	X	X	X	X	periodic, default: enabled, 1Hz (max: 10Hz)
TM_ADB	6	(3,25)	105	59	-	-	X	X	-	periodic, default: enabled, 10Hz (max: 10Hz)
TM_TDB	6	(3,25)	106	167	-	-	X	X	-	periodic, default: disabled, 1Hz (max: 10Hz)
TM_EDB	2	(3,26)	128	443	-	X	X	X	X	periodic, default: disabled, 1Hz (max: 10Hz)
TM_PDB	2	(3,26)	188..203	363	-	-	X	X	X	Periodic, default: disabled, 1Hz (max: 10Hz)
TM_SIDSTATES_DUMP	3	(3,129)	-	101	-	X	X	X	X	TC_DUMP_SIDSTATES
TM_EVENT	7	(5,1)…(5,4)	-	22+k (k depends on EID)	X	X	X	X	X	on any event cause (for EIDs see §9)
TM_DISEVENTS_DUMP	3	(5,134)	-	22+n*2 (n≥0)	X	X	X	X	X	TC_DUMP_DISEVENTS
TM_MEM_DUMP	9	(6,6)	-	30+n (n≥1)	X	X	X	X	X	TC_DUMP_MEMORY
TM_MEM_CHECK	3	(6,10)	-	32	X	X	X	X	X	TC_CHECK_MEMORY
TM_PING	1	(17,2)	-	20	X	X	X	X	X	TC_PING
TM_CALIBRATION_DUMP	3	(221,30)	-	24+n (n=1..1148)	-	X	X	X	X	TC_DUMP_CALIBRATION
TM_PREPROCESSOR_DUMP	3	(221,31)	-	36	-	X	X	X	X	TC_DUMP_PREPROCESSOR
TM_ALGORITHM_DUMP	3	(221,32)	-	136	-	X	X	X	X	TC_DUMP_ALGORITHM
TM_GSC_DUMP	3	(221,33)	-	24+n*16 (n=1..4000)	-	X	X	X	X	TC_DUMP_GSC
TM_GSC_INDICES_DUMP	3	(221,34)	-	22+n*2 (n=1..216)	-	X	X	X	X	TC_DUMP_GSC_INDICES

Table 2-3 STR Telemetry Summary

Note: The set of services/subservices implemented in the STR deviates from the “PUS Generic Service/Subservice Allocation” specified in section B3 of AD1. See AD5 for details.

[1] See also information in section 2.3.1 on telemetry update rates

3 DATA FIELD DESCRIPTIONS

3.1 LITERALS

If not stated otherwise, numeric literals denote numbers in base-10 (decimal) notation. A prefix of `0x` means base-16 (hexadecimal) notation.

3.2 BIT NUMBERING

The bits of a data unit of 'n' bits are numbered from 0 for the MSB to n-1 for the LSB. For instance, in a 32 bit unsigned integer, bit 0 has value 2^{31} and bit 31 has value 2^0.

3.3 DATA TYPES (FORMAT, LENGTH)

The following subsections define the data types and their notation used within this document.

3.3.1 BOOLEAN – B

The Boolean format is assigned to fields with a length of 1 bit and is mainly used for flags. If not explicitly noted otherwise the flag is active if set to 1.

The Boolean type may also be used for spare (unused/reserved) fields of any size.

3.3.2 ENUMERATION – E

This format indicates an enumeration which is located in 'n' bits. See section 11 for possible values of the corresponding enumeration.

3.3.3 UNSIGNED INTEGER – UI

This format indicates an **unsigned integer number** which is located in 'n' bits. The range of a number of the format "U*n*" is 0 … 2^n–1.

3.3.4 SIGNED INTEGER – SI

This format indicates a **signed integer number** which is located in 'n' bits using **two's complement** encoding. The range of a number of the format "S*n*" is -2^{n-1} … $+2^{n-1}$–1.

3.3.5 FLOATING-POINT NUMBER – F

The IEEE Standard 754/854 defines a 32-bit single-precision floating point format:

Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	Bit 8	Bit 9	Bit 10	Bit 11	Bit …	Bit 29	Bit 30	Bit 31
s	e_7	e_6	e_5	e_4	e_3	e_2	e_1	e_0	f_{22}	f_{21}	f_{20}	…	f_2	f_1	f_0

The unsigned exponent e can range between $1 \le e \le 254$ for numbers in the single precision format. This exponent is biased by +127. To calculate the true unbiased exponent, 127 must be subtracted from e.

A number in this format consists of a sign bit s, a 24-bit significant, and an 8-bit unsigned-magnitude exponent e. For normalized numbers, the significant consists of a 23-bit fraction f and a "hidden" bit of 1 that is implicitly presumed to precede f_{22} in the significant. The binary point is presumed to lie between this hidden bit and f_{22}. The LSB of the fraction is f_0; the LSB of the exponent is e_0. The hidden bit effectively increases the precision of the floating point significant to 24 bits from the 23 bits actually stored in the data format. It also ensures that the significant of any number in the IEEE normalized number format is always greater than or equal to 1 and less than 2.

Note, that the IEEE Standard also provides the following special data types in the single-precision floating point format.

- An exponent value of 255 (all ones) with a nonzero fraction is a NAN. NANs are usually used as flags for data flow control, for the values of uninitialized variables, and for the results of invalid operations as 0•∞.
- Infinity is represented as an exponent of 255 and a zero fraction. Note that because the fraction is signed, both positive and negative Infinity can be represented.
- Zero is represented by a zero exponent and a zero fraction. As with infinity, both positive Zero and negative Zero can be represented.

3.3.6 TIME STAMP – TS

This format indicates a time stamp as used in the PUS protocol layer.

The STR receives and reports time stamps in the following format:

Bit 0–55
56-bit unsigned integer CUC-time-Format with b_0…b_{31} = seconds part, b_{32}..b_{55} = subseconds part (➔unit 2^{-24}s)

3.3.7 STR APPLICATION TIME STAMP – T

This format indicates a time stamp as used by the STR within its application data structures.

It is identical to the TS format (see above) but with the least significant byte (bits b_{48}..b_{55}) omitted.

3.4 CRC IMPLEMENTATION

The STR uses the polynomial $x^{16} + x^{12} + x^5 + 1$ (CCITT CRC16) with an initial value of `0xffff` for CRC verification and generation and is thus fully compliant to AD1, section B3.4.

3.5 CONVERSION FROM RAW TO ENGINEERING VALUE

If the transmitted raw data value is not equivalent to the engineering value, i.e. if the value of the LSB is not 1, a conversion is necessary. To convert the transmitted raw data to the equivalent engineering value it is necessary to multiply the raw data with the given LSB value.

4 MIL BUS

4.1 TERMINAL CONFIGURATION COMMANDS

According to AD3, section 4.7.5.2.2, the STR uses SA01R to receive Terminal Configuration Commands from the OBC.

Only the Reset_RT_Health word, which is the first word (16 bit) of this message, will be regarded by the STR.

The purpose of the Reset_RT_Health word is to request the STR to clear (reset) specified bits of its RT_Health (word 0 of the STR's health status message) by ANDing the RT_Health word with the 1's complement of the received Reset_RT_Health word (the terminal configuration command word specifies a reset mask for the RT_Health word).

4.2 HEALTH STATUS MESSAGE

According to AD3, section 4.7.5.2, the STR uses SA01T to store its health status message. As far as there are status information (flags, counters, a.s.o.) which have changed since the last update this message will be updated once every 100ms STR cycle.

One exception to this automatic update performed by the STR is the RT_Health word which is the first word of the health status message. The status bits in this word may be set by the STR at any time but may only be reset by the Reset_RT_Health word of a terminal configuration command.

If the STR is in BOOT mode the health message is especially important for the OBC side because in BOOT mode no periodic HK TM packets are generated by the STR. However, the health message will be maintained in any STR mode.

The detailed structure of the STR's health status message is given below.

The detailed structure of the MIL bus health status message maintained by the STR is shown by the following table. The offsets are counted in bits starting with 0 for the msb of the first health message word:

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
	word 0 (RT_Health)					
initializationCompleted	Set after completion of the initialization of the RT	0	B	1		
initializationFailure	Set after a failure during the initialization of the HW or SW	1	B	1		
hwTestFailure	Set in case of detection of hardware failure	2	B	1		
swFailure	Set in case of detection of a software failure	3	B	1		
realTimeExecutionFailure	Set in case of detection of a violation of a real-time constraint (see section 9.32)	4	B	1		
watchDogSet	Set in case of a watch dog setting	5	B	1		
sensorFailure	not used, will always be set to 0	6	B	1		
secondaryVoltageFailure	not used, will always be set to 0	7	B	1		
ioFailure	not used, will always be set to 0	8	B	1		
internalIfFailure	Set in case of TM output overflow	9	B	1		
temporaryFailureOnData	Set in case of detection of corrupt DBT receptions	10	B	1		
rtNotSynchronized	Set when errors are detected with time synchronization	11	B	1		

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
reserved	set to 0	12	B	4		
	word 1					
minorFrameNumber	The latest received minor frame number	16	UI	16		
	word 2, …, 14 (complementary information)					
cycleNumber	Current STR cycle number	32	UI	16		
opMode	Current operational mode	48	E	3	See OpMode enumeration	
cycleTriggerSource	Current trigger source; 0: timer; 1: preprocessor; 2: external	51	E	2		
spare	set to 0	53	B	3		
bootStorageError	Set when the CRC of the boot mode storage (in PROM) has failed	56	B	1		
applicationStorageError	Set when the CRC of the application storage (in EEPROM) has failed	57	B	1		
calibrationStorageError	Set when the CRC of the calibration storage (in EEPROM) has failed	58	B	1		
testApplicationStorageError	Set when the CRC of the test application storage (in RAM) has failed	59	B	1		

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
testCalibrationStorageError	Set when the CRC of the test calibration storage (in RAM) has failed	60	B	1		
spare	set to 0	61	B	3		
latestRamErrorLocation	Address of the latest RAM error (1 or more failed bits)	64	UI	32		
latestTrapLocation	Location of the latest trapped instruction	96	UI	32		
accumulatedNumberOfRamErrors	Counter for detected 1-bit RAM errors	128	UI	8		
latestTrapNumber	Identification number of the latest CPU trap	136	UI	8		
latestDeterminedTimeDeviation	Deviation in ms between LOBT and received OBT	144	SI	8		
secondsSinceLatestSync	Number of seconds since latest time synchronization	152	UI	8		
temperatureDetector	Detector temperature (set to 0x8000 in Boot mode)	160	SI	16		10^{-1} centigrade
temperatureOptics	Optics temperature (set to 0x8000 in Boot mode)	176	SI	16		10^{-1} centigrade
temperatureHousing	Housing temperature (set to 0x8000 in Boot mode)	192	SI	16		10^{-1} centigrade
currentTecPwmSetting	Setting of the pulse width modulator for the TEC	208	UI	8		

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
accumulatedNumberOfTcErrors	Counter for detected telecommand errors	216	UI	8		
spare	set to 0	224	B	16		
	word 15, …, 31 (configuration monitoring data)					
swVersion	Software version	240	UI	16		
startMode	Start mode specified in latest TC_RESET	256	E	2	See StartMode enumeration	
syncSource	Selected PPS line	258	E	2	See SyncSource enumeration	
tecMode	Thermo electric cooler mode	260	E	2	See TecMode enumeration	
spare	set to 0	262	B	2		
timeShiftThreshold	time shift threshold	264	UI	8		
targetTemperature	Current target temperature	272	SI	16		10^{-1} centigrade
readEnaState_MemID_A3	Read Enable state of the ASIC registers area (memory ID 0xA3; see section 13)	288	B	1	0 – read disabled 1 – read enabled	
readEnaState_MemID_B3	Read Enable state of the RT controller registers area (memory ID 0xB3; see section 13)	289	B	1	0 – read disabled 1 – read enabled	

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
readEnaState_MemID_C3	Read Enable state of the RT controller RAM area (memory ID 0xC3; see section 13)	290	B	1	0 – read disabled 1 – read enabled	
readEnaState_MemID_D3	Read Enable state of the Processor registers area (memory ID 0xD3; see section 13)	291	B	1	0 – read disabled 1 – read enabled	
writeEnaState_MemID_A3	Write Enable state of the ASIC registers area (memory ID 0xA3; see section 13)	292	B	1	0 – write disabled 1 – write enabled	
writeEnaState_MemID_B3	Write Enable state of the RT controller registers area (memory ID 0xB3; see section 13)	293	B	1	0 – write disabled 1 – write enabled	
writeEnaState_MemID_C3	Write Enable state of the RT controller RAM area (memory ID 0xC3; see section 13)	294	B	1	0 – write disabled 1 – write enabled	
writeEnaState_MemID_D3	Write Enable state of the Processor registers area (memory ID 0xD3; see section 13)	295	B	1	0 – write disabled 1 – write enabled	
writeEnaState_MemID_11	Write Enable state of the APSW area in EEPROM (memory ID 0x11; see section 13)	296	B	1	0 – write disabled 1 – write enabled	
writeEnaState_MemID_21	Write Enable state of the APSW runtime area in RAM (memory ID 0x21; see section 13)	297	B	1	0 – write disabled 1 – write enabled	
writeEnaState_MemID_31	Write Enable state of the SGM area in RAM (memory ID 0x31; see section 13)	298	B	1	0 – write disabled 1 – write enabled	
writeEnaState_MemID_51	Write Enable state of the Calibration data area in EEPROM (memory ID 0x51; see section 13)	299	B	1	0 – write disabled 1 – write enabled	

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
writeEnaState_MemID_71	Write Enable state of the Test Application upload area in RAM (memory ID 0x71; see section 13)	300	B	1	0 – write disabled 1 – write enabled	
writeEnaState_MemID_81	Write Enable state of the TestApp calib data upload area in RAM (memory ID 0x81; see section 13)	301	B	1	0 – write disabled 1 – write enabled	
writeEnaState_MemID_91	Write Enable state of the Communication failure buffer in RAM (memory ID 0x91; see section 13)	302	B	1	0 – write disabled 1 – write enabled	
spare	set to 0	303	B	17		
maxDumpSizeInBytes	amount per dump packet	320	UI	16		
spare	set to 0	336	B	176		
	Overall length			512		

4.3 DATA BLOCK TRANSFER PROTOCOL

For TC packet reception and TM packet delivery the STR implements the MIL-bus data block transfer protocol as specified in AD3, sections 4.7.2 and 4.8.

4.3.1 TC PACKET RECEPTION

The stream of TC packets addressed to the STR will be split by the OBC into a stream of blocks. Any such block will be delivered by the OBC to the STR by sending the block via MIL-bus to SA11R up to SA26R and then sending a corresponding Distribution Transfer Descriptor (DTD) to SA27R.

The STR's transport layer at first checks the DTD and, if a new block is indicated (the block number in the DTD has changed), checks the received block against the length specified in the DTD. From the check results the transport layer of the STR generates an appropriate Distribution Transfer Confirmation (DTC), stores this DTC in SA27T, and, if the checks were successful, forwards the new TC data block and the length of the block to the STR's session layer.

Note that the OBC has to wait for the DTC before it sends the next TC data block and DTD.

The first header pointer contained in each block header will be used by the STR's session layer to extract (and reassemble) TC packets from the incoming stream of blocks and forward the extracted TC packets to the PUS service 1 handler of the STR where the TC packets will be checked, forwarded for execution and acknowledged accordingly (see section 5.3).

The STR MIL bus driver will be able to receive and confirm up to 2 TC blocks, each with up to 1kB, from OBC per 100ms STR cycle. If a third block incomes within an 100ms cycle the processing and confirmation of this block will be postponed to the next STR cycle. See also section 2.2.1 on the impact of STR mode on data block reception.

4.3.2 TM PACKET DELIVERY

The STR's session layer splits the stream of TM source packets generated by the STR into blocks of up to 1022 bytes, prepends a 2-byte block header to each block and forwards these blocks to the STR's transport layer.

The transport layer delivers each TM block to the OBC by performing the following 3 steps in order:

- store the block in SA11T up to SA26T
- generate a corresponding Acquisition Transfer Request (ATR) and store it in SA28T
- wait for an Acquisition Transfer Confirmation (ATC) from the OBC; the ATC is expected in SA28R

As long as the ATC is outstanding, no other TM block may be provided by the STR. If the time the STR waits for an outstanding ATC exceeds 600ms the STR discards the TM data, requests a reset of the block protocol (see section 4.3.3) and generates an appropriate error event.

The STR's MIL bus driver will be able to provide up to 8 acquisition data blocks, each of up to 1kB, per 100ms STR cycle to the OBC.

However, because the OBC will accept only one maximum block per MIL-bus minor frame from the STR (see AD3, req. MIL-355), i.e. up to 1kB every 50ms, the TM output to be transferred to the OBC will be restricted to 2kB per 100ms STR cycle. If more than 1kB are to be transferred to the OBC, the next ATR will be issued within 5ms after reception of the ATC. Any TM packet generated by the STR which exceeds the 2kB limit will be discarded and a EV_TM_OVERFLOW event will be generated (will be sent in the next STR cycle).

4.3.3 Generation of a Block Protocol Reset request by the STR

The STR requests a protocol reset if the RT has been reset (RT initialization) or if a protocol error was detected by the STR which requires a protocol reset (e.g. timeout in waiting for ATC).

If the last sent ATR had a block number > 0 (i.e. the last ATR was not a protocol reset) the STR generates and sends an ATR completely cleared except Reset flag set to 1.

The STR awaits for an ATC with Reset=1 (all other data in this ATC should be 0) within the time span as specified in section 4.3.2. This ATC indicates that the OBC grants the protocol reset. The STR saves the current time of the ATC reception

After reception of ATC from OBC the next ATR generated by the STR will have block number 1.

Furthermore as long as the sensor shows an ATR with set reset flag (whether confirmed by ATC or not) it does not perform a new reset.

4.3.4 Reception of a Block Protocol Reset request by the STR

The OBC generates and sends a DTD completely cleared except Reset flag set to 1.

After reception of this DTD the STR checks that

- the last sent ATR had a block number > 0 (i.e. the last ATR was not a protocol reset)
- the last valid ATC with Reset=1 was received more than 1200 ms ago

If the checks are successful the STR initiates a protocol reset as described in 4.3.3. Otherwise the STR does not initiate a protocol reset.

The STR generates and sends a DTC completely cleared except Reset set to 1 to indicate that the protocol reset has been carried out.

The next ATR generated by the STR will have block number 1.

4.4 MODE COMMANDS

The STR supports the mode commands listed in Table 4-1. All other mode commands are considered illegal.

Mode commands can be sent to either sub-address 0 or sub-address 31.

Mode Code	T/R	Broadcast	Function	Comment
00010	T	illegal	Transmit status word	
00011	T	allowed	Initiate selftest	
00100	T	allowed	Transmitter shutdown	
00101	T	allowed	Override transmitter shutdown	
01000	T	allowed	Reset remote terminal	
10001	R	required	Synchronize with data word	Reception of minor frame number
10010	T	illegal	Transmit last command	
10011	T	illegal	Transmit BIT word	

Table 4-1 Supported Mode Commands

The “Broadcast” column in Table 4-1 defines whether the corresponding mode command is supported when the remote terminal address is set to 31 in the command word. An entry “illegal” means that the corresponding mode command is only accepted when sent explicitly to the remote terminal address of the STR. An entry “required” means that the corresponding mode command is only accepted when sent to the broadcast address. An entry “allowed” means that the corresponding mode command is accepted when sent either to the own address of the STR or to the broadcast address.

4.5 LEGAL MESSAGE SUMMARY

The STR considers the messages listed in Table 4-2 legal. Messages not listed in Table 4-2 are considered illegal. With the excpetion of OBT reception, any other broadcast message in the sub-address range from 1 to 30 is considered illegal.

Sub-address	T/R	Word Count	Comment
0	*	*	Mode Commands (see Table 4-1)
1	R	1–32	Terminal Configuration Command (see section 4.2)
1	T	1–32	Health and Status Message (see section 4.2)

Sub-address	T/R	Word Count	Comment
11–26	R	1–32	Data block reception (see section 4.3)
11–26	T	1–32	Data block transmission (see section 4.3)
27	R	2	DTD (see section 4.3)
27	T	2	DTC (see section 4.3)
28	T	2	ATR (see section 4.3)
28	R	2	ATC (see section 4.3)
29	R	5	OBT reception (see section 5.8); also legal as broadcast
30	R	1–32	Data Wrap-around reception
30	T	1–32	Data Wrap-around transmission
31	*	*	Mode Commands (see Table 4-1)

Table 4-2 Legal Messages

5 PUS PROTOCOL IMPLEMENTATION

Note that all PUS services which are not mentioned in this chapter are not supported by the STR.

5.1 PRID NUMBERS USED BY THE STR

According to the PRID allocation table given in AD3, section B2.1, the STRs mounted on the satellite will use the following different PRID values:

PRID	used by
0x25	STR-1
0x26	STR-2
0x27	STR-3
0x29	fall-back (see below)

To achieve this, each specific STR must be able to detect which of the 3 STR roles (STR-1 or STR-2 or STR-3) it shall play. This role assignment will be derived from the remote terminal address configured to the STR. For interchangeability of the sensors (at least in the frame of a specific satellite) a dedicated Look-up table will be stored in ROM mapping the 3 STR PRIDs to the 3 RT addresses currently reserved for the 3 STRs (see AD3, Table 4.6-1). Thus, knowing the RT-Address configured to an STR, the PRID to be used by this STR will be obtained from the Look-up table in ROM.

If the STR during bootup detects that the RT-address currently configured to it by the OBC is none of those listed in the STR's lookup table the STR will use PRID value 0x29. With this wrong PRID value the STR will then reject any incoming telecommand with a TM(1,2) packet reporting the fault identifier FID_UNKNOWN_PRID (note that if one assumes that any incoming TC is valid at least wrt. the PRID value contained in its packet header this PRID can never match that wrong (fake) one which is currently used by the STR).

The behaviour of the STR in case of a non-matching RT-address as proposed above is agreed with the customer.

5.2 SYNCHRONIZATION OF LOCAL ON-BOARD TIME

Time synchronization by the OBC is performed by sending a new time value to the STR (via MIL-Bus Time-Broadcast to SA29R). The new provided COBT will become valid, i.e. will become the new LOBT, upon reception of the next external synchronization pulse (1pps pulse).

- If no synchronization is configured (i.e. in Boot mode) or the synchronization fails or is lost, the STR is incrementing the time locally.
- If the initial synchronisation is not configured (i.e. in Boot mode) or fails, the COBT available on SA29R will be used to initialize the LOBT of the STR. After that the time will be incremented locally.
- If the initial synchronization is not configured (i.e. in Boot mode) or fails and no COBT is available on SA29R, the STR will use the time elapsed in the STR since last boot and the TimeSync/Quality field of the DFH will be set to indicate 'SCET'.

The STR uses

- LOBT time stamps, flagged "synchronized"
  - if the synchronization is successful
  - if a re-synchronization is successful after a lost synchronization
- LOBT time stamps, flagged "not synchronized"
  - if the initial synchronization was not successful but COBT has already been received on SA29R
  - if a synchronization after startup was not required but COBT is already received on SA29R
  - if the synchronization is lost later on
- SCET time stamps, flagged "not synchronized"
  - if the synchronization was not successful and no COBT had been received on SA29R
  - if a synchronization after startup was not required and no COBT had been received on SA29R

Note: In some cases, the STR does not use *synchronized* LOBT for time stamping. This deviates from the timing requirements in AD2. See AD6 for details.

Initialization BMSW and ASW

In certain error situations during initialization, the following subset of events can be generated:

- EV_DBT_TIMEOUT - DBT timeout
- EV_WATCHDOG_RESET - watchdog timeout

- EV_TRAP_MAJOR - major trap
- EV_RAM_ERROR - RAM error
- EV_BAD_APSW - check of ASW in EEPROM failed
- EV_BAD_TESTAPP - check of Test Application S/W failed

These events are generated with:

- LOBT time stamps, flagged "not synchronized" if COBT has already been received on SA29R
- SCET time stamps, flagged "not synchronized" if no COBT had been received on SA29R

BMSW

The BMSW does not perform an automatic synchronization to LOBT.

Entering Boot mode is delayed by one second. Thus, it is ensured that the EV_OPMODE_BOOT event is sent with LOBT time stamp, flagged "not synchronized" (provided OBT has been received within that time span on SA29R).

The synchronization to LOBT can be requested by the TC_SELECT_SYNC_SOURCE command. Telemetry created after the successful synchronization will contain synchronized LOBT .

ASW

The ASW performs an automatic synchronization to LOBT after startup.

Entering Standby mode (i.e. nominal STR cycle handling) will be delayed until a synchronization result is available (either successful or failed synchronization). The delay is at maximum 3 seconds. This delays

- the generation of periodically generated telemetry
- command handling
- the generation of mode change events.

Thus, all periodically generated TM, TM generated in response to TC, and mode related events are generated with synchronized LOBT time stamps (provided synchronization was successful).

5.3 PUS SERVICE 1 – TC VERIFICATION

Incoming TC packets will be checked and acknowledged according AD1, section 4.1.

The following TC acknowledge packets are supported by the STR:

- TM(1,1) – denotes the TM_ACK_VERISUCC packet (see section 7.1.1) which will be generated if all static and consistency checks performed on the TC packet have been passed and if the acceptance ACK flag in the DFH of the TC packet was set.
- TM(1,2) – denotes the TM_ACK_VERIFAIL packet (see section 7.1.2) which will be generated, regardless of the ACK flags settings in the DFH of the TC packet, if any of the static and consistency checks on the TC packet failed
- TM(1,7) – denotes the TM_ACK_EXECSUCC packet (see section 7.1.3) which will be generated if the TC packet was accepted and the command execution was successful and the execution ACK flag in the DFH of the TC packet was set
- TM(1,8) – denotes the TM_ACK_EXECFAIL packet (see section 7.1.4) which will be generated, regardless of the ACK flags settings in the DFH of the TC packet, if the TC packet was accepted and the command execution failed

For the FID fields in the TM(1,2) and TM(1,8) packets the predefined FID values listed in AD1, section B2.8, will be used as far as applicable to the specific error case to be reported.

Other TM(1,x) packets will not be generated by the STR.

For a complete list of the TC acceptance and execution errors reported by the STR see section 10.

5.4 PUS SERVICE 3 – HK REPORTING

This service will be used by the STR for reporting of periodically generated S/C ancillary data (PCAT=6), status/HK data (PCAT=4), and diagnostic data (PCAT=2).

The following subservices are supported:

- TC(3,5): denotes the TC_ENABLE_HKPKT command described in section 6.1.1
- TC(3,6): denotes the TC_DISABLE_HKPKT command described in section 6.1.2
- TC(3,7): denotes the TC_ENABLE_DIAGPKT command described in section 6.1.3
- TC(3,8): denotes the TC_DISABLE_DIAGPKT command described in section 6.1.4
- TM(3,25): HK report
- TM(3,26): Diag report
- TC(3,130): denotes the TC_SET_HKPKTRATE command described in section 6.1.6
- TC(3,131): denotes the TC_SET_DIAGPKTRATE command described in section 6.1.7
- TC(3,128): denotes the TC_DUMP_SIDSTATES command described in section 6.1.5

- TM(3,129): denotes the TM_SIDSTATES_DUMP report described in section 7.2.6
- TC(3,136): denotes the TC_REPORT_HKPKT command described in section 6.1.8

Other TC(3,x) commands and TM(3,x) reports are not supported by the STR. Especially, as agreed during the PDR, all service 3 TCs which relate to the definition of new HK or diagnostic SIDs and also the reporting of existing SIDs definitions are not supported (see the panel disposition 3 for COM_SW-OPS-14).

If enabled or requested by a TC(3,136) command the TM_SDB packet (Status and Health Data Block, see section 7.2.2) will be delivered in the form of a TM(3,25) packet of category 4 (HK TM) with SID=1.

If enabled or requested by a TC(3,136) command the TM_ADB packet (Attitude Data Block, see section 7.2.1) will be delivered in the form of a TM(3,25) packet of category 6 (S/C ancillary data) with SID=105.

If enabled or requested by a TC(3,136) command the TM_TDB packet (Tracker Data Block, see section 7.2.3) will be delivered in the form of a TM(3,25) packet of category 6 (S/C ancillary data) with SID=106.

If enabled or requested by a TC(3,136) command the TM_EDB packet (Engineering Data Block, see section 7.2.5) will be delivered in the form of a TM(3,26) packet of category 2 (Diagnostic data) with SID=128.

If enabled or requested by a TC(3,136) command the TM_PDB packets (Pixel Data Blocks, see section 7.2.4) will be delivered in the form of TM(3,26) packets of category 2 (Diagnostic data) with SIDs 188…203.

5.5 PUS SERVICE 5 – EVENT REPORTING

This service will be used by the STR to indicate nominal and failure events to the OBC (and, of course, to the ground).

The following subservices are supported by the STR:

- TM(5,1): normal/progress event reports
- TM(5,2): low severity (warning) error event reports
- TM(5,3): medium severity error event reports
- TC(5,5): denotes the TC_ENABLE_EVENTS command described in section 6.2.1
- TC(5,6): denotes the TC_DISABLE_EVENTS command described in section 6.2.2
- TC(5,133): denotes the TC_DUMP_DISEVENTS command described in section 6.2.3
- TM(5,134): denotes the TM_DISEVENTS_DUMP report packet described in section 7.3.2

TM(5,1) reports will be used e.g.:

- to indicate automatic mode switches from lower to higher modes
- to indicate the success of all commanded mode switches
- to indicate the execution progress of long-duration commands

TM(5,2) reports will be used for warnings, e.g.:

- to indicate that a minor CPU trap occurred in the STR

TM(5,3) reports will be used e.g.:

- to indicate failures detected by the STR during a SELFTEST
- to indicate when the STR lost the attitude and switched back from NAT to AAD mode

For a complete list of all events reported by the STR see section 9.

5.6 PUS SERVICE 6 – MEMORY LOAD/DUMP/CHECK

The following service 6 TCs are supported by the STR:

- TC(6,2) denotes the TC_LOAD_MEMORY command
- TC(6,5) denotes the TC_DUMP_MEMORY command
- TC(6,9) denotes the TC_CHECK_MEMORY command

Other TC(6,x) commands will not be supported by the STR.

The command data structures of the supported 3 TCs are in full compliance to AD1, section 4.6. The details for these 3 TCs are given in sections 6.3.1, 6.3.2, 6.3.3.

The combination of *Start Address* and *Length* specified in the command packet will be checked to be valid for the *MemID* parameter specified with the command.

A TC(6,2) command will only be executed if the memory specified by the *MemID* parameter is writable and if writing to the associated memory has been enabled (see sections 5.12 and 13).

Depending on the *MemID* parameter specified with a TC(6,5) or TC(6,9) command the successful execution of these commands may require the enabling of memory read operations (see sections 5.12 and 13).

A TM(6,6) packet will be used to deliver the result of a TC(6,5) command.

A TM(6,10) packet will be used to deliver the result of a TC(6,9) command.

The application data structures of these TM packets are in full compliance to AD1, section 4.6. The details for these 2 TM packets are given in sections 7.4.1, and 7.4.2.

5.7 PUS SERVICE 8 – FUNCTION MANAGEMENT

Service 8 will be supported to execute the STR selftest command and to manage long duration commands.

The following service 8 TCs are supported by the STR:

- TC(8,1) shall be used to execute the TC_SELFTEST command (see section 6.4.1)
- TC(8,220) shall be used to execute the TC_MANAGE_LONGDURCMDS command (see section 6.4.2)

Other TC(8,x) commands are currently not supported by the STR.

5.8 PUS SERVICE 9 – TIME SYNCHRONIZATION

Time synchronization by the OBC is performed by sending a new time value to the STR (via MIL-Bus Time-Broadcast to SA29R). The new provided COBT will become valid, i.e. will become the new LOBT, upon reception of the next external synchronization pulse (1pps pulse).

For a more detailed description of time synchronization see section 5.2.

The following service 9 TCs are supported by the STR:

- TC(9,135) – denotes the TC_VERIFY_TIME_SYNC command described in section 6.5.1 to trigger the time synchronisation verification
- TC(9,136) – denotes the TC_SELECT_SYNC_SOURCE command described in section 6.5.2 used to select the synchronization source

Both commands are supported as required by AD1, section 4.8.

Acc. To AD1, section 4.8.7, the response on a TC(9,135) command will be a dedicated TM(5,1) event report of type EV_TIME_SYNC_SUCCESS if the synchronization was successful, and a TM(5,3) report of type EV_TIME_SYNC_FAILED if the synchronization failed.

Other TC(9,x) commands will not be supported by the STR.

Note: The command TC_TIME_SHIFT_THRESHOLD (described in section 6.10.7) is used to specifiy the threshold for the generation of event EV_TIME_SHIFT.

5.9 PUS SERVICE 17 – CONNECTION TEST

Service 17 will be supported as required by AD1, i.e. a TM(17,2) packet (see section 7.5.1) will be generated in response to a TC(17,1) command.

5.10 PUS SERVICE 220 – STR MODE MANAGEMENT

According to AD1, section B2.12.7.1, the STR specific service 220 is used for all STR mode management commands.

The following service 220 TCs are supported by the STR:

- TC(220,1) denotes the TC_STANDBY command described in section 6.7.1
- TC(220,2) denotes the TC_ACQUIRE command described in section 6.7.2
- TC(220,3) denotes the TC_PHOTO command described in section 6.7.3
- TC(220,4) denotes the TC_RESET command described in section 6.7.4

The TC_STANDBY command does not have any command parameters and thus the application data field of this command must be empty.

5.11 PUS SERVICE 221 – STR SPECIFIC TABLE LOAD/DUMP

The STR specific service 221 is used for STR specific table load/dump commands.

The following service 221 TCs are supported by the STR:

- TC(221,1),…, TC(221,7) denote the TC_LOAD_CALIBRATION1, …, TC_LOAD_CALIBRATION7 commands used to modify the STR calibration data structure
- TC(221,10) denotes the TC_LOAD_PREPROCESSOR command used to modify the STR preprocessor configuration parameters
- TC(221,11) denotes the TC_LOAD_ALGORITHM command used to modify the STR algorithm parameters
- TC(221,12) denotes the TC_LOAD_GSC command used to modify the Guide Star Catalog
- TC(221,13) denotes the TC_LOAD_GSC_INDICES command used to modify the GSC index list
- TC(221,20) denotes the TC_DUMP_CALIBRATION command used to dump the STR calibration data
- TC(221,21) denotes the TC_DUMP_PREPROCESSOR command used to dump the STR preprocessor configuration parameters
- TC(221,22) denotes the TC_DUMP_ALGORITHM command used to dump the STR algorithm parameters
- TC(221,23) denotes the TC_DUMP_GSC command used to dump a specified part of the Guide Star Catalog
- TC(221,24) denotes the TC_DUMP_GSC_INDICES command used to dump a specified part of the GSC index list

The TC_DUMP_PREPROCESSOR command is described in section 6.8.13.

The TC_DUMP_ALGORITHM command is described in section 6.8.14.

The TC_DUMP_CALIBRATION command is described in section 6.8.12.

The TC_DUMP_GSC command is described in section 6.8.15.

The TC_DUMP_GSC_INDICES command is described in section 6.8.16.

The TC_LOAD_PREPROCESSOR command is described in section 6.8.8.

The TC_LOAD_CALIBRATION*x* commands are described in sections 6.8.1, 6.8.2, 6.8.3, 6.8.4, 6.8.5, 6.8.6, 6.8.7. Note that because the calibration data structure does not fit in a single TC packet, several dedicated “partial upload” TCs have been defined.

The TC_LOAD_ALGORITHM command is described in section 6.8.9.

The TC_LOAD_GSC command is described in section 6.8.10.

The TC_LOAD_GSC_INDICES command is described in section 6.8.11.

The response on any of the table dump commands TC(221,20),…,TC(221,24) is delivered in the form of a dedicated TM(221,30),…,TM(221,34) packet:

- the response on a TC(221,20) command is a TM(221,30) packet described in section 7.6.1
- the response on a TC(221,21) command is a TM(221,31) packet described in section 7.6.2
- the response on a TC(221,22) command is a TM(221,32) packet described in section 7.6.3
- the response on a TC(221,23) command is a TM(221,33) packet described in section 7.6.4
- the response on a TC(221,24) command is a TM(221,34) packet described in section 7.6.5

5.12 PUS SERVICE 223 – STR SPECIFIC MEMORY HANDLING TCS

The STR specific service 223 is used to enable/disable the memory load/dump/check commands for the memory ranges listed in section 13. Note that service 8 which includes a “function enable” subservice does not satisfy our needs because this subservice does not support timeout controlled enable states.

Furthermore, service 223 covers some copy operations which because of their very specific nature do not fit in any of the stardard PUS services.

Service 223 covers the following:

- TC(223,1) denotes the TC_ENABLE_MEMREAD command described in section 6.9.1
- TC(223,2) denotes the TC_ENABLE_MEMWRITE command described in section 6.9.2
- TC(223,3) denotes the TC_DISABLE_MEMREAD command described in section 6.9.3
- TC(223,4) denotes the TC_DISABLE_MEMWRITE command described in section 6.9.4
- TC(223,10) denotes the TC_COPY_EE2TAPP command described in section 6.9.5
- TC(223,11) denotes the TC_COPY_TAPP2EE command described in section 6.9.6

The *MemID* parameter given with each of the ENABLE/DISABLE commands specifies the memory for which the memory read (or write, resp.) commands shall be enabled or disabled.

The ENABLE commands additionally specify a *timeout* after which any read (or write, resp.) commands for the specified *MemID* will automatically be rejected by the STR.

The two COPY commands do not support the specification of MemIDs. Instead, the source and destination MemIDs implicitly used by the COPY commands are fix as follows:

TC_COPY_EE2TAPP: srcMemID=11 and 51, destMemID=71 and 81

TC_COPY_TAPP2EE: srcMemID=71 and 81, destMemID=11 and 51

The usage of the service 223 commands is as follows:

- initially all I/O related memories listed in section 13 are in the state ‘commanded reads disabled’ which means that the execution of TC_DUMP_MEMORY and TC_CHECK_MEMORY commands with a *MemID* parameter specifying any of the I/O related memories will initially fail (note that TC_DUMP_MEMORY and TC_CHECK_MEMORY commands for other than the I/O related memories are not affected by this behavior)
- initially all writeable memories listed in section 13 are in the state ‘commanded writes disabled’ which means that the execution of TC_LOAD_MEMORY, TC_COPY_EE2TAPP and TC_COPY_TAPP2EE commands will initially fail
- a TC_ENABLE_MEMREAD command for a *MemID* which denotes one of the I/O related memories enables the execution of TC_DUMP_MEMORY and TC_CHECK_MEMORY commands for this same *MemID* and starts a timeout counter; a TC_DUMP_MEMORY or TC_CHECK_MEMORY command for this same *MemID* can then be executed successfully as long as the *timeout* value given in the TC_ENABLE_MEMREAD command has not elapsed (note that if the *MemID* given in the TC_ENABLE_MEMREAD command does not specify one of the I/O related memories the TC_ENABLE_MEMREAD command will be rejected)
- if a TC_DUMP_MEMORY or TC_CHECK_MEMORY command for any of the I/O related memories has been accepted and successfully executed the read related timeout counter for the *MemID* will automatically be restarted (this is to allow a series of dump/check commands for the same *MemID* without the need of intermingled TC_ENABLE_MEMREAD commands)
- the TC_DISABLE_MEMREAD command forces the ‘read timeout elapsed’ state for the *MemID* specified with the DISABLE command and thus immediately disables the execution of further TC_DUMP_MEMORY and TC_CHECK_MEMORY commands for this *MemID* (note that if the *MemID* given in the TC_DISABLE_MEMREAD command does not specify one of the I/O related memories the TC_DISABLE_MEMREAD command will fail)

- a TC_ENABLE_MEMWRITE command for a specified *MemID* enables the execution of TC_LOAD_MEMORY commands for this *MemID* and starts a timeout counter; the TC_LOAD_MEMORY command can then be executed successfully as long as the *timeout* value given in the TC_ENABLE_MEMWRITE command has not elapsed (note that for a TC_COPY_EE2TAPP command two TC_ENABLE_MEMWRITE commands, one for MemID 71 and one for MemID 81, are necessary, and for a TC_COPY_TAPP2EE command also two TC_ENABLE_MEMWRITE commands are necessary, one for MemID 11 and one for MemID 51)
- if a TC_LOAD_MEMORY command has successfully been executed the write related timeout counter which is associated with the *MemID* will automatically be restarted (this is to allow a series of load commands to the same destination memory without the need of intermingled TC_ENABLE_MEMWRITE commands)
- the TC_DISABLE_MEMWRITE command forces the 'write timeout elapsed' state for the *MemID* specified with the DISABLE command and thus immediately disables the execution of further TC_LOAD_MEMORY commands for this *MemID* (note that a TC_DISABLE_MEMWRITE command for MemID 71 or 81 additionally disables the execution of TC_COPY_EE2TAPP commands and a TC_DISABLE_MEMWRITE command for MemID 11 or 51 additionally disables the execution of TC_COPY_TAPP2EE commands)

5.13 PUS SERVICE 224 – STR SPECIFIC PARAMETER MANAGEMENT

According to AD1, section B2.12.7.3, the STR specific service 224 is used for specific STR parameter update commands.

The following service 224 commands are supported by the STR:

- TC(224,1) denotes the TC_ENABLE_TEMPERATURE command described in section 6.10.1. Its purpose is to enable the execution of the TC_TEMPERATURE command.
- TC(224,4) denotes the TC_PRECESSION command described in section 6.10.2.
- TC(224,5) denotes the TC_ABBERATION command described in section 6.10.3
- TC(224,6) denotes the TC_RATE_LIMIT command described in section 6.10.4
- TC(224,7) denotes the TC_TEMPERATURE command described in section 6.10.5
- TC(224,8) denotes the TC_ZERO_COUNTERS command described in section 6.10.6
- TC(224,9) denotes the TC_TIME_SHIFT_THRESHOLD command described in section 6.10.7
- TC(224,10) denotes the TC_MAX_DUMPSIZE command described in section 6.10.8

6 COMMAND STRUCTURES

6.1 SERVICE 3

6.1.1 TC_ENABLE_HKPKT – TURN ON THE REPORTING OF A SPECIFIED HK PACKET

Description: This command enables the periodic reporting of a specified HK TM packet.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 3
Service Subtype: 5
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 1 byte
Structure of the TC application data field:

Name	Description	Format	Range
SID	the HK TM structure ID for which reporting shall be enabled	UI8	1, 105, 106 (see section 8)

Note:

1. The reporting rate which is currently configured for the HK packet will not be changed by this command.

6.1.2 TC_DISABLE_HKPKT – TURN OFF THE REPORTING OF A SPECIFIED HK PACKET

Description: This command disables the periodic reporting of a specified HK TM packet.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 3
Service Subtype: 6
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 1 byte

Structure of the TC application data field:

Name	Description	Format	Range
SID	the HK TM structure ID for which reporting shall be disabled	UI8	1, 105, 106 (see section 8)

Note:

1. The reporting rate which is currently configured for the HK packet will not be changed by this command.

6.1.3 TC_ENABLE_DIAGPKT – TURN ON THE REPORTING OF A SPECIFIED DIAG PACKET

Description: This command enables the periodic reporting of a specified Diagnostics TM packet.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 3
Service Subtype: 7
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 1 byte

Structure of the TC application data field:

Name	Description	Format	Range
SID	the Diag TM structure ID for which reporting shall be enabled	UI8	128, 188…203 (see section 8)

Note:

1. The reporting rate which is currently configured for the Diag packet will not be changed by this command.

6.1.4 TC_DISABLE_DIAGPKT – TURN OFF THE REPORTING OF A SPECIFIED DIAG PACKET

Description: This command disables the periodic reporting of a specified Diagnostics TM packet.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 3

Service Subtype: 8
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 1 byte

Structure of the TC application data field:

Name	Description	Format	Range
SID	the Diag TM structure ID for which reporting shall be disabled	UI8	128, 188…203 (see section 8)

Note:

1. The reporting rate which is currently configured for the Diag packet will not be changed by this command.

6.1.5 TC_DUMP_SIDSTATES – REPORT THE ENABLE STATES AND RATES OF ALL HK/DIAG PACKETS

Description: This command reports the current enable states and reporting periods of all HK and Diagnostics packets.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 3
Service Subtype: 128
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 0

Note:

1. The response generated by this command is a TM_SIDSTATES_DUMP packet.

6.1.6 TC_SET_HKPKTRATE – SET THE REPORTING RATE OF A SPECIFIED HK PACKET

Description: This command sets the reporting period for a specified HK TM packet.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 3

Service Subtype: 130
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 3 byte
Structure of the TC application data field (offsets are wrt. The start of the application data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
SID	the HK TM structure ID for which the reporting period shall be set	0	UI8	1	1, 105, 106 (see section 8)	
period	reporting period for the HK packet [number of STR cycles]	1	UI16	2	1…65535	0.1s

Note:

1. The report generation of the HK packet specified by SID must be disabled (the command will be rejected if this is not the case).
2. The reporting enable state of the packet will not be changed by this command.

6.1.7 TC_SET_DIAGPKTRATE – SET THE REPORTING RATE OF A SPECIFIED DIAG PACKET

Description: This command sets the reporting period for a specified Diagnostics TM packet.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 3
Service Subtype: 131
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 3 byte
Structure of the TC application data field (offsets are wrt. The start of the application data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
SID	the Diag TM structure ID for which the reporting period shall be set	0	UI8	1	128, 188…203 (see section 8)	
period	reporting period for the Diag packet [number of STR cycles]	1	UI16	2	1…65535	0.1s

Note:

1. The report generation of the Diag packet specified by SID must be disabled (the command will be rejected if this is not the case).
2. The reporting enable state of the packet will not be changed by this command.

6.1.8 TC_REPORT_HKPKT – REPORT A SPECIFIED HK OR DIAG PACKET ONCE

Description: This command requests a one-time report of a specified HK or Diagnostics packet.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 3
Service Subtype: 136
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 1 byte
Structure of the TC application data field:

Name	Description	Format	Range
SID	the HK or Diag structure ID for which a one-time report is requested	UI8	any of the SIDs supported by the STR (see section 8)

Note:

1. Neither the reporting enable state nor the rate which is currently configured for the specified SID will be changed by this command.

6.2 SERVICE 5

6.2.1 TC_ENABLE_EVENTS – ENABLE THE REPORTING OF SPECIFIED EVENTS

Description: This command enables the reporting of all events which are specified by their EIDs in command.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 5

Service Subtype: 5
Duration Type: short
STR modes accepting the TC: any
Length of the application data: 1+n*2 byte (see note 1)

Structure of the TC application data field (offsets are wrt. The start of the application data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
n	the number of events to be enabled	0	UI8	1	≥1 (see note 1)
eids[*n*]	list of event identifiers for which the reporting shall be enabled	1	UI16	*n**2	any of the EIDs supported by the STR (see section 9)

Note:

1. To be within the length restriction of 206 bytes given for TC packets in AD1, section 1.5, the count parameter *n* must be ≤96. However, the STR itself will accept TC_ENABLE_EVENTS packets with a count parameter $n \leq 106$.

6.2.2 TC_DISABLE_EVENTS – DISABLE THE REPORTING OF SPECIFIED EVENTS

Description: This command disables the reporting of all events which are specified by their EIDs in command.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 5
Service Subtype: 6
Duration Type: short
STR modes accepting the TC: any
Length of the application data: 1+n*2 byte (see note 1)

Structure of the TC application data field (offsets are wrt. The start of the application data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
n	the number of events to be disabled	0	UI8	1	≥1 (see note 1)
eids[*n*]	list of event identifiers for which the reporting shall be disabled	1	UI16	*n**2	any of the EIDs supported by the STR (see section 9)

Note:

1. To be within the length restriction of 206 bytes given for TC packets in AD1, section 1.5, the count parameter *n* must be ≤96. However, the STR itself will accept TC_DISABLE_EVENTS packets with a count parameter $n \leq 106$.

6.2.3 TC_DUMP_DISEVENTS – REPORT ALL CURRENTLY DISABLED EVENTS

Description:	This command reports all events the reporting of which is currently disabled.
PRID:	see section 5.1
PCAT:	12 (telecommand)
Service type:	5
Service Subtype:	133
Duration Type:	short
STR modes accepting the TC:	any
Length of the application data:	0

Note:

1. The response generated by this command is a TM_DISEVENTS_DUMP packet.

6.3 SERVICE 6

6.3.1 TC_LOAD_MEMORY – UPLOAD MEMORY CONTENT TO CONSECUTIVE ADDRESSES

Description:	The TC_LOAD_MEMORY command uploads a data block which shall be written into a specified memory area.
PRID:	see section 5.1
PCAT:	12 (telecommand)
Service type:	6
Service Subtype:	2
Duration Type:	short
STR modes accepting the TC:	BOOT, STANDBY
Length of the application data:	10+n*SAU, n≥1 (see notes 3 and 4)

Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
MemID	memory id	0	UI16	2	see note 1 and section 13	
addr	physical start address	2	UI32	4	see note 2 and section 13	
n	amount of data being uploaded	6	UI32	4	≥1 (see notes 3 and 4)	SAU
data[*n*]	the block of data to be written to memory	10		*n**SAU		

Notes:

1. The command will fail if memory writing for the specified *MemID* is currently disabled (see TC_ENABLE_MEMWRITE).
2. The start address specified by *addr* must be a multiple of the SAU size defined for the specified *MemID* (see section 13).
3. The count parameter *n* must be >0 (a NULL upload makes no sense) and the upload address range defined by *addr*…*addr*+*n**SAU-1 must be within the limits associated with the specified *MemID.*
4. To be within the length restriction of 206 bytes given for TC packets in AD1, section 1.5, the value *n**SAU must be ≤184. However, the STR itself will accept TC packets with a total size of up to 2044 bytes.

6.3.2 TC_DUMP_MEMORY – DUMP MEMORY CONTENT

Description: The TC_DUMP_MEMORY command requests to dump a specified memory area.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type : 6
Service Subtype : 5
Duration Type : long
STR modes accepting the TC: any
Length of the application data: 10 byte

Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
MemID	memory id	0	UI16	2	see note 1 and section 13	
saddr	physical start address	2	UI32	4	see note 2 and section 13	
nToDump	total amount of data to be dumped	6	UI32	4	≥1 (see note 3)	SAU

Notes:

1. The MemIDs which are assigned to I/O related address ranges require that the read operation on these ranges must be enabled (see section 13). Thus, for these MemIDs the TC_DUMP_MEMORY command will fail if memory reading for the specified *MemID* is currently disabled (see TC_ENABLE_MEMREAD).

2. The start address specified by *saddr* must be a multiple of the SAU size, where SAU is the smallest addressable unit defined for the specified *MemID* (see section 13).

3. The count parameter *nToDump* must be >0 and the address range defined by *saddr*…*saddr*+*nToDump**SAU-1 must be within the limits associated with the specified *MemID*.

4. The total dump range will be split into pieces of memory units from the specified MemID. Each piece will be delivered with a separate TM_MEM_DUMP packet. The size of source data within each TM_MEM_DUMP is specified by TC_MAX_DUMPSIZE. If downlink bandwidth is a concern size of source data in a TM_MEM_DUMP packet has to be restricted accordingly by the caller using TC_MAX_DUMPSIZE.

6.3.3 TC_CHECK_MEMORY – CHECK MEMORY CONTENT

Description: The TC_CHECK_MEMORY command requests to calculate and report the checksum for a specified memory area.
PRID: see section 5.1
PCAT: must be set to 12 (telecommand)
Service type : 6
Service Subtype : 9
Duration Type : long
STR modes accepting the TC: any
Length of the application data: 10 byte

Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
MemID	memory id	0	UI16	2	see section 13	

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
addr	physical start address	2	UI32	4	see section 13	SAU
n	amount of data for which the checksum shall be calculated	6	UI32	4	see note 3	SAU

Notes:

1. The MemIDs which are assigned to I/O related address ranges require that the read operation on these ranges must be enabled (see section 13). Thus, for these MemIDs the TC_CHECK_MEMORY command will fail if memory reading for the specified *MemID* is currently disabled (see TC_ENABLE_MEMREAD).

2. The start address specified by *addr* must be a multiple of the SAU size, where SAU is the smallest addressable unit defined for the specified *MemID* (see section 13).

3. The count parameter *n* must be >0 (a check of an empty memory block makes no sense) and the address range defined by *addr*…*addr*+*n**SAU-1 must be within the limits associated with the specified *MemID*.The memory check result will be delivered in the form of a TM(6,10) packet.

4. Depending on the value of *n* the checksum calculation may be a lengthy operation which cannot be completed within the same frame (this is especially true for MemID 11 which specifies the EEPROM). To indicate an ongoing checksum calculation the STR will generate progress events of type EV_MEMCHK_INPROGRESS at a 0.1Hz rate (1 event packet every 10s); the first one generated 1s after the checksum calculation has been started). See also information in section 2.3.1 on telemetry update rates.

6.4 SERVICE 8

6.4.1 TC_SELFTEST – PERFORM A SELF TEST

Description: The TC_SELFTEST command requests a selftest of the STR.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type : 8
Service Subtype : 1
Duration Type : long
STR modes accepting the TC: BOOT, STANDBY
Length of the application data: 2 byte
Structure of the application data field:

Argument Name	Description	Format	Length (byte)	Range
fid	ID of the function to be executed	UI8	1	0 (see FunctionID enumeration)
reqTests	a bitfield indicating which tests to be performed	B8	1	see note 1

Note:

1. The bit meanings of the *reqTests* parameter are as follows (b_0 is msb, b_7 is lsb):

   b_0: if 1: verify the CRC of the Boot Mode Handler S/W in ROM; if 0: skip this check

   b_1: if 1: verify the CRC of the Application S/W in EEPROM; if 0: skip this check

   b_2: if 1: verify the CRC of the APSW Calibration Data block in EEPROM; if 0: skip this check

   b_3: if 1: verify the CRC of the Test Application S/W in the dedicated RAM area; if 0: skip this check

   b_4: if 1: verify the CRC of the Testapp Calibration Data block in the dedicated RAM area; if 0: skip this check

   b_5…b_7: spare (must be 0)

   Example values for *reqTests*:

   0xF8 selftest of Boot Mode Handler, Application SW, APSW Calibration Data, Test Application S/W, Testapp Calibration Data

   0xE0 selftest of Boot Mode Handler, Application SW, APSW Calibration Data

   0x18 selftest of Test Application S/W, Testapp Calibration Data

6.4.2 TC_MANAGE_LONGDURCMDS – MANAGE LONG DURATION COMMANDS

Description: This command may be used to request the abortion of the long duration command which is currently in progress (if any) and/or to clear the long duration command queue maintained by the STR.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 8
Service Subtype: 220

Duration Type: short
STR modes accepting the TC: any
Length of the application data: 2
Structure of the TC application data field:

Argument Name	Description	Format	Length (byte)	Range
abortActiveCmd	specifies if the long duration TC, if any is currently in progress, shall be aborted	UI8	1	0: do not abort the running long duration cmd 1: abort the running long duration cmd
clearQueue	specifies if the long duration TC queue shall be cleared	UI8	1	0: do not clear the queue 1: clear the queue

Notes:

1. If the *abortActiveCmd* parameter is 1 and a long duration command is currently in progress it will be aborted and a TM(1,8) completion report will be sent by the aborted cmd indicating 'TC aborted by request'. Recall (see section 2.2) that at any time only one long duration command may be active, i.e. be in progress.

2. If the *clearQueue* parameter is 1 and the long duration command queue maintained by the STR is not empty, any TC which is currently contained in the queue will be removed and a TM(1,8) completion report will be sent for each removed TC indicating 'TC aborted by request'.

6.5 SERVICE 9

6.5.1 TC_VERIFY_TIME_SYNC – TRIGGER TIME SYNCHRONIZATION VERIFICATION

Description: This is the standard TC(9,135) command.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 9
Service Subtype: 135
Duration Type: short
STR modes accepting the TC: any
Length of the application data: 0
Notes:

1. On success a TM(5,1) event report of type EV_TIME_SYNC_SUCCESS will be generated.

2. If time synchronization fails a TM(5,3) event report of type EV_TIME_SYNC_FAILED will be generated.

6.5.2 TC_SELECT_SYNC_SOURCE – SELECT SOURCE FOR EXTERNAL SYNCHRONIZATION

PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 9
Service Subtype: 136
Duration Type: short
STR modes accepting the TC: any
Length of the application data: 1 byte
Structure of the application data field:

Argument Name	Description	Format	Length (byte)	Range
syncSource	Synchronize sensor cycle with this external source	E	1	see SyncSource enumeration

6.6 SERVICE 17

6.6.1 TC_PING – CONNECTION TEST

Description: The TC_PING command is used to check if a connection at the application layer between the ground application and the STR application is possible and working correctly.
PRID: see section 5.1
PCAT: must be set to 12 (telecommand)
Service type: 17
Service Subtype: 1
Duration Type: short
STR modes accepting the TC: any
Length of the application data: 0

Note:

1. On this command the STR will perform no action but responding with a TM_PING packet.

6.7 SERVICE 220

6.7.1 TC_STANDBY – PASS TO STANDBY MODE

Description: The TC_STANDBY command switches the STR to the STANDBY mode.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 220
Service Subtype: 1
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 0

6.7.2 TC_ACQUIRE – PASS TO AAD MODE

Description: The TC_ACQUIRE command switches the STR to the AAD mode.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 220
Service Subtype: 2
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT
Length of the application data: 38 bytes
Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
aprioriQuality	quality of a-priori information	0	E	2	see AprioriQuality enumeration	

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
enablePrecessionCorrection	If set to 1, precession correction will be performed using the Julian date specified below	2	B	1	[0,1]	
enableAberrationCorrection	if set to 1, aberration correction will be performed using the velocity vector specified below	3	B	1	[0,1]	
spare		4	B	4		
rateLimit	expected maximum of the absolutes of the x and y components of the rate; a value of 0 means: use defaultRateLimit Note: the specified value will always be limited by the maximum supported rate.	8	UI	8	0…255	2^{-5} deg/s
tValid	time of validity for a-priori attitude quaternion	16	T	48	acc. 3.3.6 but with last byte omitted	1ms
qv1	vector part of a-priori attitude quaternion	64	SI	32	see notes 1 and 2	2^{-30}
qv2	vector part of a-priori attitude quaternion	96	SI	32		2^{-30}
qv3	vector part of a-priori attitude quaternion	128	SI	32		2^{-30}
qs	scalar part of a-priori attitude quaternion	160	SI	32		2^{-30}
rateX	a-priori rate in BRF around x-axis	192	SI	16	see notes 1 and 2	2^{-11} deg/s
rateY	a-priori rate in BRF around y-axis	208	SI	16		2^{-11} deg/s
rateZ	a-priori rate in BRF around z-axis	224	SI	16		2^{-11} deg/s
julianDate	Julian date, specified in number of days since the epoch date of the built-in star catalog; day #0 corresponds to JD 2451545	240	UI	16	see note 3	Julian days
velocityVectorX	S/C velocity vector (in heliocentric, inertial coordinate frame); used for aberration correction	256	SI	16	see note 4	2^{-22} c_{vac}
velocityVectorY		272	SI	16		2^{-22} c_{vac}
velocityVectorZ		288	SI	16		2^{-22} c_{vac}

Notes:

1. If the aprioriQuality parameter indicates **completeApriori**, then the quaternion (qv1,qv2,qv3,qs) and the rate vector (rateX,rateY,rateZ) define the attitude and rate of the STR's BRF (Boresight Reference Frame) wrt. The inertial frame. The rateLimit parameter is meaningless in this case (will be ignored).

2. If the aprioriQuality parameter indicates **losApriori**, then the vector part of the quaternion (qv1,qv2,qv3) defines the STR's LOS direction (+z direction of the STR's BRF) vector wrt. The inertial frame. The rate vector (rateX,rateY,rateZ) is meaningless in this case (will be ignored) but the rateLimit parameter will be used instead.

3. The julianDate parameter will be ignored if the enablePrecessionCorrection parameter is 0.

4. The velocity vector will be ignored if the enableAberrationCorrection parameter is 0. Otherwise the vector (velocityVectorX,velocityVectorY,velocityVectorZ) must specify the S/C velocity in the sun centered inertial frame multiplied by the factor $2^{22}/\mathbf{c}_{vac}$, where $\mathbf{c}_{vac}$ denotes the vacuum speed of light (299792.458 km/s). For example, given a S/C velocity of magnitude 35 km/s. Then the vector to be delivered in the TC_ACQUIRE command should have a magnitude which can be calculated by $35 \cdot 2^{22}/299792.458$ which is about the value 490, i.e. the length of the vector (velocityVectorX,velocityVectorY,velocityVectorZ) should be about 490.

5. The a-priori time information must be greater than the time of the last reboot of the STR.

6.7.3 TC_PHOTO – PASS TO PHOTO MODE

Description: The TC_PHOTO command switches the STR to the PHOTO mode.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 220
Service Subtype: 3
Duration Type: short
STR modes accepting the TC: STANDBY, PHOTO
Length of the application data: 10 bytes

Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
xMin	minimum x-coordinate of the rectangular area to be dumped	0	UI	16	0..1023	pixel
yMin	minimum y-coordinate of the rectangular area to be dumped	16	UI	16	0..1023	pixel
xWidth	width of the rectangular area to be dumped (size in x direction)	32	UI	16	0..1020	pixel
yHeight	height of the rectangular area to be dumped (size in in y direction)	48	UI	16	0..1020	pixel
tInt	integration time will automatically be limited to supported values	64	UI	8		ms
maxPdbsToUse	maximum number of PDBs generated in each cycle; value 0 means: 16 PDBs	72	UI	4	0…15	

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
repetitive	if set to 1: repeat photo of the specified area until changed or stopped explicitly if set to 0: go back to Standby mode after one shot of the specified area	76	B	1	0, 1	
enableBLReadOut	if set to 1, the photo is taken with black-level correction enabled	77	B	1	0, 1	
gain	ADC gain	78	E	2	see Gain enumeration	

Notes:

1. In PHOTO mode the delivers the pixel dump of a rectangular part of the detector area. In x direction this rectangle starts at *xMin* (left edge of the rectangle) and ends at *xMin*+*xWidth*-1 (right edge of the rectangle) and in y direction it starts at *yMin* (bottom edge of the rectangle) and ends at *yMin*+*yHeight*-1 (top edge of the rectangle).

2. The dump of the specified rectangular detector part will be delivered as a series of TM_PDB packets. The rectangular area will be split into small windows of a fixed size 15*15 pixels. Each TM_PDB packet contains the information about one window including the location and size of the window as well as the pixel data. Location of the window means the detector x and y coordinates (column and row number) of the first pixel (minimum x and y coordinates) covered by the window, and size of the window means the window width and height (number of columns and rows).

   The first window will be located at the position specified by the *xMin* and *yMin* parameters of the TC_PHOTO command. Each further window will then be located at the right edge of the previous window (no horizontal overlap of the windows). This is repeated until the right rectangle edge *xMin*+15-1 is reached, i.e. until one complete horizontal stripe of the rectangle has been dumped. The windowing then restarts at the left edge of the next horizontal stripe. This will be repeated until the whole rectangle is dumped. The total count of TM_PDB packets necessary to dump the whole rectangle depends on the size of the rectangle.

   During one sensor cycle the content of *maxPdbsToUse* windows (at most 16) can be dumped. If these *maxPdbsToUse* windows does not cover the complete rectangular area the next *maxPdbsToUse* windows are dumped in the next sensor cycle (read out of pixel data one cycle later as well).

   The number of sensor cycles necessary for the complete dump is given by

   ((((*xWidth*+**15**-1)/**15**)*((*yHeight*+**15**-1)/**15**))+*maxPdbsToUse*-1)/*maxPdbsToUse*.

   Note that in this formula only the integer parts of the division results must be used. To obtain the duration of the complete dump in seconds divide the result by 10.

   If bandwidth is a concern, *maxPdbsToUse* has to be restricted accordingly by the caller.

3. If the flag parameter *repetitive* is 0 the STR autonomously switches to STANDBY mode when the rectangle dump is complete. Otherwise the STR will stay in PHOTO mode and will repeat the rectangle dumps until explicitely aborted (with a TC_STANDBY or a TC_RESET command).

4. If the STR receives a TC_PHOTO command while already in PHOTO mode the currently running rectangle dump will be aborted and a new dump will be started based on the parameters specified with the new TC_PHOTO command.

5. In the command TC_PHOTO the parameters *xMin, xWidth, yMin* and *yHeight* refer to the rectangular area, i.e. the complete dump. In the TM_PDB packet the parameters *xMin, xWidth, yMin* and *yHeight* refer to the window covered by the TM_PDB, i.e. the only part of the dump data.

6.7.4 TC_RESET – PERFORM A RESET (REBOOT) OF THE STAR SENSOR

Description: The TC_RESET command stores the specified startMode in SGM and forces a H/W reset of the STR.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 220
Service Subtype: 4
Duration Type: short
STR modes accepting the TC: any
Length of the application data: 1
Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Format	Length (byte)	Range
startMode	the mode in which the boot loader shall start	UI8	1	see StartMode enumeration

Note:

1. After generating the command acknowledge packets TM(1,1) and TM(1,7) for this command the STR waits 0.5s before forcing the H/W reset in order to allow the BC to fetch the TM(1,1) and TM(1,7) packets from the bus (the delay is necessary because after the reset the MIL bus I/F H/W of the STR will be re-initialized).

6.8 SERVICE 221

6.8.1 TC_LOAD_CALIBRATION1 – UPLOAD PART1 OF SENSOR CALIBRATION DATA

Description: The TC_LOAD_CALIBRATION1 command uploads part 1 of the STR's calibration data.

PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 221
Service Subtype: 1
Duration Type: short
STR modes accepting the TC: STANDBY
Length of the application data: 104 byte

Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)
optMainPtX	Detector column coordinate (subpixel) of the optical main point	0	F	4
optMainPtY	Detector row coordinate (subpixel) of the optical main point	4	F	4
$a_{0,0}$	$a_{0,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	8	F	4
$a_{0,1}$	$a_{0,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	12	F	4
$a_{0,2}$	$a_{0,2}$ coefficient for transformation from detector(x,y) to tan(alpha)	16	F	4
$a_{0,3}$	$a_{0,3}$ coefficient for transformation from detector(x,y) to tan(alpha)	20	F	4
$a_{1,0}$	$a_{1,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	24	F	4
$a_{1,1}$	$a_{1,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	28	F	4
$a_{1,2}$	$a_{1,2}$ coefficient for transformation from detector(x,y) to tan(alpha)	32	F	4
$a_{1,3}$	$a_{1,3}$ coefficient for transformation from detector(x,y) to tan(alpha)	36	F	4
$a_{2,0}$	$a_{2,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	40	F	4
$a_{2,1}$	$a_{2,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	44	F	4
$a_{2,2}$	$a_{2,2}$ coefficient for transformation from detector(x,y) to tan(alpha)	48	F	4
$a_{2,3}$	$a_{2,3}$ coefficient for transformation from detector(x,y) to tan(alpha)	52	F	4
$a_{3,0}$	$a_{3,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	56	F	4
$a_{3,1}$	$a_{3,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	60	F	4
$a_{3,2}$	$a_{3,2}$ coefficient for transformation from detector(x,y) to tan(alpha)	64	F	4
$a_{3,3}$	$a_{3,3}$ coefficient for transformation from detector(x,y) to tan(alpha)	68	F	4
$a_{0,4}$	$a_{0,4}$ coefficient for transformation from detector(x,y) to tan(alpha)	72	F	4
$a_{1,4}$	$a_{1,4}$ coefficient for transformation from detector(x,y) to tan(alpha)	76	F	4
$a_{4,0}$	$a_{4,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	80	F	4
$a_{5,0}$	$a_{5,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	84	F	4
$a_{temp,0,0}$	$a_{temp,0,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	88	F	4

Argument Name	Description	Offset (byte)	Format	Length (byte)
$a_{temp,0,1}$	$a_{temp,0,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	92	F	4
$a_{temp,1,0}$	$a_{temp,1,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	96	F	4
$a_{temp,1,1}$	$a_{temp,1,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	100	F	4

Note:

1. For the complete sensor calibration data structure see section 12.2.

6.8.2 TC_LOAD_CALIBRATION2 – UPLOAD PART2 OF SENSOR CALIBRATION DATA

Description: The TC_LOAD_CALIBRATION2 command uploads part 2 of the STR's calibration data.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 221
Service Subtype: 2
Duration Type: short
STR modes accepting the TC: STANDBY
Length of the application data: 96 byte

Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)
$b_{0,0}$	$b_{0,0}$ coefficient for transformation from detector(x,y) to tan(beta)	0	F	4
$b_{0,1}$	$b_{0,1}$ coefficient for transformation from detector(x,y) to tan(beta)	4	F	4
$b_{0,2}$	$b_{0,2}$ coefficient for transformation from detector(x,y) to tan(beta)	8	F	4
$b_{0,3}$	$b_{0,3}$ coefficient for transformation from detector(x,y) to tan(beta)	12	F	4
$b_{1,0}$	$b_{1,0}$ coefficient for transformation from detector(x,y) to tan(beta)	16	F	4
$b_{1,1}$	$b_{1,1}$ coefficient for transformation from detector(x,y) to tan(beta)	20	F	4
$b_{1,2}$	$b_{1,2}$ coefficient for transformation from detector(x,y) to tan(beta)	24	F	4
$b_{1,3}$	$b_{1,3}$ coefficient for transformation from detector(x,y) to tan(beta)	28	F	4
$b_{2,0}$	$b_{2,0}$ coefficient for transformation from detector(x,y) to tan(beta)	32	F	4

Argument Name	Description	Offset (byte)	Format	Length (byte)
$b_{2,1}$	$b_{2,1}$ coefficient for transformation from detector(x,y) to tan(beta)	36	F	4
$b_{2,2}$	$b_{2,2}$ coefficient for transformation from detector(x,y) to tan(beta)	40	F	4
$b_{2,3}$	$b_{2,3}$ coefficient for transformation from detector(x,y) to tan(beta)	44	F	4
$b_{3,0}$	$b_{3,0}$ coefficient for transformation from detector(x,y) to tan(beta)	48	F	4
$b_{3,1}$	$b_{3,1}$ coefficient for transformation from detector(x,y) to tan(beta)	52	F	4
$b_{3,2}$	$b_{3,2}$ coefficient for transformation from detector(x,y) to tan(beta)	56	F	4
$b_{3,3}$	$b_{3,3}$ coefficient for transformation from detector(x,y) to tan(beta)	60	F	4
$b_{0,4}$	$b_{0,4}$ coefficient for transformation from detector(x,y) to tan(beta)	64	F	4
$b_{0,5}$	$b_{0,5}$ coefficient for transformation from detector(x,y) to tan(beta)	68	F	4
$b_{4,0}$	$b_{4,0}$ coefficient for transformation from detector(x,y) to tan(beta)	72	F	4
$b_{4,1}$	$b_{4,1}$ coefficient for transformation from detector(x,y) to tan(beta)	76	F	4
$b_{temp,0,0}$	$b_{temp,0,0}$ coefficient for transformation from detector(x,y) to tan(beta)	80	F	4
$b_{temp,0,1}$	$b_{temp,0,1}$ coefficient for transformation from detector(x,y) to tan(beta)	84	F	4
$b_{temp,1,0}$	$b_{temp,1,0}$ coefficient for transformation from detector(x,y) to tan(beta)	88	F	4
$b_{temp,1,1}$	$b_{temp,1,1}$ coefficient for transformation from detector(x,y) to tan(beta)	92	F	4

Note:

2. For the complete sensor calibration data structure see section 12.2.

6.8.3 TC_LOAD_CALIBRATION3 – UPLOAD PART3 OF SENSOR CALIBRATION DATA

Description: The TC_LOAD_CALIBRATION3 command uploads part 3 of the STR's calibration data.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 221
Service Subtype: 3
Duration Type: short
STR modes accepting the TC: STANDBY
Length of the application data: 180 byte

Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)
$c_{0,0}$	$c_{0,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	0	F	4
$c_{0,1}$	$c_{0,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	4	F	4
$c_{0,2}$	$c_{0,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	8	F	4
$c_{0,3}$	$c_{0,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	12	F	4
$c_{1,0}$	$c_{1,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	16	F	4
$c_{1,1}$	$c_{1,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	20	F	4
$c_{1,2}$	$c_{1,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	24	F	4
$c_{1,3}$	$c_{1,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	28	F	4
$c_{2,0}$	$c_{2,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	32	F	4
$c_{2,1}$	$c_{2,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	36	F	4
$c_{2,2}$	$c_{2,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	40	F	4
$c_{2,3}$	$c_{2,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	44	F	4
$c_{3,0}$	$c_{3,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	48	F	4
$c_{3,1}$	$c_{3,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	52	F	4
$c_{3,2}$	$c_{3,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	56	F	4
$c_{3,3}$	$c_{3,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	60	F	4
$c_{0,4}$	$c_{0,4}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	64	F	4
$c_{1,4}$	$c_{1,4}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	68	F	4
$c_{4,0}$	$c_{4,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	72	F	4
$c_{5,0}$	$c_{5,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	76	F	4
$d_{0,0}$	$d_{0,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	80	F	4
$d_{0,1}$	$d_{0,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	84	F	4
$d_{0,2}$	$d_{0,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	88	F	4
$d_{0,3}$	$d_{0,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	92	F	4
$d_{1,0}$	$d_{1,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	96	F	4
$d_{1,1}$	$d_{1,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	100	F	4
$d_{1,2}$	$d_{1,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	104	F	4
$d_{1,3}$	$d_{1,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	108	F	4
$d_{2,0}$	$d_{2,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	112	F	4
$d_{2,1}$	$d_{2,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	116	F	4
$d_{2,2}$	$d_{2,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	120	F	4
$d_{2,3}$	$d_{2,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	124	F	4

Argument Name	Description	Offset (byte)	Format	Length (byte)
$d_{3,0}$	$d_{3,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	128	F	4
$d_{3,1}$	$d_{3,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	132	F	4
$d_{3,2}$	$d_{3,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	136	F	4
$d_{3,3}$	$d_{3,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	140	F	4
$d_{0,4}$	$d_{0,4}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	144	F	4
$d_{0,5}$	$d_{0,5}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	148	F	4
$d_{4,0}$	$d_{4,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	152	F	4
$d_{4,1}$	$d_{4,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	156	F	4
s_x	s-curve correction factor for x-correction	160	F	4
s_y	s-curve correction factor for y-correction	164	F	4
tcRef0	reference total charge for t_{int}=1s and gain=1.0 and magnitude=0.0	168	F	4
sigmaX	size of star shape in x direction	172	F	4
sigmaY	size of star shape in y direction	176	F	4

Note:

1. For the complete sensor calibration data structure see section 12.2.

6.8.4 TC_LOAD_CALIBRATION4 – UPLOAD PART4 OF SENSOR CALIBRATION DATA

Description: The TC_LOAD_CALIBRATION4 command uploads part 4 of the STR's calibration data.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 221
Service Subtype: 4
Duration Type: short
STR modes accepting the TC: STANDBY
Length of the application data: 192 byte
Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range
x_1	x coordinate of first defect pixel	0	UI	10	0..1023
y_1	y coordinate of first defect pixel	10	UI	10	0..1023
$mode_1$	0: no correction 1: copy from (x_1-1,y_1) 2: copy from (x_1+1,y_1) 3: average	20	UI	2	0..3
$spare_1$	filler bits	22	B	2	
x_2	x coordinate of second defect pixel	24	UI	10	0..1023
y_2	y coordinate of second defect pixel	34	UI	10	0..1023
$mode_2$	0: no correction 1: copy from (x_2-1,y_2) 2: copy from (x_2+1,y_2) 3: average	44	UI	2	0..3
$spare_2$	filler bits	46	B	2	
...	...				
x_{64}	x coordinate of 64th defect pixel	1512	UI	10	0..1023
y_{64}	y coordinate of 64th defect pixel	1522	UI	10	0..1023
$mode_{64}$	0: no correction 1: copy from (x_{64}-1,y_{64}) 2: copy from (x_{64}+1,y_{64}) 3: average	1532	UI	2	0..3
$spare_{64}$	filler bits	1534	B	2	

Note:

1. For the complete sensor calibration data structure see section 12.2.

6.8.5 TC_LOAD_CALIBRATION5 – UPLOAD PART5 OF SENSOR CALIBRATION DATA

Description: The TC_LOAD_CALIBRATION5 command uploads part 5 of the STR's calibration data.
PRID: see section 5.1
PCAT: 12 (telecommand)

Service type: 221
Service Subtype: 5
Duration Type: short
STR modes accepting the TC: STANDBY
Length of the application data: 192 byte

Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range
x_{65}	x coordinate of 65th defect pixel	0	UI	10	0..1023
y_{65}	y coordinate of 65th defect pixel	10	UI	10	0..1023
$mode_{65}$	0: no correction 1: copy from (x_{65}-1,y_{65}) 2: copy from (x_{65}+1,y_{65}) 3: average	20	UI	2	0..3
$spare_{65}$	filler bits	22	B	2	
x_{66}	x coordinate of 66th defect pixel	24	UI	10	0..1023
y_{66}	y coordinate of 66th defect pixel	34	UI	10	0..1023
$mode_{66}$	0: no correction 1: copy from (x_{66}-1,y_{66}) 2: copy from (x_{66}+1,y_{66}) 3: average	44	UI	2	0..3
$spare_{66}$	filler bits	46	B	2	
...	...				
x_{128}	x coordinate of 128th defect pixel	1512	UI	10	0..1023
y_{128}	y coordinate of 128th defect pixel	1522	UI	10	0..1023
$mode_{128}$	0: no correction 1: copy from (x_{128}-1,y_{128}) 2: copy from (x_{128}+1,y_{128}) 3: average	1532	UI	2	0..3
$spare_{128}$	filler bits	1534	B	2	

Note:

1. For the complete sensor calibration data structure see section 12.2.

6.8.6 TC_LOAD_CALIBRATION6 – UPLOAD PART6 OF SENSOR CALIBRATION DATA

Description: The TC_LOAD_CALIBRATION6 command uploads part 6 of the STR's calibration data.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 221
Service Subtype: 6
Duration Type: short
STR modes accepting the TC: STANDBY
Length of the application data: 192 byte

Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range
x_{129}	x coordinate of 129th defect pixel	0	UI	10	0..1023
y_{129}	y coordinate of 129th defect pixel	10	UI	10	0..1023
$mode_{129}$	0: no correction 1: copy from (x_{129}-1,y_{129}) 2: copy from (x_{129}+1,y_{129}) 3: average	20	UI	2	0..3
$spare_{129}$	filler bits	22	B	2	
x_{130}	x coordinate of 130th defect pixel	24	UI	10	0..1023
y_{130}	y coordinate of 130th defect pixel	34	UI	10	0..1023
$mode_{130}$	0: no correction 1: copy from (x_{130}-1,y_{130}) 2: copy from (x_{130}+1,y_{130}) 3: average	44	UI	2	0..3
$spare_{130}$	filler bits	46	B	2	
...	...				
x_{192}	x coordinate of 192th defect pixel	1512	UI	10	0..1023
y_{192}	y coordinate of 192th defect pixel	1522	UI	10	0..1023

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range
$mode_{192}$	0: no correction 1: copy from (x_{192}-1,y_{192}) 2: copy from (x_{192}+1,y_{192}) 3: average	1532	UI	2	0..3
$spare_{192}$	filler bits	1534	B	2	

Note:

1. For the complete sensor calibration data structure see section 12.2.

6.8.7 TC_LOAD_CALIBRATION7 – UPLOAD PART7 OF SENSOR CALIBRATION DATA

Description: The TC_LOAD_CALIBRATION7 command uploads part 7 of the STR's calibration data.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 221
Service Subtype: 7
Duration Type: short
STR modes accepting the TC: STANDBY
Length of the application data: 192 byte

Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range
x_{193}	x coordinate of 193rd defect pixel	0	UI	10	0..1023
y_{193}	y coordinate of 193rd defect pixel	10	UI	10	0..1023
$mode_{193}$	0: no correction 1: copy from (x_{193}-1,y_{193}) 2: copy from (x_{193}+1,y_{193}) 3: average	20	UI	2	0..3
$spare_{193}$	filler bits	22	B	2	
x_{194}	x coordinate of 194th defect pixel	24	UI	10	0..1023

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range
y_{194}	y coordinate of 194th defect pixel	34	UI	10	0..1023
$mode_{194}$	0: no correction 1: copy from (x_{194}-1,y_{194}) 2: copy from (x_{194}+1,y_{194}) 3: average	44	UI	2	0..3
$spare_{194}$	filler bits	46	B	2	
...	...				
x_{256}	x coordinate of 256th defect pixel	1512	UI	10	0..1023
y_{256}	y coordinate of 256th defect pixel	1522	UI	10	0..1023
$mode_{256}$	0: no correction 1: copy from (x_{256}-1,y_{256}) 2: copy from (x_{256}+1,y_{256}) 3: average	1532	UI	2	0..3
$spare_{256}$	filler bits	1534	B	2	

Note:

1. For the complete sensor calibration data structure see section 12.2.

6.8.8 TC_LOAD_PREPROCESSOR – UPLOAD PREPROCESSOR REGISTER CONFIGURATION

Description: The TC_LOAD_PREPROCESSOR command uploads a new preprocessor configuration.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 221
Service Subtype: 10
Duration Type: short
STR modes accepting the TC: STANDBY
Length of the application data: 16 byte
Structure of the TC application data field:

Argument Name	Description	Length (byte)
PreProcPars	the complete preprocessor configuration structure(see section 12.1)	16

6.8.9 TC_LOAD_ALGORITHM – UPLOAD ALGORITHM CONTROL PARAMETERS

Description: The TC_LOAD_ALGORITHM command uploads a new set of algorithm control parameters.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 221
Service Subtype: 11
Duration Type: short
STR modes accepting the TC: STANDBY
Length of the application data: 116 byte

Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Length (byte)
AlgPars	the complete algorithm control parameters structure (see section 12.3)	116

6.8.10 TC_LOAD_GSC – UPLOAD PART OF THE GSC TABLE

Description: The TC_LOAD_GSC command uploads a specified part of the Guide Star Catalogue.
PRID: see section 5.1
PCAT: must be set to 12 (telecommand)
Service type: 221
Service Subtype: 12
Duration Type: short
STR modes accepting the TC: STANDBY
Length of the application data: 2+*n**16 byte (see note 2)

Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
n	number of catalog entries uploaded with this command	0	UI16	2	1..4000 (see note 2)
gscEntries[*n*]	the new data for the *n* catalog entries	2	see note 1	*n**16	

Notes:

1. Each GSC entry delivered in the *gscEntries[]* array occupies 16 bytes and has the following structure (see section 12.4):

Parameter Name	Description	Format	Length (byte)	Range	Unit
cnr	catalogue number of the guide star	UI16	2	1..4000	n.a.
mag	total charge of the guide star in detector LSB	UI16	2		[LSB]
v_x	unit vector of the guide star	F	4	−1.0 … +1.0	n.a.
v_y		F	4	−1.0 … +1.0	n.a.
v_z		F	4	−1.0 … +1.0	n.a.

The catalogue number given by *cnr* directs which catalogue entry will be overwritten. For example, *cnr*=5 specifies that the magnitude and direction vector of that catalogue entry which has the catalogue number 5 will be overwritten with the new data. The *n* entries provided with the *gscEntries[]* array may be given in any order of their *cnr* values.

The vector given by the components v_x, v_y, and v_z must be a unit vector, i.e. must have norm 1.

2. To be within the length restriction of 206 bytes given for TC packets in AD1, section 1.5, the count parameter *n* must be ≤12. However, the STR itself will accept TC packets with a total size of up to 2044 bytes.

6.8.11 TC_LOAD_GSC_INDICES – UPLOAD PART OF THE GSC INDEX TABLE

Description: The TC_LOAD_GSC_INDICES command uploads a specified part of the GSC tile index list.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 221
Service Subtype: 13

Duration Type: short
STR modes accepting the TC: STANDBY
Length of the application data: 2+*n**2 byte (see note 3)
Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
start	tile number (0-based) of the first GSC tile for which a new last star number is uploaded with this command	0	UI8	1	0..215
n	number of GSC tiles for which new last star numbers are uploaded with this command	1	UI8	1	1..216 (see notes 1 and 3)
gscIndices[*n*]	the new values for the *n* entries in the GSC Index list	2	UI16	*n**2	each value in the gscIndices array must be a number in range 1…4000)

Notes:

1. The count parameter *n* must fulfil the condition *start*+*n* ≤ 216.
2. See section 12.5 for the structure and meaning of the GSC Index table.
3. To be within the length restriction of 206 bytes given for TC packets in AD1, section 1.5, the count parameter *n* must be ≤96. However, the STR itself will accept TC packets with a total size of up to 2044 bytes.

6.8.12 TC_DUMP_CALIBRATION – DUMP SENSOR CALIBRATION DATA

Description: The TC_DUMP_CALIBRATION command requests to dump a specified part of the STR's calibration data structure.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 221
Service Subtype: 20
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT,PHOTO
Length of the application data: 4
Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
start	byte offset (0-based) within the calibration data structure at which the dump shall start	0	UI16	2	0..1147 (see note 2)
n	number of bytes to dump from the calibration data structure	2	UI16	2	1..1148 (see notes 1 and 2)

Notes:

1. The length parameter *n* must satisfy the condition $start+n \leq 1148$.
2. The calibration data structure contains several data items which are neither integer multiples of bytes nor aligned on byte edges. Though, in principle, every byte offset and length values in the given ranges are allowed, it is strongly recommended to specify a *start* parameter which points to the start of a data item and to specify a length parameter *n* which does not cut off the dump in the middle of a data item.
3. For a detailed description of the STR's complete calibration data structure see section 12.2.
4. In response to this command the STR sends a TM_CALIBRATION_DUMP packet.

6.8.13 TC_DUMP_PREPROCESSOR – DUMP PREPROCESSOR CONFIGURATION

Description:	The TC_DUMP_PREPROCESSOR command requests to dump the preprocessor configuration of the STR.
PRID:	see section 5.1
PCAT:	12 (telecommand)
Service type:	221
Service Subtype:	21
Duration Type:	short
STR modes accepting the TC:	STANDBY, AAD, NAT,PHOTO
Length of the application data:	0

Note:

1. In response to this command the STR sends a TM_PREPROCESSOR_DUMP packet.

6.8.14 TC_DUMP_ALGORITHM – DUMP ALGORITHM CONTROL PARAMETERS

Description: The TC_DUMP_ALGORITHM command requests to dump the algorithm control parameters of the STR.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 221
Service Subtype: 22
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT,PHOTO
Length of the application data: 0

Note:

1. In response to this command the STR sends a TM_ALGORITHM_DUMP packet.

6.8.15 TC_DUMP_GSC– DUMP PART OF THE GSC TABLE

Description: The TC_DUMP_GSC command requests to dump a specified part of the Guide Star Catalogue.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type : 221
Service Subtype : 23
Duration Type : long
STR modes accepting the TC: STANDBY, AAD, NAT,PHOTO
Length of the application data: 6 byte

Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
start	index (0-based) of the first catalog entry to be dumped.	0	UI16	2	0..3999
nToDump	total number of catalog entries to be dumped	2	UI16	2	1..4000 (see note 2)

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
maxDumpSizeInBytes	maximum amount per dump packet in bytes	4	UI16	2	≥16+4 (see note 3)

Notes:

1. In response to this command the STR sends one or more TM_GSC_DUMP packets.
2. The count parameter *nToDump* must fulfil the condition *start*+*nToDump* ≤ 4000.
3. The length parameter *maxDumpSizeInBytes* specifies the maximum amount to be dumped with each dump packet; it defines the amount of GSC data to be dumped and not the overall packet size of the TM_GSC_DUMP packet. If *maxDumpSizeInBytes* exceeds the limit min(4096,*nToDump**16+4) it will automatically be set to this limit. If downlink bandwidth is a concern, *maxDumpSizeInBytes* has to be restricted accordingly by the caller.
4. The total dump range *start*..*start*+*nToDump*-1 will be split into pieces each providing up to (*maxDumpSizeInBytes*–4)/16 GSC entries (the last piece may contain less) and each piece will be delivered with a separate TM_GSC_DUMP packet. The TM_GSC_DUMP packets resulting from a TC_DUMP_GSC command will be generated with 10Hz update rate. See also information in section 2.3.1 on telemetry update rates.

6.8.16 TC_DUMP_GSC_INDICES – DUMP PART OF THE GSC INDEX TABLE

Description: The TC_DUMP_GSC_INDICES command requests to dump a specified part of the GSC tile index list.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type : 221
Service Subtype : 24
Duration Type : long
STR modes accepting the TC: STANDBY, AAD, NAT,PHOTO
Length of the application data: 3 byte

Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
start	tile number (0-based) of the first GSC Index entry to be dumped	0	UI8	1	0..215
nToDump	total number of GSC Index entries to be dumped	1	UI8	1	1..216 (see note 2)

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
maxDumpSizeInBytes	maximum amount per dump packet in bytes	2	UI8	1	≥2+2 (see note 3)

Notes:

1. In response to this command the STR sends one or more TM_GSC_INDICES_DUMP packets
2. The count parameter *nToDump* must fulfil the condition *start*+*nToDump* ≤ 216.
3. The length parameter *maxDumpSizeInBytes* specifies the maximum amount to be dumped with each dump packet; it defines the amount of GSC Index data to be dumped and not the overall packet size of the TM_GSC_INDICES_DUMP packet. If *maxDumpSizeInBytes* exceeds the limit *nToDump**2+2 it will automatically be set to this limit. If downlink bandwidth is a concern, *maxDumpSizeInBytes* has to be restricted accordingly.
4. The specified dump range *start*..*start*+*nToDump*-1 will be split into pieces each providing up to (*maxDumpSizeInBytes*–2)/2 GSC Index entries (the last piece may contain less) and each piece will be delivered with a separate TM_GSC_INDICES_DUMP packet. The TM_GSC_INDICES_DUMP packets resulting from a TC_DUMP_GSC_INDICES command will be generated with 10Hz update rate. See also information in section 2.3.1 on telemetry update rates.

6.9 SERVICE 223

6.9.1 TC_ENABLE_MEMREAD – ENABLE TC(6,5) AND TC(6,9) FOR A SPECIFIED MEMID

Description: The TC_ENABLE_MEMREAD command is used to temporarily enable memory DUMP and CHECK commands for one of the I/O related memory areas of the STR.
PRID: see section 5.1
PCAT: must be set to 12 (telecommand)
Service type: 223
Service Subtype: 1
Duration Type: short
STR modes accepting the TC: any
Length of the application data: 4 byte
Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
MemID	memory id for which TC(6,5) and TC(6,9) commands shall be enabled	0	UI16	2	must be one of the I/O related MemIDs, see section 13
Timeout	timeout [s] for successful execution of TC(6,5) and TC(6,9) commands	2	UI16	2	0, ..., 2^{16}-1

Notes:

1. A timeout value of 0 means 'disable memory reads' to the memory associated with *MemID*, i.e. the timeout value 0 has the same effect as a TC_DISABLE_MEMREAD command for this *MemID*.
2. For the usage of this command see section 5.12.
3. This command executes successfully only for MemIDs which are assigned to I/O related address ranges (see section 13). For all other MemIDs, command execution fails.

6.9.2 TC_ENABLE_MEMWRITE – ENABLE TC(6,2) FOR A SPECIFIED MEMID

Description: The TC_ENABLE_MEMWRITE command is used to temporarily enable memory LOAD commands for a specified memory.
PRID: see section 5.1
PCAT: must be set to 12 (telecommand)
Service type: 223
Service Subtype: 2
Duration Type: short
STR modes accepting the TC: BOOT, STANDBY
Length of the application data: 4 byte

Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
MemID	memory id for which TC(6,2) commands shall be enabled	0	UI16	2	see section 13
Timeout	timeout [s] for successful execution of TC(6,2) commands	2	UI16	2	0, ..., 2^{16}-1

Notes:

1. Execution of this command fails if the memory associated with the specified *MemID* is read-only (see section 13).
2. If *MemID* 71 (test application upload area in RAM) is specified this command enables not only TC(6,2) commands for this *MemID* but additionally the TC(223, 10) command.
3. If *MemID* 11 (application S/W in EEPROM) is specified this command enables not only TC(6,2) commands for this *MemID* but additionally the TC(223, 11) command.
4. A timeout value of 0 means 'disable memory writes' to the memory associated with *MemID*, i.e. the timeout value 0 has the same effect as a TC_DISABLE_MEMWRITE command for this *MemID*.
5. For the usage of this command see section 5.12.
6. For EEPROM related MemIDs this command does **not** affect the H/W related write protection of the EEPROM.

6.9.3 TC_DISABLE_MEMREAD – DISABLE TC(6,5) AND TC(6,9) FOR A SPECIFIED MEMID

Description: The TC_DISABLE_MEMREAD command is used to disable memory DUMP and CHECK commands for one of the I/O related memory areas of the STR.
PRID: see section 5.1
PCAT: must be set to 12 (telecommand)
Service type: 223
Service Subtype: 3
Duration Type: short
STR modes accepting the TC: any
Length of the application data: 2 byte
Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Format	Length (byte)	Range
MemID	memory id for which TC(6,5) and TC(6,9) commands shall be disabled	UI16	2	must be one of the I/O related MemIDs, see section 13

Note:

1. For the usage of this command see section 5.12.
2. This command executes successfully only for MemIDs which are assigned to I/O related address ranges (see section 13). For all other MemIDs, command execution fails.

6.9.4 TC_DISABLE_MEMWRITE – DISABLE TC(6,2) FOR A SPECIFIED MEMID

Description: The TC_DISABLE_MEMWRITE command is used to disable memory LOAD commands for a specified memory.
PRID: see section 5.1
PCAT: must be set to 12 (telecommand)
Service type: 223
Service Subtype: 4
Duration Type: short
STR modes accepting the TC: BOOT, STANDBY
Length of the application data: 2 byte
Structure of the TC application data field:

Argument Name	Description	Format	Length (byte)	Range
MemID	memory id for which TC(6,2) commands shall be disabled	UI16	2	see section 13

Notes:

1. Execution of this command fails if the memory associated with the specified *MemID* is read-only (see section 13).
2. If *MemID* 71 (test application upload area in RAM) is specified this command disables not only TC(6,2) commands for this *MemID* but additionally the TC(223, 10) command.
3. If *MemID* 11 (application S/W in EEPROM) is specified this command disables not only TC(6,2) commands for this *MemID* but additionally the TC(223, 11) command.
4. For the usage of this command see section 5.12.
5. For EEPROM related MemIDs this command does **not** affect the H/W related write protection of the EEPROM.

6.9.5 TC_COPY_EE2TAPP – COPY THE APSW FROM EEPROM TO THE TEST APPLICATION AREA

Description:	The TC_COPY_EE2TAPP command is used to copy the application S/W (incl. calibration data and star catalog) from EEPROM to the test application upload area in RAM.
PRID:	see section 5.1
PCAT:	must be set to 12 (telecommand)
Service type :	223
Service Subtype :	10
Duration Type :	long
STR modes accepting the TC:	BOOT, STANDBY
Length of the application data:	0

Notes:

1. Command execution fails if MemID 71 or MemID 81 (see section 13) is currently not enabled for writing, i.e. two TC(223,2) – for MemID 71 and 81 – must precede a TC(223,10).

2. For the usage of this command see also section 5.12.

3. During execution of this command the STR will generate progress events of type EV_EEREAD_INPROGRESS at a 0.1Hz rate (1 event packet every 10s). See also information in section 2.3.1 on telemetry update rates.

6.9.6 TC_COPY_TAPP2EE – COPY THE APSW FROM THE TEST APPLICATION AREA TO EEPROM

Description:	The TC_COPY_TAPP2EE command is used to copy the test application (incl. calibration data and star catalog) which is currently contained in the test application upload area to EEPROM.
PRID:	see section 5.1
PCAT:	must be set to 12 (telecommand)
Service type :	223
Service Subtype :	11
Duration Type :	long
STR modes accepting the TC:	BOOT, STANDBY
Length of the application data:	0

Notes:

1. Command fails if MemID 11 or MemID 51 (see section 13) is currently not enabled for writing, i.e. two TC(223,2) – for MemID 11 and 51 – must precede a TC(223,11).
2. For the usage of this command see also section 5.12.
3. During execution of this command the STR will generate progress events of type EV_EEWRITE_INPROGRESS at a 0.1Hz rate (1 event packet every 10s). See also information in section 2.3.1 on telemetry update rates.

6.10 SERVICE 224

6.10.1 TC_ENABLE_TEMPERATURE – ENABLE/DISABLE THE TC_TEMPERATURE COMMAND

Description: The purpose of this command is to enable the TC_TEMPERATURE command which is treated as a critical command.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 224
Service Subtype: 1
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 2 byte

Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
Timeout	timeout [s] for successful execution of a TC(224,7) command	2	UI16	2	0, ..., 2^{16}-1

Note:

1. A timeout value of 0 means ‘disable the TC_TEMPERATURE’ command.

6.10.2 TC_PRECESSION – UPDATE PRECESSION CORRECTION PARAMETERS

Description: The TC_PRECESSION command specifies if precession correction shall be performed on the calculated attitude and for which date.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 224
Service Subtype: 4
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 3 bytes

Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
julianDate	Julian date, specified in number of days since the epoch date of the built-in star catalog; day #0 corresponds to JD 2451545	0	UI16	2		Julian days
enablePrecessionCorrection	if set to 1, precession correction will be performed using the date specified above if set to 0, no precession correction will be performed	2	B	1	0, 1	

Note:

1. If this command is received in STANDBY or PHOTO mode, it has no immediate effect. The parameters will only be copied for later usage in AAD and NAT mode.

6.10.3 TC_ABERRATION – UPDATE ABERRATION CORRECTION PARAMETERS

Description: The TC_ABBERATION command specifies if aberration correction shall be performed on the measured star positions and for which velocity vector.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 224
Service Subtype: 5

Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 7 bytes
Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
velocityVectorX	velocity vector (in heliocentric, inertial co ordinate frame) of the spacecraft (current value used for correction of the aberration).	0	SI16	2		$2^{-22}\ c_{vac}$
velocityVectorY		2	SI16	2		$2^{-22}\ c_{vac}$
velocityVectorZ		4	SI16	2		$2^{-22}\ c_{vac}$
enableAberrationCorrection	if set to 1, aberration correction will be performed using the velocity vector specified above if set to 0, no aberration correction will be performed	6	B	1	0, 1	

Notes:

1. The velocity vector will be ignored if the enableAberrationCorrection parameter is 0. Otherwise the vector (velocityVectorX,velocityVectorY,velocityVectorZ) must specify the S/C velocity in the sun centered inertial frame multiplied by the factor $2^{22}/\mathbf{c_{vac}}$, where $\mathbf{c_{vac}}$ denotes the vacuum speed of light (299792.458 km/s). For example, given a S/C velocity of magnitude 35 km/s. Then the vector delivered in the TC_ABERRATION command should have the magnitude $35 \cdot 2^{22}/299792.458$ which is about the value 490, i.e. the length of the vector (velocityVectorX,velocityVectorY,velocityVectorZ) should be about 490.

2. If this command is received in STANDBY or PHOTO mode, it has no immediate effect. The parameters will only be copied for later usage in AAD and NAT mode.

6.10.4 TC_RATE_LIMIT – SET UPPER LIMIT FOR X AND Y COMPONENTS OF THE RATE

Description: The TC_RATE_LIMIT command specifies new upper limits for the angular rates about x and y.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 224
Service Subtype: 6
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 1 byte

Structure of the application data field:

Argument Name	Description	Format	Length (byte)	Range	Unit
updatedRateLimit	new expected maximum of the absolutes of the x and y components of the rate; a value of 0 means: use defaultRateLimit	UI8	1	0…255	2^{-5} deg/s

Note:

1. If this command is received in STANDBY or PHOTO mode, it has no immediate effect. The parameters will only be copied for later usage in AAD and NAT mode.

6.10.5 TC_TEMPERATURE – SPECIFY TARGET DETECTOR TEMPERATURE

Description: The TC_TEMPERATURE command specifies a new target temperature for the APS detector.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 224
Service Subtype: 7
Duration Type: short
STR modes accepting the TC: STANDBY, AAD, NAT, PHOTO
Length of the application data: 2 byte
Structure of the application data field:

Argument Name	Description	Format	Length (byte)	Unit
targetTemperature	Target temperature for the detector: if < -30°C: peltier cooler operates at maximum power if > +40°C: peltier cooler is switched off	SI16	2	10^{-1}°C

Notes:

1. If the *targetTemperature* value is ≤+40°C, i.e. if the command does not request to switch OFF the peltier cooler, command execution fails if it is currently disabled. To enable the TC_TEMPERATURE command the TC_ENABLE_TEMPERATURE command must be used.

2. If the *targetTemperature* value requests to switch OFF the peltier cooler, the command execution will always succeed regardless of its current enable state.

6.10.6 TC_ZERO_COUNTERS – SET COUNTERS TO ZERO

Description: The TC_ZERO_COUNTERS command is used to reset the error counters maintained by the STR.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 224
Service Subtype: 8
Duration Type: short
STR modes accepting the TC: any
Length of the application data: 1 byte
Structure of the application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
edacErrors	If set, the numEdacErrors counter and the ramError flag are set to 0	0	B	1	0, 1	
interfaceError	If set, the interfaceError flag is set to 0	1	B	1	0, 1	
watchdogError	If set, the watchdogError flag is set to 0	2	B	1	0, 1	
synchronizationError	If set, the synchronizationError flag is set to 0	3	B	1	0, 1	
telecommandErrors	If set, the numTcErrors counter is set to 0	4	B	1	0, 1	
lostTracking	If set, the numLostTracking counter is set to 0	5	B	1	0, 1	
spare		6	B	2		

6.10.7 TC_TIME_SHIFT_THRESHOLD – SET THRESHOLD FOR TIME SHIFT EVENT

Description: The TC_TIME_SHIFT_THRESHOLD is used to specify the threshold for the generation of event EV_TIME_SHIFT.
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type: 224
Service Subtype: 9
Duration Type: short

STR modes accepting the TC: any
Length of the application data: 1 byte
Structure of the application data field:

Argument Name	Description	Format	Length (byte)	Range
shiftThresh	Threshold [ms] for generation of EV_TIME_SHIFT	UI8	1	1, …, 127

6.10.8 TC_MAX_DUMPSIZE – SET AMOUNT OF DATA PER DUMP PACKET

Description: The TC_MAX_DUMPSIZE is used to specify the maximum size of source data for a TM_MEM_DUMP (default: 42=10+32).
PRID: see section 5.1
PCAT: 12 (telecommand)
Service type : 224
Service Subtype : 10
Duration Type : short
STR modes accepting the TC: any
Length of the application data: 2 byte
Structure of the TC application data field (offsets are wrt. The start of the application data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
maxDumpSizeInBytes	amount per dump packet	0	UI16	2	≥ SAU+10 (see note 1)	Byte

Notes:

1. If *maxDumpSizeInBytes* exceeds the limit 4096 it will automatically be set to this limit. The total dump range specified in a TC_DUMP_MEMORY will be split into pieces each covering up to (*maxDumpSizeInBytes*–10)*/SAU* memory units from the specified MemID (the last piece may contain less) and each piece will be delivered with a separate TM_MEM_DUMP packet. The *maxDumpSizeInBytes* represents the size of source data in the TM_MEM_DUMP packet. It contains the additional information about the dump data as well as the dump data itself. If downlink bandwidth is a concern *maxDumpSizeInBytes* has to be restricted accordingly by the caller.

7 TM STRUCTURES

This chapter contains the detailed descriptions of all TM packets supported by the STR.

Each TM packet is described by a dedicated subsection which is titled by the name (mnemonic) of the TM packet.

Each description lists the packet's PRID and PCAT values, its Service and Subservice values, and the length, structure and contents of the packet's source data field.

7.1 SERVICE 1

7.1.1 TM_ACK_VERISUCC – TC ACCEPTANCE REPORT – SUCCESS

Description: This is the standard TM(1,1) acknowledge packet of the STR indicating that the corresponding TC packet was accepted.
PRID: see section 5.1
PCAT: 1 (acknowledge)
Service type: 1
Service Subtype: 1
Length of source data: 4

Structure of the source data field (offsets are wrt. The start of the source data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)
pktID	copy of the 16 bit packet ID field taken from the TC packet header	0	UI16	2
seqCtrl	copy of the 16 bit sequence control field taken from the TC packet header	2	UI16	2

Note:

1. This kind of TM packet is the response on any TC packet if all acceptance checks for the TC packet were successful and the acceptance ACK flag in the TC data field header was set.

7.1.2 TM_ACK_VERIFAIL – TC ACCEPTANCE REPORT – FAILURE

Description: This is the standard TM(1,2) acknowledge packet of the STR providing the information why the corresponding TC was not accepted by the STR.
PRID: see section 5.1
PCAT: 1 (acknowledge)
Service type: 1
Service Subtype: 2
Length of source data: 6+k
Structure of the source data field (offsets are wrt. The start of the source data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)
pktID	copy of the 16 bit packet ID field taken from the TC packet header	0	UI16	2
seqCtrl	copy of the 16 bit sequence control field taken from the TC packet header	2	UI16	2
fid	fault ID indicating the error which was detected during the acceptance checks	4	UI16	2
…	complementary information (depends on the FID value, see Note 2)	6	any	k

Note:

1. This kind of TM packet is the response on any TC packet if the acceptance checks for the TC packet failed.
2. For detailed descriptions of the format of complementary information provided with the FIDs used by the STR see chapter 10.

7.1.3 TM_ACK_EXECSUCC – TC EXECUTION COMPLETION REPORT – SUCCESS

Description: This is the standard TM(1,7) acknowledge packet of the STR indicating that the corresponding TC was successfully executed.
PRID: see section 5.1
PCAT: 1 (acknowledge)
Service type: 1
Service Subtype: 7
Length of source data: 4
Structure of the source data field (offsets are wrt. The start of the source data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)
pktID	copy of the 16 bit packet ID field taken from the TC packet header	0	UI16	2
seqCtrl	copy of the 16 bit sequence control field taken from the TC packet header	2	UI16	2

Note:

1. This kind of TM packet is the response on any TC packet if the excution of the command was successful and the execution ACK flag in the TC data field header was set.

7.1.4 TM_ACK_EXECFAIL – TC EXECUTION COMPLETION REPORT – FAILURE

Description: This is the standard TM(1,8) acknowledge packet of the STR providing the information why the execution of the corresponding TC failed.
PRID: see section 5.1
PCAT: 1 (acknowledge)
Service type: 1
Service Subtype: 8
Length of source data: 6+k

Structure of the source data field (offsets are wrt. The start of the source data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)
pktID	copy of the 16 bit packet ID field taken from the TC packet header	0	UI16	2
seqCtrl	copy of the 16 bit sequence control field taken from the TC packet header	2	UI16	2
fid	fault ID indicating the error which was detected during command execution	4	UI16	2
…	complementary information (depends on the FID value, see Note 2)	6	any	k

Note:

1. This kind of TM packet is the response on any TC packet if the execution of the command failed.
2. For detailed descriptions of the format of complementary information provided with the FIDs used by the STR see chapter 10.

7.2 SERVICE 3

7.2.1 TM_ADB – ATTITUDE DATA BLOCK

Description: The TM_ADB packet delivers the attitude quaternion and angular rates determined by the STR.
PRID: see section 5.1
PCAT: 6 (S/C Ancillary Data)
Service type: 3
Service Subtype: 25
Length of source data: 39 bytes
Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
SID	the ID of this structure	0	UI	8	105 (note: SID selected acc. AD1, section B2.6)	
qv1	vector part of attitude quaternion, x component	8	SI	32		2^{-30}
qv2	vector part of attitude quaternion, y component	40	SI	32		2^{-30}
qv3	vector part of attitude quaternion, z component	72	SI	32		2^{-30}
qs	scalar part of attitude quaternion	104	SI	32		2^{-30}
rateX	angular rate in BRF	136	SI	16		2^{-11} deg/s
rateY	angular rate in BRF	152	SI	16		2^{-11} deg/s
rateZ	angular rate in BRF	168	SI	16		2^{-11} deg/s
centerOfIntegrationTimeStamp	Time stamp of the center of integration associated with the attitude.	184	T	48		2^{-16}s
julianDate	Julian date used for precession correction (if enabled); specified in number of days since the epoch date of the built-in star catalog; day #0 corresponds to JD 2451545	232	UI	16		Julian days
velocityVectorSciX	If aberration correction is enabled: X coordinate of velocity vector (in SCI) Otherwise: 0	248	SI	16	see Note 1	$2^{-22}c_{vac}$

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
velocityVectorSciY	If aberration correction is enabled: Y coordinate of velocity vector (in SCI) Otherwise: 0	264	SI	16		$2^{-22}c_{vac}$
velocityVectorSciZ	If aberration correction is enabled: Z coordinate of velocity vector (in SCI) Otherwise: 0	280	SI	16		$2^{-22}c_{vac}$
attitudeQuality	attitude quality	296	E	3	see AttitudeQuality enumeration	
isPrecessionCorrected	if set to 1: precession correction has been applied	299	B	1	0, 1	
isAberrationCorrected	if set to 1: aberration correction has been applied	300	B	1	0, 1	
rateQuality	rate quality	301	E	2	see RateQuality enumeration	
isValidRate	if set to 1: the rate information is valid (i.e. derived from current measurements)	303	B	1	0, 1	
attitudeQualityIndex	sum of single-star qualities of attitude stars	304	UI	8	≤240	

Note:

1. The velocity vector reported by the 3 parameters velocityVectorSciX, velocityVectorSciY, and velocityVectorSciZ is exactly that one which is used for the aberration correction. In case of enabled aberration correction it was most recently provided to the STR, either by a TC_ACQUIRE command, or by a TC_ABERRATION command. In case of disabled aberration correction the values are 0 (the same as aberration correction with velocityVector=(0,0,0).

2. The julianDate is exactly that one which is used for the precession correction. The disabled precession correction is the same as precession correction for julianDate=0.

7.2.2 TM_SDB – STATUS AND HEALTH DATA BLOCK

Description: The TM_SDB packet delivers the housekeeping data of the STR.
PRID: see section 5.1
PCAT: 4 (HK TM)
Service type: 3
Service Subtype: 25
Length of source data: 45 bytes

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
SID	the ID of this structure	0	UI	8	1 (note: SID selected acc. AD1, section B2.6)	
cycle	STR internal cycle counter	8	UI	16		
cycleStartTimeStamp	Time stamp of the cycle start	24	T	48		2^{-16}s
cycleTriggerSource	0: timer; 1: preprocessor; 2: external	72	UI	2		
opMode	STR operating mode	74	E	3	see OpMode enumeration	
isStreakMode	1: streak processing mode is active	77	B	1		
tecMode	Thermo electric cooler mode	78	E	2	see TecMode enumeration	
	Hardware Status					
EepromErrApp	1: error in application part on EEPROM	80	B	1		
EepromErrCal	1: error in calibration data part on EEPROM	81	B	1		
ramError	1: RAM failure; see numEdacErrors for more information; can be reset by TC_ZERO_COUNTERS telecommand	82	B	1		
interfaceError	1: Failure during initialization of the operational interface; can be reset by TC_ZERO_COUNTERS telecommand	83	B	1		
watchdogError	1: Reboot was caused by the watchdog timer; can be reset by TC_ZERO_COUNTERS telecommand	84	B	1		
synchronizationError	1: External synchronization failed; can be reset by TC_ZERO_COUNTERS telecommand	85	B	1		
timingError	1: processing exceeded timing constraints	86	B	1		
fifoError	1: the hardware FIFO capacity has been exceeded (e.g. too many objects detected in FullFrame mode)	87	B	1		
targetTemperature	target temperature for temperature control	88	SI	16		0.1°C
temperatureDetector	detector temperature	104	SI	16		0.1°C
temperatureOptics	optics temperature	120	SI	16		0.1°C
temperatureHousing	housing temperature	136	SI	16		0.1°C
numEdacErrors	Number of errors detected by RAM EDAC; can be reset by TC_ZERO_COUNTERS telecommand	152	UI	8		

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
syncSource	The source selected for external synchronization	160	E	2	see SyncSource enumeration	
secondsSinceTimeSync	The number of seconds since the latest received time synchronization; saturates at 63 seconds	162	UI	6		s
	Processing Results					
meanBackground	mean background	168	SI	16		
numObjectsDetected	number of detected objects (FullFrame mode only)	184	UI	16		
numObjectsAcquired	number of acquired objects	200	UI	16		
numSinglePixelsRemoved	number of single pixels removed by h/w (FullFrame mode only)	216	UI	16		
numEoDetected	number of extended objects detected by h/w (FullFrame mode only)	232	UI	16		
numStarsTrackable	number of trackable stars	248	UI	8		
numStarsTracked	number of stars tracked	256	UI	8		
numStarsUsedForRate	number of stars used for rate determination	264	UI	8		
numStarsIdentified	number of identified stars	272	UI	8		
numStarsUsedForAttitude	number of stars used for attitude determination	280	UI	8		
numLostTracking	number of fall-backs from NAT to AAD; can be reset by TC_ZERO_COUNTERS telecommand	288	UI	8		
spare		296	UI	16		
spare		312	UI	8		
numTcErrors	Number of telecommand errors; can be reset by TC_ZERO_COUNTERS telecommand	320	UI	8		
AttResult	last result of attitude determination	328	E	4	see AttResult enumeration	
IdResult	last result of star identification	332	E	4	see IdResult enumeration	
	Control Parameters					
tInt	integration time	336	UI	8		1ms
offset	binarisation offset	344	UI	16		detector LSB

7.2.3 TM_TDB – TRACKER DATA BLOCK

Description: The TM_TDB packet delivers the detailed tracking status of the STR.
PRID: see section 5.1
PCAT: 6 (S/C Ancillary Data)
Service type: 3
Service Subtype: 25
Length of source data: 147 bytes

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (bit)	Format	Length (bit)	RANGE	Unit
SID	the ID of this structure	0	UI	8	106 (note: SID selected acc. AD1, B2.6)	
S1_TA	tangent value of star 1 alpha coordinate	8	SI	24		2^{-24}
S1_TB	tangent value of star 1 beta coordinate	30	SI	24		2^{-24}
S2_TA	tangent value of star 2 alpha coordinate	56	SI	24		2^{-24}
S2_TB	tangent value of star 2 beta coordinate	80	SI	24		2^{-24}
S3_TA	tangent value of star 3 alpha coordinate	104	SI	24		2^{-24}
S3_TB	tangent value of star 3 beta coordinate	128	SI	24		2^{-24}
S4_TA	tangent value of star 4 alpha coordinate	152	SI	24		2^{-24}
S4_TB	tangent value of star 4 beta coordinate	176	SI	24		2^{-24}
S5_TA	tangent value of star 5 alpha coordinate	200	SI	24		2^{-24}
S5_TB	tangent value of star 5 beta coordinate	224	SI	24		2^{-24}
S6_TA	tangent value of star 6 alpha coordinate	248	SI	24		2^{-24}
S6_TB	tangent value of star 6 beta coordinate	272	SI	24		2^{-24}
S7_TA	tangent value of star 7 alpha coordinate	296	SI	24		2^{-24}
S7_TB	tangent value of star 7 beta coordinate	320	SI	24		2^{-24}
S8_TA	tangent value of star 8 alpha coordinate	344	SI	24		2^{-24}
S8_TB	tangent value of star 8 beta coordinate	368	SI	24		2^{-24}
S9_TA	tangent value of star 9 alpha coordinate	392	SI	24		2^{-24}
S9_TB	tangent value of star 9 beta coordinate	416	SI	24		2^{-24}

Name	Description	Offset (bit)	Format	Length (bit)	RANGE	Unit
S10_TA	tangent value of star 10 alpha coordinate	440	SI	24		2^{-24}
S10_TB	tangent value of star 10 beta coordinate	464	SI	24		2^{-24}
S11_TA	tangent value of star 11 alpha coordinate	488	SI	24		2^{-24}
S11_TB	tangent value of star 11 beta coordinate	512	SI	24		2^{-24}
S12_TA	tangent value of star 12 alpha coordinate	536	SI	24		2^{-24}
S12_TB	tangent value of star 12 beta coordinate	560	SI	24		2^{-24}
S13_TA	tangent value of star 13 alpha coordinate	584	SI	24		2^{-24}
S13_TB	tangent value of star 13 beta coordinate	608	SI	24		2^{-24}
S14_TA	tangent value of star 14 alpha coordinate	632	SI	24		2^{-24}
S14_TB	tangent value of star 14 beta coordinate	656	SI	24		2^{-24}
S15_TA	tangent value of star 15 alpha coordinate	680	SI	24		2^{-24}
S15_TB	tangent value of star 15 beta coordinate	704	SI	24		2^{-24}
S16_TA	tangent value of star 16 alpha coordinate	728	SI	24		2^{-24}
S16_TB	tangent value of star 16 beta coordinate	752	SI	24		2^{-24}
S1_CNR	star 1 catalog number in on-board GSC; 0 means: not identified	776	UI	16	0,...,4000	
S2_CNR	star 2 catalog number in on-board GSC; 0 means: not identified	792	UI	16	0,...,4000	
S3_CNR	star 3 catalog number in on-board GSC; 0 means: not identified	808	UI	16	0,...,4000	
S4_CNR	star 4 catalog number in on-board GSC; 0 means: not identified	824	UI	16	0,...,4000	
S5_CNR	star 5 catalog number in on-board GSC; 0 means: not identified	840	UI	16	0,...,4000	
S6_CNR	star 6 catalog number in on-board GSC; 0 means: not identified	856	UI	16	0,...,4000	
S7_CNR	star 7 catalog number in on-board GSC; 0 means: not identified	872	UI	16	0,...,4000	
S8_CNR	star 8 catalog number in on-board GSC; 0 means: not identified	888	UI	16	0,...,4000	
S9_CNR	star 9 catalog number in on-board GSC; 0 means: not identified	904	UI	16	0,...,4000	
S10_CNR	star 10 catalog number in on-board GSC; 0 means: not identified	920	UI	16	0,...,4000	
S11_CNR	star 11 catalog number in on-board GSC; 0 means: not identified	936	UI	16	0,...,4000	
S12_CNR	star 12 catalog number in on-board GSC; 0 means: not identified	952	UI	16	0,...,4000	
S13_CNR	star 13 catalog number in on-board GSC; 0 means: not identified	968	UI	16	0,...,4000	
S14_CNR	star 14 catalog number in on-board GSC; 0 means: not identified	984	UI	16	0,...,4000	
S15_CNR	star 15 catalog number in on-board GSC; 0 means: not identified	1000	UI	16	0,...,4000	
S16_CNR	star 16 catalog number in on-board GSC; 0 means: not identified	1016	UI	16	0,...,4000	
S1_MAG	star 1 instrumental magnitude	1032	SI	8		10^{-1} mi
S2_MAG	star 2 instrumental magnitude	1040	SI	8		10^{-1} mi

Name	Description	Offset (bit)	Format	Length (bit)	RANGE	Unit
S3_MAG	star 3 instrumental magnitude	1048	SI	8		10^{-1} mi
S4_MAG	star 4 instrumental magnitude	1056	SI	8		10^{-1} mi
S5_MAG	star 5 instrumental magnitude	1064	SI	8		10^{-1} mi
S6_MAG	star 6 instrumental magnitude	1072	SI	8		10^{-1} mi
S7_MAG	star 7 instrumental magnitude	1080	SI	8		10^{-1} mi
S8_MAG	star 8 instrumental magnitude	1088	SI	8		10^{-1} mi
S9_MAG	star 9 instrumental magnitude	1096	SI	8		10^{-1} mi
S10_MAG	star 10 instrumental magnitude	1104	SI	8		10^{-1} mi
S11_MAG	star 11 instrumental magnitude	1112	SI	8		10^{-1} mi
S12_MAG	star 12 instrumental magnitude	1120	SI	8		10^{-1} mi
S13_MAG	star 13 instrumental magnitude	1128	SI	8		10^{-1} mi
S14_MAG	star 14 instrumental magnitude	1136	SI	8		10^{-1} mi
S15_MAG	star 15 instrumental magnitude	1144	SI	8		10^{-1} mi
S16_MAG	star 16 instrumental magnitude	1152	SI	8		10^{-1} mi
S1_VALID	if set to 1 the star 1 data are valid	1160	B	1	0, 1	
S2_VALID	if set to 1 the star 2 data are valid	1161	B	1	0, 1	
S3_VALID	if set to 1 the star 3 data are valid	1162	B	1	0, 1	
S4_VALID	if set to 1 the star 4 data are valid	1163	B	1	0, 1	
S5_VALID	if set to 1 the star 5 data are valid	1164	B	1	0, 1	
S6_VALID	if set to 1 the star 6 data are valid	1165	B	1	0, 1	
S7_VALID	if set to 1 the star 7 data are valid	1166	B	1	0, 1	
S8_VALID	if set to 1 the star 8 data are valid	1167	B	1	0, 1	
S9_VALID	if set to 1 the star 9 data are valid	1168	B	1	0, 1	
S10_VALID	if set to 1 the star 10 data are valid	1169	B	1	0, 1	
S11_VALID	if set to 1 the star 11 data are valid	1170	B	1	0, 1	
S12_VALID	if set to 1 the star 12 data are valid	1171	B	1	0, 1	
S13_VALID	if set to 1 the star 13 data are valid	1172	B	1	0, 1	
S14_VALID	if set to 1 the star 14 data are valid	1173	B	1	0, 1	
S15_VALID	if set to 1 the star 15 data are valid	1174	B	1	0, 1	
S16_VALID	if set to 1 the star 16 data are valid	1175	B	1	0, 1	

7.2.4 TM_PDB – PIXEL DATA BLOCK

Description: The TM_PDB packets are used to deliver the pixel raw data of up to 16 detector windows.
PRID: see section 5.1
PCAT: 2 (Diagnostic TM)
Service type: 3
Service Subtype: 26
Length of source data: 343 bytes

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
SID	the ID of this structure	0	UI	8	188..203	
xMin	Minimum X coordinate	8	UI	10	0, ..., 1023	pixel
xWidth	Width of the window	18	UI	6	9, 11, 15	pixel
yMin	Minimum Y coordinate	24	UI	10	0, ..., 1023	pixel
yHeight	Height of the window	34	UI	6	9, 11, 15	pixel
pv0	p(xMin,yMin): pixel value at 1st window position, increased by pixelOffsetPdb (see TM_EDB)	40	UI	12	0, ..., 4095	detector LSB
pv1	p(xMin+1,yMin): pixel value at 2nd window position, increased by pixelOffsetPdb (see TM_EDB)	52	UI	12	0, ..., 4095	detector LSB
...	...	...	...	...	...	...
pv224	pixel value at 225th window position, increased by pixelOffsetPdb (see TM_EDB)	2728	UI	12	0, …, 4095	detector LSB
spare	unused bits	2740		4		

Note:

1. This TM packet provides the pixel values of a detector window having the size of 9x9, 11x11, or 15x15 pixels, where the actual window size is given by xWidth and yHeight.

   Note that the window size is 15*15 for TM_PDB created during PHOTO mode.

2. The pixel values of the window are delivered row by row in ascending row order, i.e. starting with the pixel values of row yMin, followed by the pixel values of row yMin+1, a.s.o. up to the pixel values of row yMin+yHeight-1. Within each row the pixel values are delivered in ascending column order, i.e. starting with the pixel value of column xMin, followed by the pixel value of column xMin+1, a.s.o. up to the pixel value of column xMin+xWidth-1.

3. If the window size is 9x9 then the 81 pixel values will be delivered in pv0 to pv80 and the remaining fields pv81 to pv224 will be unused.

4. If the window size is 11x11 then the 121 pixel values will be delivered in pv0 to pv120 and the remaining fields pv121 to pv224 will be unused.

7.2.5 TM_EDB – ENGINEERING DATA BLOCK FOR SENSOR CHECKOUT

Description: The TM_EDB packet delivers engineering data of the STR.
PRID: see section 5.1
PCAT: 2 (Diagnostic TM)
Service type: 3
Service Subtype: 26
Length of source data: 423 bytes
Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
SID	the ID of this structure	0	UI	8	128	
S1_x	x position in detector coordinate frame	8	F	32		pixel
S1_y	y position in detector coordinate frame	40	F	32		pixel
S1_dx	x component of displacement vector normalized to 1 second	72	F	32		pixel
S1_dy	y component of displacement vector normalized to 1 second	104	F	32		pixel
S1_tc	total charge	136	UI	16		LSB
S1_bg	local background	152	SI	16		LSB
S2_x	x position in detector coordinate frame	168	F	32		pixel
S2_y	y position in detector coordinate frame	200	F	32		pixel
S2_dx	x component of displacement vector normalized to 1 second	232	F	32		pixel

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
S2_dy	y component of displacement vector normalized to 1 second	264	F	32		pixel
S2_tc	total charge	296	UI	16		LSB
S2_bg	local background	312	SI	16		LSB
S3_x	x position in detector coordinate frame	328	F	32		pixel
S3_y	y position in detector coordinate frame	360	F	32		pixel
S3_dx	x component of displacement vector normalized to 1 second	392	F	32		pixel
S3_dy	y component of displacement vector normalized to 1 second	424	F	32		pixel
S3_tc	total charge	456	UI	16		LSB
S3_bg	local background	472	SI	16		LSB
S4_x	x position in detector coordinate frame	488	F	32		pixel
S4_y	y position in detector coordinate frame	520	F	32		pixel
S4_dx	x component of displacement vector normalized to 1 second	552	F	32		pixel
S4_dy	y component of displacement vector normalized to 1 second	584	F	32		pixel
S4_tc	total charge	616	UI	16		LSB
S4_bg	local background	632	SI	16		LSB
S5_x	x position in detector coordinate frame	648	F	32		pixel
S5_y	y position in detector coordinate frame	680	F	32		pixel
S5_dx	x component of displacement vector normalized to 1 second	712	F	32		pixel
S5_dy	y component of displacement vector normalized to 1 second	744	F	32		pixel
S5_tc	total charge	776	UI	16		LSB
S5_bg	local background	792	SI	16		LSB
S6_x	x position in detector coordinate frame	808	F	32		pixel
S6_y	y position in detector coordinate frame	840	F	32		pixel
S6_dx	x component of displacement vector normalized to 1 second	872	F	32		pixel

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
S6_dy	y component of displacement vector normalized to 1 second	904	F	32		pixel
S6_tc	total charge	936	UI	16		LSB
S6_bg	local background	952	SI	16		LSB
S7_x	x position in detector coordinate frame	968	F	32		pixel
S7_y	y position in detector coordinate frame	1000	F	32		pixel
S7_dx	x component of displacement vector normalized to 1 second	1032	F	32		pixel
S7_dy	y component of displacement vector normalized to 1 second	1064	F	32		pixel
S7_tc	total charge	1096	UI	16		LSB
S7_bg	local background	1112	SI	16		LSB
S8_x	x position in detector coordinate frame	1128	F	32		pixel
S8_y	y position in detector coordinate frame	1160	F	32		pixel
S8_dx	x component of displacement vector normalized to 1 second	1192	F	32		pixel
S8_dy	y component of displacement vector normalized to 1 second	1224	F	32		pixel
S8_tc	total charge	1256	UI	16		LSB
S8_bg	local background	1272	SI	16		LSB
S9_x	x position in detector coordinate frame	1288	F	32		pixel
S9_y	y position in detector coordinate frame	1320	F	32		pixel
S9_dx	x component of displacement vector normalized to 1 second	1352	F	32		pixel
S9_dy	y component of displacement vector normalized to 1 second	1384	F	32		pixel
S9_tc	total charge	1416	UI	16		LSB
S9_bg	local background	1432	SI	16		LSB
S10_x	x position in detector coordinate frame	1448	F	32		pixel
S10_y	y position in detector coordinate frame	1480	F	32		pixel
S10_dx	x component of displacement vector normalized to 1 second	1512	F	32		pixel

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
S10_dy	y component of displacement vector normalized to 1 second	1544	F	32		pixel
S10_tc	total charge	1576	UI	16		LSB
S10_bg	local background	1592	SI	16		LSB
S11_x	x position in detector coordinate frame	1608	F	32		pixel
S11_y	y position in detector coordinate frame	1640	F	32		pixel
S11_dx	x component of displacement vector normalized to 1 second	1672	F	32		pixel
S11_dy	y component of displacement vector normalized to 1 second	1704	F	32		pixel
S11_tc	total charge	1736	UI	16		LSB
S11_bg	local background	1752	SI	16		LSB
S12_x	x position in detector coordinate frame	1768	F	32		pixel
S12_y	y position in detector coordinate frame	1800	F	32		pixel
S12_dx	x component of displacement vector normalized to 1 second	1832	F	32		pixel
S12_dy	y component of displacement vector normalized to 1 second	1864	F	32		pixel
S12_tc	total charge	1896	UI	16		LSB
S12_bg	local background	1912	SI	16		LSB
S13_x	x position in detector coordinate frame	1928	F	32		pixel
S13_y	y position in detector coordinate frame	1960	F	32		pixel
S13_dx	x component of displacement vector normalized to 1 second	1992	F	32		pixel
S13_dy	y component of displacement vector normalized to 1 second	2024	F	32		pixel
S13_tc	total charge	2056	UI	16		LSB
S13_bg	local background	2072	SI	16		LSB
S14_x	x position in detector coordinate frame	2088	F	32		pixel
S14_y	y position in detector coordinate frame	2120	F	32		pixel
S14_dx	x component of displacement vector normalized to 1 second	2152	F	32		pixel

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
S14_dy	y component of displacement vector normalized to 1 second	2184	F	32		pixel
S14_tc	total charge	2216	UI	16		LSB
S14_bg	local background	2232	SI	16		LSB
S15_x	x position in detector coordinate frame	2248	F	32		pixel
S15_y	y position in detector coordinate frame	2280	F	32		pixel
S15_dx	x component of displacement vector normalized to 1 second	2312	F	32		pixel
S15_dy	y component of displacement vector normalized to 1 second	2344	F	32		pixel
S15_tc	total charge	2376	UI	16		LSB
S15_bg	local background	2392	SI	16		LSB
S16_x	x position in detector coordinate frame	2408	F	32		pixel
S16_y	y position in detector coordinate frame	2440	F	32		pixel
S16_dx	x component of displacement vector normalized to 1 second	2472	F	32		pixel
S16_dy	y component of displacement vector normalized to 1 second	2504	F	32		pixel
S16_tc	total charge	2536	UI	16		LSB
S16_bg	local background	2552	SI	16		LSB
S1_state	state of the star	2568	E	3	see TrackStarState enumeration	
S1_quality	counts how often the star has been updated in the latest 15 cycles	2571	UI	4		
S1_preset	if set to 1 the star is preset by attitude tracking	2575	B	1		
S1_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2576	B	1		
S1_idsel	if set to 1 the star has been selected for identification	2577	B	1		
S1_attdet	if set to 1 star has been used for attitude determination	2578	B	1		
spare		2579	B	5		

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
S2_state	state of the star	2584	E	3	see TrackStarState enumeration	
S2_quality	counts how often the star has been updated in the latest 15 cycles	2587	UI	4		
S2_preset	if set to 1 the star is preset by attitude tracking	2591	B	1		
S2_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2592	B	1		
S2_idsel	if set to 1 the star has been selected for identification	2593	B	1		
S2_attdet	if set to 1 star has been used for attitude determination	2594	B	1		
spare		2595	B	5		
S3_state	state of the star	2600	E	3	see TrackStarState enumeration	
S3_quality	counts how often the star has been updated in the latest 15 cycles	2603	UI	4		
S3_preset	if set to 1 the star is preset by attitude tracking	2607	B	1		
S3_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2608	B	1		
S3_idsel	if set to 1 the star has been selected for identification	2609	B	1		
S3_attdet	if set to 1 star has been used for attitude determination	2610	B	1		
spare		2611	B	5		
S4_state	state of the star	2616	E	3	see TrackStarState enumeration	
S4_quality	counts how often the star has been updated in the latest 15 cycles	2619	UI	4		
S4_preset	if set to 1 the star is preset by attitude tracking	2623	B	1		
S4_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2624	B	1		
S4_idsel	if set to 1 the star has been selected for identification	2625	B	1		

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
S4_attdet	if set to 1 star has been used for attitude determination	2626	B	1		
spare		2627	B	5		
S5_state	state of the star	2632	E	3	see TrackStarState enumeration	
S5_quality	counts how often the star has been updated in the latest 15 cycles	2635	UI	4		
S5_preset	if set to 1 the star is preset by attitude tracking	2639	B	1		
S5_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2640	B	1		
S5_idsel	if set to 1 the star has been selected for identification	2641	B	1		
S5_attdet	if set to 1 star has been used for attitude determination	2642	B	1		
spare		2643	B	5		
S6_state	state of the star	2648	E	3	see TrackStarState enumeration	
S6_quality	counts how often the star has been updated in the latest 15 cycles	2651	UI	4		
S6_preset	if set to 1 the star is preset by attitude tracking	2655	B	1		
S6_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2656	B	1		
S6_idsel	if set to 1 the star has been selected for identification	2657	B	1		
S6_attdet	if set to 1 star has been used for attitude determination	2658	B	1		
spare		2659	B	5		
S7_state	state of the star	2664	E	3	see TrackStarState enumeration	
S7_quality	counts how often the star has been updated in the latest 15 cycles	2667	UI	4		
S7_preset	if set to 1 the star is preset by attitude tracking	2671	B	1		

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
S7_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2672	B	1		
S7_idsel	if set to 1 the star has been selected for identification	2673	B	1		
S7_attdet	if set to 1 star has been used for attitude determination	2674	B	1		
spare		2675	B	5		
S8_state	state of the star	2680	E	3	see TrackStarState enumeration	
S8_quality	counts how often the star has been updated in the latest 15 cycles	2683	UI	4		
S8_preset	if set to 1 the star is preset by attitude tracking	2687	B	1		
S8_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2688	B	1		
S8_idsel	if set to 1 the star has been selected for identification	2689	B	1		
S8_attdet	if set to 1 star has been used for attitude determination	2690	B	1		
spare		2691	B	5		
S9_state	state of the star	2696	E	3	see TrackStarState enumeration	
S9_quality	counts how often the star has been updated in the latest 15 cycles	2699	UI	4		
S9_preset	if set to 1 the star is preset by attitude tracking	2703	B	1		
S9_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2704	B	1		
S9_idsel	if set to 1 the star has been selected for identification	2705	B	1		
S9_attdet	if set to 1 star has been used for attitude determination	2706	B	1		
spare		2707	B	5		
S10_state	state of the star	2712	E	3	see TrackStarState enumeration	

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
S10_quality	counts how often the star has been updated in the latest 15 cycles	2715	UI	4		
S10_preset	if set to 1 the star is preset by attitude tracking	2719	B	1		
S10_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2720	B	1		
S10_idsel	if set to 1 the star has been selected for identification	2721	B	1		
S10_attdet	if set to 1 star has been used for attitude determination	2722	B	1		
spare		2723	B	5		
S11_state	state of the star	2728	E	3	see TrackStarState enumeration	
S11_quality	counts how often the star has been updated in the latest 15 cycles	2731	UI	4		
S11_preset	if set to 1 the star is preset by attitude tracking	2735	B	1		
S11_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2736	B	1		
S11_idsel	if set to 1 the star has been selected for identification	2737	B	1		
S11_attdet	if set to 1 star has been used for attitude determination	2738	B	1		
spare		2739	B	5		
S12_state	state of the star	2744	E	3	see TrackStarState enumeration	
S12_quality	counts how often the star has been updated in the latest 15 cycles	2747	UI	4		
S12_preset	if set to 1 the star is preset by attitude tracking	2751	B	1		
S12_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2752	B	1		
S12_idsel	if set to 1 the star has been selected for identification	2753	B	1		
S12_attdet	if set to 1 star has been used for attitude determination	2754	B	1		

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
spare		2755	B	5		
S13_state	state of the star	2760	E	3	see TrackStarState enumeration	
S13_quality	counts how often the star has been updated in the latest 15 cycles	2763	UI	4		
S13_preset	if set to 1 the star is preset by attitude tracking	2767	B	1		
S13_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2768	B	1		
S13_idsel	if set to 1 the star has been selected for identification	2769	B	1		
S13_attdet	if set to 1 star has been used for attitude determination	2770	B	1		
spare		2771	B	5		
S14_state	state of the star	2776	E	3	see TrackStarState enumeration	
S14_quality	counts how often the star has been updated in the latest 15 cycles	2779	UI	4		
S14_preset	if set to 1 the star is preset by attitude tracking	2783	B	1		
S14_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2784	B	1		
S14_idsel	if set to 1 the star has been selected for identification	2785	B	1		
S14_attdet	if set to 1 star has been used for attitude determination	2786	B	1		
spare		2787	B	5		
S15_state	state of the star	2792	E	3	see TrackStarState enumeration	
S15_quality	counts how often the star has been updated in the latest 15 cycles	2795	UI	4		
S15_preset	if set to 1 the star is preset by attitude tracking	2799	B	1		
S15_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2800	B	1		

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
S15_idsel	if set to 1 the star has been selected for identification	2801	B	1		
S15_attdet	if set to 1 star has been used for attitude determination	2802	B	1		
spare		2803	B	5		
S16_state	state of the star	2808	E	3	see TrackStarState enumeration	
S16_quality	counts how often the star has been updated in the latest 15 cycles	2811	UI	4		
S16_preset	if set to 1 the star is preset by attitude tracking	2815	UI	1		
S16_fpn	if set to 1 the fixed pattern noise has been removed from pixel data	2816	UI	1		
S16_idsel	if set to 1 the star has been selected for identification	2817	UI	1		
S16_attdet	if set to 1 star has been used for attitude determination	2818	UI	1		
spare		2819	B	5		
numObjectsL1	number of objects in object list 1	2824	UI	16		
numObjectsL2	number of objects in object list 2	2840	UI	16		
focusScale	effective focal length scale factor (measured_focal_Length / reference_focal_length)	2856	F	32		
tcScale	effective total charge (TC) scale factor (measured_TC / reference_TC)	2888	F	32		
rateLimit	the currently applicable rate limit	2920	F	32		deg/s
catchDistance	the current catch distance	2952	F	32		pixels
sgmQv1	from SGM: vector part of attitude	2984	SI	32		2^{-30}
sgmQv2	from SGM: vector part of attitude	3016	SI	32		2^{-30}
sgmQv3	from SGM: vector part of attitude	3048	SI	32		2^{-30}
sgmQs	from SGM: scalar part of attitude	3080	SI	32		2^{-30}
sgmRateX	from SGM: angular rate in BRF	3112	SI	16		2^{-11} deg/s
sgmRateY	from SGM: angular rate in BRF	3128	SI	16		2^{-11} deg/s
sgmRateZ	from SGM: angular rate in BRF	3144	SI	16		2^{-11} deg/s

Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
sgmCenterOfIntegrationTimeStamp	from SGM: Time stamp of the center of integration associated with the attitude.	3160	T	48		ms
sgmCycleStartTimeStamp	from SGM: Time stamp of the cycle start	3208	T	48		ms
timeAfterProcessing	time span from cycle start until after processing	3256	UI	16		ms
timeAfterTmHandler	time span from cycle start until after telemetry generation	3272	UI	16		ms
timeAfterCmdHandler	time span from cycle start until after command handling	3288	UI	16		ms
timePreparationReady	time span from cycle start until after preparation of next cycle	3304	UI	16		ms
tCycle	time duration of the cycle	3320	UI	16		ms
pixelOffsetPdb	subtract this value from the raw values in PDBs in order to get measured values	3336	SI	16		detector LSB
fifoLevel	the number of bytes stored in the FIFO at time of TM handling	3352	UI	16		bytes
tecPower	the current on-time of the PWM register for the TEC	3368	UI	8		PWM LSB
gain	ADC gain	3376	E	2	see Gain enumeration	
enableBLReadOut	if set to 1, black-level read-out is enabled	3378	B	1		
numberOfStreaks	number of streaks in full frame mode	3379	UI	5		

7.2.6 TM_SIDSTATES_DUMP – REPORT OF THE ENABLE STATES AND RATES OF ALL HK/DIAG PACKETS

Description: This report lists the current enable states and reporting periods of all STR's HK and Diagnostics packets.
PRID: see section 5.1
PCAT: 3 (table)
Service type: 3
Service Subtype: 129
Length of source data: 81 bytes
Structure of the source data field (offsets are wrt. The start of the source data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
numSID	number of SIDs maintained by the STR	0	UI8	1	20	
SID_SDB	SID of the TM_SDB packet	1	UI8	1	1 (see section 8)	
enaflag_SDB	reporting enable state of the TM_SDB packet	2	UI8	1	0 (disabled), 1 (enabled)	
period_SDB	reporting period of the TM_SDB packet	3	UI16	2	1…65535	
SID_ADB	SID of the TM_ADB packet	5	UI8	1	105 (see section 8)	
enaflag_ADB	reporting enable state of the TM_ADB packet	6	UI8	1	0 (disabled), 1 (enabled)	
period_ADB	reporting period of the TM_ADB packet	7	UI16	2	1…65535	
SID_TDB	SID of the TM_TDB packet	9	UI8	1	106 (see section 8)	
enaflag_TDB	reporting enable state of the TM_TDB packet	10	UI8	1	0 (disabled), 1 (enabled)	
period_TDB	reporting period of the TM_TDB packet	11	UI16	2	1…65535	
SID_EDB	SID of the TM_EDB packet	13	UI8	1	128 (see section 8)	
enaflag_EDB	reporting enable state of the TM_EDB packet	14	UI8	1	0 (disabled), 1 (enabled)	
period_EDB	reporting period of the TM_EDB packet	15	UI16	2	1…65535	
SID_PDB1	SID of the TM_PDB1 packet	17	UI8	1	188 (see section 8)	
enaflag_PDB1	reporting enable state of the TM_PDB1 packet	18	UI8	1	0 (disabled), 1 (enabled)	
period_PDB1	reporting period of the TM_PDB1 packet	19	UI16	2	1…65535	
SID_PDB2	SID of the TM_PDB2 packet	21	UI8	1	189 (see section 8)	
enaflag_PDB2	reporting enable state of the TM_PDB2 packet	22	UI8	1	0 (disabled), 1 (enabled)	
period_PDB2	reporting period of the TM_PDB2 packet	23	UI16	2	1…65535	
SID_PDB3	SID of the TM_PDB3 packet	25	UI8	1	190 (see section 8)	
enaflag_PDB3	reporting enable state of the TM_PDB3 packet	26	UI8	1	0 (disabled), 1 (enabled)	
period_PDB3	reporting period of the TM_PDB3 packet	27	UI16	2	1…65535	
SID_PDB4	SID of the TM_PDB4 packet	29	UI8	1	191 (see section 8)	
enaflag_PDB4	reporting enable state of the TM_PDB4 packet	30	UI8	1	0 (disabled), 1 (enabled)	
period_PDB4	reporting period of the TM_PDB4 packet	31	UI16	2	1…65535	
SID_PDB5	SID of the TM_PDB5 packet	33	UI8	1	192 (see section 8)	
enaflag_PDB5	reporting enable state of the TM_PDB5 packet	34	UI8	1	0 (disabled), 1 (enabled)	
period_PDB5	reporting period of the TM_PDB5 packet	35	UI16	2	1…65535	
SID_PDB6	SID of the TM_PDB6 packet	37	UI8	1	193 (see section 8)	
enaflag_PDB6	reporting enable state of the TM_PDB6 packet	38	UI8	1	0 (disabled), 1 (enabled)	
period_PDB6	reporting period of the TM_PDB6 packet	39	UI16	2	1…65535	
SID_PDB7	SID of the TM_PDB7 packet	41	UI8	1	194 (see section 8)	

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
enaflag_PDB7	reporting enable state of the TM_PDB7 packet	42	UI8	1	0 (disabled), 1 (enabled)	
period_PDB7	reporting period of the TM_PDB7 packet	43	UI16	2	1…65535	
SID_PDB8	SID of the TM_PDB8 packet	45	UI8	1	195 (see section 8)	
enaflag_PDB8	reporting enable state of the TM_PDB8 packet	46	UI8	1	0 (disabled), 1 (enabled)	
period_PDB8	reporting period of the TM_PDB8 packet	47	UI16	2	1…65535	
SID_PDB9	SID of the TM_PDB8 packet	49	UI8	1	196 (see section 8)	
enaflag_PDB9	reporting enable state of the TM_PDB8 packet	50	UI8	1	0 (disabled), 1 (enabled)	
period_PDB9	reporting period of the TM_PDB8 packet	51	UI16	2	1…65535	
SID_PDB10	SID of the TM_PDB10 packet	53	UI8	1	197 (see section 8)	
enaflag_PDB10	reporting enable state of the TM_PDB10 packet	54	UI8	1	0 (disabled), 1 (enabled)	
period_PDB10	reporting period of the TM_PDB10 packet	55	UI16	2	1…65535	
SID_PDB11	SID of the TM_PDB11 packet	57	UI8	1	198 (see section 8)	
enaflag_PDB11	reporting enable state of the TM_PDB11 packet	58	UI8	1	0 (disabled), 1 (enabled)	
period_PDB11	reporting period of the TM_PDB11 packet	59	UI16	2	1…65535	
SID_PDB12	SID of the TM_PDB12 packet	61	UI8	1	199 (see section 8)	
enaflag_PDB12	reporting enable state of the TM_PDB12 packet	62	UI8	1	0 (disabled), 1 (enabled)	
period_PDB12	reporting period of the TM_PDB12 packet	63	UI16	2	1…65535	
SID_PDB13	SID of the TM_PDB13 packet	65	UI8	1	200 (see section 8)	
enaflag_PDB13	reporting enable state of the TM_PDB13 packet	66	UI8	1	0 (disabled), 1 (enabled)	
period_PDB13	reporting period of the TM_PDB13 packet	67	UI16	2	1…65535	
SID_PDB14	SID of the TM_PDB14 packet	69	UI8	1	201 (see section 8)	
enaflag_PDB14	reporting enable state of the TM_PDB14 packet	70	UI8	1	0 (disabled), 1 (enabled)	
period_PDB14	reporting period of the TM_PDB14 packet	71	UI16	2	1…65535	
SID_PDB15	SID of the TM_PDB15 packet	73	UI8	1	202 (see section 8)	
enaflag_PDB15	reporting enable state of the TM_PDB15 packet	74	UI8	1	0 (disabled), 1 (enabled)	
period_PDB15	reporting period of the TM_PDB15 packet	75	UI16	2	1…65535	
SID_PDB16	SID of the TM_PDB16 packet	77	UI8	1	203 (see section 8)	
enaflag_PDB16	reporting enable state of the TM_PDB16 packet	78	UI8	1	0 (disabled), 1 (enabled)	
period_PDB16	reporting period of the TM_PDB16 packet	79	UI16	2	1…65535	

Note:

1. This TM packet is the response on a TC_DUMP_SIDSTATES command.

7.3 SERVICE 5

7.3.1 TM_EVENT – EVENT REPORT

Description: This kind of TM packet is used to report the occurrence of an event.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1, 2, 3, or 4 (see note 1)
Length of source data: 2+k (k depends on the reported event)
Structure of the source data field (offsets are wrt. The start of the source data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)
EID	Event identifier	0	UI16	2
…	complementary information (depends on the EID value, see section 9)	2	any	k

Note:

1. The service subtype of the packet indicates the severity of the event:

   1 normal/progress (used e.g. to indicate the progress of a long-term command execution; see section 9)
   2 low severity error/anomaly (used for warnings; see section 9)
   3 medium severity error/anomaly (used e.g. if the STR autonomously switched back from NAT to AAD mode because of a loss of tracking; see section 9)
   4 high severity error/anomaly (currently not used by the STR)

7.3.2 TM_DISEVENTS_DUMP – DISABLED EVENTS REPORT

Description: This reports list the EIDs of all events the reporting of which is currently disabled.
PRID: see section 5.1
PCAT: 3 (table)
Service type: 5

Service Subtype: 134
Length of source data: 2+n*2 byte (see note 2)
Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
n	the number of events for which the reporting is currently disabled	0	UI16	2	≥0 (see note 1)
eids[*n*]	list of event identifiers for which the reporting is currently disabled	2	UI16	*n**2	any of the EIDs supported by the STR (see section 9)

Note:

1. This kind of TM packet is the response on a TC_DUMP_DISEVENTS command.

2. If no events are currently disabled the *n* parameter in the source data field will be 0 and no further parameters will be reported.

7.4 SERVICE 6

7.4.1 TM_MEM_DUMP – MEMORY DUMP REPORT

Description: The TM_MEM_DUMP packet delivers the dump of a memory area.
PRID: see section 5.1
PCAT: 9 (dump)
Service type: 6
Service Subtype: 6
Length of source data: 10+n bytes, n≥1 (see note 2)
Structure of the source data field (offsets are wrt. The start of the source data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
MemID	memory id	0	UI16	2	see section 13	
addr	starting address of this memory block	2	UI32	4	see note 3	
n	amount of memory dumped with this packet	6	UI32	4	see note 3	SAU
memData[n]	content of the memory block specified by *addr* and *n*	10		*n**SAU		

Note:

1. This TM packet type is generated in response to a TC_DUMP_MEMORY command. The *addr* and *n* parameters specify the address range *addr*..*addr*+*n**SAU-1 of memory dumped with this packet, where SAU is the size of the smallest addressable unit defined for the specified MemID (see section 13).

2. The number nP of dump packets generated from a single TC_DUMP_MEMORY command is calculated by nP=(*nToDump*+*nPerPkt-1*)/*nPerPkt*, where the *nToDump* parameter value is directly taken from the command packet and the nPerPkt value is derived from the parameter *maxDumpSizeInBytes* (set by TC_MAX_DUMPSIZE) by the formula nPerPkt=(*maxDumpSizeInBytes*–10)/SAU (note that because *maxDumpSizeInBytes* is required to be ≥SAU+10, the nPerPkt value will be ≥1).

3. The parameters *addr* and *n* for the generated dump packets are selected as follows:

$$addr_i = saddr + (i-1)\cdot nPerPkt * SAU, \quad i = 1,...,nP$$, where *saddr* is taken from the command packet

$$n_i = \begin{cases} nPerPkt & i = 1,...,nP-1 \\ nToDump - (nP-1)\cdot nPerPkt & i = nP \end{cases}$$

4. The dump packets will be generated and delivered with a 1Hz frequency (1 dump packet per second). See also information in section 2.3.1 on telemetry update rates.

7.4.2 TM_MEM_CHECK – MEMORY CHECK REPORT

Description: The TM_MEM_CHECK packet reports the checksum of a memory area.
PRID: see section 5.1
PCAT: 3 (table)
Service type: 6
Service Subtype: 10
Length of source data: 12 bytes
Structure of the source data field (offsets are wrt. The start of the source data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
MemID	memory id	0	UI16	2	see section 13	
addr	starting address of the memory block for which the checksum has been calculated	2	UI32	4		
n	amount of memory block for which the checksum has been calculated	6	UI32	4	see note 2	SAU

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
chksum	the CRC 16 bit checksum value calculated over the specified address range	10	UI16	2		

Note:

1. This TM packet is the response on a TC_CHECK_MEMORY command.
2. The length value *n* always specifies SAUs and thus the amount of memory covered by the *chksum* value depends on the SAU size which is associated to the given *MemID* (see section 13).
3. The address range covered by the *chksum* value is given by *addr*..*addr*+*n**SAU-1.

7.5 SERVICE 17

7.5.1 TM_PING – CONNECTION TEST REPORT

Description: This is the response on a TC(17,1) command.
PRID: see section 5.1
PCAT: 1 (acknowledge)
Service type: 17
Service Subtype: 2
Length of source data: 0

7.6 SERVICE 221

7.6.1 TM_CALIBRATION_DUMP – CALIBRATION DATA DUMP

Description: The TM_CALIBRATION_DUMP packet delivers the STR's current calibration data (whole or a specified part of it).
PRID: see section 5.1
PCAT: 3 (table)
Service type: 221
Service Subtype: 30

Length of source data: 4+n bytes (1≤n≤1148)
Structure of the source data field:

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
start	the starting byte offset (0-based) within the calibration data structure	0	UI16	2	0..1147
n	the amount of calibration data dumped with this packet	2	UI16	2	1..1148
calData[n]	the current contents of the specified part of the calibration data structure	4		n	

Note:

1. This TM packet is the response on a TC_DUMP_CALIBRATION command.
2. The parameters *start* and *n* are a copy from the command packet.
3. The calibration data structure is described in section 12.2.

7.6.2 TM_PREPROCESSOR_DUMP – PREPROCESSOR CONFIGURATION DUMP

Description: The TM_PREPROCESSOR_DUMP packet delivers the current preprocessor configuration of the STR.
PRID: see section 5.1
PCAT: 3 (table)
Service type: 221
Service Subtype: 31
Length of source data: 16 bytes
Structure of the source data field:

Argument Name	Description	Length (byte)
PreProcCfg	the current contents of the preprocessor configuration table	16

Note:

1. This TM packet is the response on a TC_DUMP_PREPROCESSOR command.
2. The preprocessor configuration data structure is described in section 12.1.

7.6.3 TM_ALGORITHM_DUMP – ALGORITHM PARAMETER TABLE DUMP

Description: The TM_ALGORITHM_DUMP packet delivers the current algorithm control parameters of the STR.
PRID: see section 5.1
PCAT: 3 (table)
Service type: 221
Service Subtype: 32
Length of source data: 116 bytes

Structure of the source data field:

Argument Name	Description	Length (byte)
AlgPars	the current contents of the algorithm parameters table	116

Note:

1. This TM packet is the response on a TC_DUMP_ALGORITHM command.
2. The algorithm parameters structure is described in section 12.3.

7.6.4 TM_GSC_DUMP – GUIDE STAR CATALOGUE DUMP

Description: The TM_GSC_DUMP packet delivers a part of the Guide Star Catalogue.
PRID: see section 5.1
PCAT: 3 (table)
Service type: 221
Service Subtype: 33
Length of source data: $4+nE*16$ bytes, $1 \leq nE \leq 4000$ (see note 3)

Structure of the source data field (offsets are wrt. The start of the source data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
sE	index (0-based) of the first GSC entry dumped with this packet	0	UI16	2	0..3999 (see note 3)
nE	number of GSC entries dumped with this packet	2	UI16	2	1..4000 (see note 3)
gscEntries[*nE*]	the contents of *nE* GSC entries beginning at entry *sE*	4	see note 3	*nE**16	see note 4

Notes:

1. This TM packet type is generated in response to a TC_DUMP_GSC command. The *sE* and *nE* parameters specify the range *sE*..*sE*+*nE*-1 of GSC entries dumped with this packet. The GSC entries are dumped in ascending table order, starting at entry *sE*.

2. The number nP of dump packets generated from a single TC_DUMP_GSC command is calculated by nP=(*nToDump*+nPerPkt-*1*)/nPerPkt, where the *nToDump* parameter value is directly taken from the command packet and the nPerPkt value is derived from the command parameter *maxDumpSizeInBytes* by the formula nPerPkt=(*maxDumpSizeInBytes*–4)/16 (note that because *maxDumpSizeInBytes* is required to be ≥16+4, the nPerPkt value will be ≥1). The TM_GSC_DUMP packets resulting from a TC_DUMP_GSC command are generated with 10Hz update rate. See also information in section 2.3.1 on telemetry update rates.

3. The parameters *sE* and *nE* for the generated dump packets are selected as follows:

$$sE_i = start + (i-1)\cdot nPerPkt, \quad \mathrm{i} = 1,...,nP$$, where *start* is taken from the command packet

$$nE_i = \begin{cases} nPerPkt & \mathrm{i} = 1,...,nP-1 \\ nToDump-(nP-1)\cdot nPerPkt & \mathrm{i} = nP \end{cases}$$

4. A single GSC entry occupies 16 bytes and has the following structure:

Parameter Name	Description	Format	Length (byte)	Range	Unit
cnr	catalogue number of the guide star	UI16	2	unique within catalogue	n.a.
mag	total charge of the guide star in detector LSB	UI16	2		[LSB]
v_x	unit vector of the guide star	F	4	−1.0 … +1.0	n.a.
v_y		F	4	−1.0 … +1.0	n.a.
v_z		F	4	−1.0 … +1.0	n.a.

The vector given by the components v_x, v_y, and v_z must be a unit vector, i.e. must have norm 1.

5. The Guide Star Catalog is described in section 12.4.

7.6.5 TM_GSC_INDICES_DUMP – GSC TILE INDEX LIST DUMP

Description: The TM_GSC_INDICES_DUMP packet delivers a part of the GSC tile index list.
PRID: see section 5.1
PCAT: 3 (table)
Service type: 221
Service Subtype: 34
Length of source data: 2+*nE**2 bytes, 1≤*nE*≤216 (see note 2)

Structure of the source data field (offsets are wrt. The start of the source data field):

Argument Name	Description	Offset (byte)	Format	Length (byte)	Range
sE	index (0-based) of the first entry of the GSC index list dumped with this packet	0	UI8	1	0..215 (see note 3)
nE	number of entries of the GSC index list dumped with this packet	1	UI8	1	1..216 (see note 3)
gscIndices[*nE*]	the contents of *nE* entries of the GSC index list beginning at entry *sE*	2	UI16	*nE**2	

Note:

1. This TM packet type is generated in response to a TC_DUMP_GSC_INDICES command. The *sE* and *nE* parameters specify the range *sE*..*sE*+*nE*-1 of GSC Index list entries dumped with this packet. The GSC Index list entries are dumped in ascending list order, starting at entry *sE*.

2. The number nP of dump packets generated from a single TC_DUMP_GSC_INDICES command is calculated by nP=(*nToDump*+*nPerPkt-1*)/*nPerPkt*, where the *nToDump* parameter value is directly taken from the command packet and the nPerPkt value is derived from the command parameter *maxDumpSizeInBytes* by the formula nPerPkt=(*maxDumpSizeInBytes*–2)/2 (note that because *maxDumpSizeInBytes* is required to be ≥2+2, the nPerPkt value will be ≥1). The TM_GSC_INDICES_DUMP packets resulting from a TC_DUMP_GSC_INDICES command are generated with 10Hz update rate. See also information in section 2.3.1 on telemetry update rates.

3. The parameters *sE* and *nE* for the generated dump packets are selected as follows:

$$sE_i = start + (i-1)\cdot nPerPkt, \quad \mathrm{i} = 1,\ldots,nP$$, where *start* is taken from the command packet

$$nE_i = \begin{cases} nPerPkt & \mathrm{i} = 1,\ldots,nP-1 \\ nToDump-(nP-1)\cdot nPerPkt & \mathrm{i} = nP \end{cases}$$

4. The GSC Index list is decribed in section 12.5.

8 STRUCTURE IDS USED BY THE STR

The following table lists all SIDs used by the STR together with the TM packets where the SID is used:

SID (dec)	Meaning	Used in
1	Status and Health Data Block	TM_SDB
105	Attitude Data Block	TM_ADB
106	Tracker Data Block	TM_TDB
128	Engineering Data Block	TM_EDB
188	Pixel Data Block No. 1	TM_PDB
189	Pixel Data Block No. 2	TM_PDB
…	…	…
203	Pixel Data Block No. 16	TM_PDB

9 STR EVENT REPORTS

9.1 STR EVENT SUMMARY

Table 9-1 gives an overview on all event reports which may be generated by the STR together with the STR modes in which the events may be generated.

Note: contrary to the range allocation specified in section B2.10 of AD1, the EID ranges used for STR events are allocated as follows:

- Nominal (informational) events: 0x420 – 0x47F
- Low severity events: 0x480 – 0x4FF
- Medium severity events: 0x500 – 0x57F
- High severity events: 0x580 – 0x5FF

See AD4 for the formal request for deviation.

The column in Table 9-1 labeled “Time of generation” can be used to determine whether the corresponding event will occur before or after the periodic telemetry (TM_ADB, TM_SDB, TM_TDB, TM_EDB, TM_PDB) within a data block (on Data Block Transfer level as in section 4.3). The entries marked with N/A denote those events which are only generated in Boot mode (in which no periodic telemetry is generated).

Event name (mnemonic)	(Service, Subservice)	EID (hex)	TM packet Length [bytes]	BOOT	STANDBY	AAD	NAT	PHOTO	Time of generation[1]
EV_STR_ALIVE	(5,1)	420	86	X	X[2]	X[2]	X[2]	X[2]	l
EV_SELFTEST_INPROGRESS	(5,1)	421	24	X	X	-	-	-	l
EV_SELFTEST_OK	(5,1)	422	23	X	X	-	-	-	l
EV_EEWRITE_INPROGRESS	(5,1)	423	22	X	X	-	-	-	l
EV_EEREAD_INPROGRESS	(5,1)	424	22	X	X	-	-	-	l
EV_MEMCHK_INPROGRESS	(5,1)	425	22	X	X	X	X	X	l
EV_PPS_RESYNC	(5,1)	426	23	X	X	X	X	X	e
EV_TIME_SYNC_SUCCESS	(5,1)	427	50	X	X	X	X	X	e
EV_OPMODE_BOOT	(5,1)	430	22	X	-	-	-	-	N/A
EV_OPMODE_STANDBY	(5,1)	431	22	-	X	-	-	-	l
EV_OPMODE_PHOTO	(5,1)	432	22	-	-	-	-	X	l
EV_OPMODE_AADF	(5,1)	433	22	-	-	X	-	-	l
EV_OPMODE_AADW	(5,1)	434	22	-	-	X	-		l
EV_OPMODE_NAT	(5,1)	435	22	-	-	-	X	-	l
EV_STREAK_MODE	(5,2)	480	35	-	-	X	-	-	l
EV_CYCLE_TOO_LONG	(5,2)	481	22	-	X[3]	X	X	X[3]	e
EV_TRAP_MINOR[4]	(5,2)	482	28	X	X	X	X	X	e
EV_SELFTEST_FAILED	(5,3)	500	24	X	X	-	-	-	l
EV_LONG_ACQUISITION	(5,3)	501	27	-	-	X	-	-	l
EV_IDENTIFICATION_FAILED	(5,3)	502	27	-	-	X	-	-	l
EV_DBT_TIMEOUT	(5,3)	503	24	X	X	X	X	X	e
EV_WATCHDOG_RESET	(5,3)	504	23	X	X	-	-	-	e
EV_TRAP_MAJOR	(5,3)	505	28	X	X	-	-	-	e

[1] With respect to generation of periodic telemetry within STR cycle; e = earlier than periodic; l = later than periodic
[2] Disabled by default
[3] Only possible with non-nominal (very small) allowedProcessingTimeInMs threshold
[4] Currently not implemented

Event name (mnemonic)	(Service, Subservice)	EID (hex)	TM packet Length [bytes]	BOOT	STANDBY	AAD	NAT	PHOTO	Time of generation[1]
EV_RAM_ERROR	(5,3)	506	30	X	X	X	X	X	e
EV_BAD_APSW	(5,3)	507	26	X	X	-	-	-	e
EV_BAD_TESTAPP	(5,3)	508	26	X	-	-	-	-	N/A
EV_TIME_SHIFT	(5,3)	509	24	X	X	X	X	X	e
EV_SYNC_LOST	(5,3)	50A	23	X	X	X	X	X	e
EV_PPS_SPURIOUS	(5,3)	50B	23	X	X	X	X	X	e
EV_TRACK_LOSS	(5,3)	50C	25	-	-	-	X	-	l
EV_TM_OVERFLOW	(5,3)	50D	24	X	X	X	X	X	e
EV_CYCLE_OVERRUN	(5,3)	50E	22	-	X[5]	X	X	X[5]	e
EV_TIME_SYNC_FAILED	(5,3)	50F	50	X	X	X	X	X	e
EV_INVALID_GSC	(5,3)	510	30	-	X	-	-	-	l

Table 9-1 STR Event Summary

The following sections describe each event in detail.

[5] Only possible with non-nominal (very large) minimumSpareTimeInMs threshold

9.2 EV_STR_ALIVE – STAR TRACKER ALIVE

Description: This info event is sent to indicate that the STR is alive.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 66
Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x420
health	The health data that is also available at SA01. For the detailed structure see 4.2	2	B512	64	see 4.2

Note:

1. This event is generated at a 0.1Hz rate (1 event packet every 10s). See also information in section 2.3.1 on telemetry update rates.
2. This event is only enabled in the Bootmode and must be explicitly enabled in all other modes.

9.3 EV_SELFTEST_INPROGRESS – SELFTEST IN PROGRESS

Description: This info event is sent to indicate that an STR selftest is in progress.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 4
Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x421
reqTests	a bit field indicating which tests were requested	2	B8	1	see note 1
curTest	indicates which of the requested tests is currently in progress	3	B8	1	see note 1

Notes:

1. The bits of *reqTests* and *curTest* correspond to the tests as follows (b_0 is msb, b_7 is lsb):

   b_0: check the Boot Mode Handler S/W in ROM

   b_1: check the Application S/W in EEPROM

   b_2: check the APSW Calibration Data block in EEPROM

   b_3: check the Test Application S/W in the dedicated RAM area

   b_4: check the Testapp Calibration Data block in the dedicated RAM area

   b_5…b_7: spare (must be 0)

   Example values for *reqTests*:

   0xF8 selftest of Boot Mode Handler, Application SW, APSW Calibration Data, Test Application S/W, Testapp Calibration Data

   0xE0 selftest of Boot Mode Handler, Application SW, APSW Calibration Data

   0x18 selftest of Test Application S/W, Testapp Calibration Data

   Only those tests for which the corresponding bit in the *reqTests* field is 1 are being performed.

   The currently active test is indicated by a 1 in *curTest* (exactly one bit in *curTest* will be set).

2. This progress event is generated at a 0.1Hz rate (1 event packet every 10s) during execution of a TC_SELFTEST command. See also information in section 2.3.1 on telemetry update rates.

9.4 EV_SELFTEST_OK – SELFTEST SUCCESSFUL

Description: This info event is sent to indicate that the STR selftest has been completed successfully.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 3

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x422
reqTests	a bitfield indicating which tests were requested	2	B8	1	see note 1

Note:

1. The bits of *reqTests* correspond to the tests as follows (b_0 is msb, b_7 is lsb):

   b_0: check the Boot Mode Handler S/W in ROM

   b_1: check the Application S/W in EEPROM

   b_2: check the APSW Calibration Data block in EEPROM

   b_3: check the Test Application S/W in the dedicated RAM area

   b_4: check the Testapp Calibration Data block in the dedicated RAM area

   b_5…b_7: spare (must be 0)

Only those tests for which the corresponding bit in the *reqTests* field is 1 have been performed.

9.5 EV_SELFTEST_FAILED – SELFTEST FAILED

Description: This error event is sent to indicate that the STR selftest has failed.
PRID: see section 5.1

PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 4

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x500
reqTests	a bitfield indicating which tests were requested	2	B8	1	see note 1
results	a bitfield indicating the test results	3	B8	1	see note 2

Notes:

1. The bits of *reqTests* correspond to the tests as follows (b_0 is msb, b_7 is lsb):

   b_0: check the Boot Mode Handler S/W in ROM

   b_1: check the Application S/W in EEPROM

   b_2: check the APSW Calibration Data block in EEPROM

   b_3: check the Test Application S/W in the dedicated RAM area

   b_4: check the Testapp Calibration Data block in the dedicated RAM area

   $b_5 \ldots b_7$: spare (must be 0)

   Only those tests for which the corresponding bit in the *reqTests* field is 1 have been performed.

2. The bits in the *results* field have a 1-to-1 mapping to the bits in the *reqTests* field. If a bit in *reqTests* is 0 the corresponding test type was not requested and thus the corresponding bit in *results* is don't care (will be set to 0). For each bit which is 1 in *reqTests* the corresponding bit in *results* shows the outcome of the test as follows: 1➔test successful, 0➔test failed.

3. The complete selftest is treated as failed if at least one the requested tests failed (*reqTests* **xor** *results*≠0.

9.6 EV_EEWRITE_INPROGRESS – EEPROM WRITE IN PROGRESS

Description: This info event is sent to indicate that an EEPROM write operation is in progress.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 2

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x423

Note:

1. This type of event report is generated at a 0.1Hz rate (1 event packet every 10s) during the execution of a TC_COPY_TAPP2EE command. See also information in section 2.3.1 on telemetry update rates.

9.7 EV_EEREAD_INPROGRESS – EEPROM READ IN PROGRESS

Description: This info event is sent to indicate that an EEPROM read operation is in progress.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 2

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x424

Note:

1. This type of event report is generated at a 0.1Hz rate (1 event packet every 10s) during the execution of a TC_COPY_EE2TAPP command. See also information in section 2.3.1 on telemetry update rates.

9.8 EV_MEMCHK_INPROGRESS – MEMORY CHECKSUMMING IN PROGRESS

Description: This info event is sent to indicate that a memory checksumming operation is in progress.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 2

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x425

Note:

1. This type of event report is generated at a 0.1Hz rate (1 event packet every 10s) during the execution of a TC_CHECK_MEMORY command. See also information in section 2.3.1 on telemetry update rates.

9.9 EV_OPMODE_BOOT – STR ENTERED THE BOOT MODE

Description: This info event is sent to indicate that the STR has entered the BOOT mode.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 2

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x430

Notes:

1. This type of event report is generated after the BOOT mode has been entered.
2. Note that during power up the BOOT mode will be entered automatically if the application S/W could not be loaded from EEPROM because of a failed EEPROM check.
3. Another way to enter the BOOT mode is a TC_RESET command specifying BOOT as the startMode.

9.10 EV_OPMODE_STANDBY – STR ENTERED THE STANDBY MODE

Description: This info event is sent to indicate that the STR has entered the STANDBY mode.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 2

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x431

Notes:

1. This type of event report is generated after the STANDBY mode has been entered.
2. During power up the STANDBY mode will be entered automatically but only in a transient way because after some basic initializations the STR goes through to AAD mode.
3. Other cases to enter the STANDBY mode are on a TC_STANDBY command and automatically when a PHOTO operation requested by a TC_PHOTO command has been completed.

9.11 EV_OPMODE_PHOTO – STR ENTERED THE PHOTO MODE

Description: This info event is sent to indicate that the STR has entered the PHOTO mode.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 2

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x432

Notes:

1. This type of event report is generated after the PHOTO mode has been entered.
2. The PHOTO mode can only be reached with a TC_PHOTO command, i.e. there is no automatic transition to the PHOTO mode.

9.12 EV_OPMODE_AADF – STR ENTERED THE AADF MODE

Description: This info event is sent to indicate that the STR has entered the full-frame submode (AADf) of the Autonomous Attitude Determination mode (AAD).
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 2

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x433

Notes:

1. This type of event report is generated after the AADf mode has been entered.
2. During power up the AADf mode is automatically entered from STANDBY mode.
3. Other cases to enter AADf mode are on a TC_ACQUIRE command (necessary when the STR switched back to STANDBY at the end of a one-shot PHOTO operation) or automatically on a loss of tracking in AADw or NAT mode but note that in the latter case the EV_OPMODE_AADF event will be preceded by an EV_TRACK_LOSS event.

9.13 EV_OPMODE_AADW – STR ENTERED THE AADW MODE

Description: This info event is sent to indicate that the STR has entered the windowing submode (AADw) of the Autonomous Attitude Determination mode (AAD).
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 2

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x434

Notes:

1. This type of event report is generated after the AADw mode has been entered.
2. The AADw mode is automatically entered from AADf mode if enough tracked and verified stars are available.
3. Another possibility to enter AADw mode is on a TC_ACQUIRE command providing a complete set of a-priori information.

9.14 EV_OPMODE_NAT – STR ENTERED THE NAT MODE

Description: This info event is sent to indicate that the STR has entered the Nominal Attitude Tracking mode (NAT).
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 2
Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x435

Notes:

1. This type of event report is generated after the NAT mode has been entered.
2. The NAT mode is automatically entered from AADw mode if a verified attitude is available. There is no other way to reach the NAT mode.

9.15 EV_STREAK_MODE – STR ENTERED STREAK MODE

Description: This warning event is sent to indicate that the STR has entered streak processing mode.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 2
Length of source data: 15
Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x480
tInt	used integration time [ms]	2	UI8	1	
offset	used binarisation offset [LSB]	3	UI16	2	

Name	Description	Offset (byte)	Format	Length (byte)	Range
numObjectsDetected	number of detected objects	5	UI16	2	
numObjectsAcquired	number of acquired objects	7	UI16	2	
numObjectsL1	number of objects in object list 1	9	UI16	2	
numObjectsL2	number of objects in object list 2	11	UI16	2	
numStarsTracked	number of tracked objects	13	UI8	1	
numberOfStreaks	new number of streaks	14	UI8	1	

Note:

1. The streak processing mode is a submode of the AADf mode in which only a part of the available FOV – the actual streak – is processed. Streak proceesing mode will only be used if it has been enabled and only if the number of objects is very high, e.g. because of solar flares or other disturbances.

9.16 EV_LONG_ACQUISITION – NAT NOT REACHED IN TIME

Description: This error event is sent in AAD mode to indicate that the time tried to reach NAT mode exceeded a predefined limit.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 7

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x501
tInt	used integration time [ms]	2	UI8	1	
offset	used binarisation offset [LSB]	3	UI16	2	
tElapsed	time [ms] elapsed since entering AAD mode	5	UI16	2	

9.17 EV_IDENTIFICATION_FAILED – STAR IDENTIFICATION FAILED

Description: This error event is sent to indicate that the last star identification activity failed.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 7

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x502
nToId	number of stars for identification	2	UI8	1	
idRes	identification result	3	E	1	see IdResult enumeration
nTrk	number of tracked stars	4	UI8	1	
nVer	number of verified stars	5	UI8	1	
attRes	attitude result	6	E	1	see AttResult enumeration

9.18 EV_DBT_TIMEOUT – DATA BLOCK TRANSFER TIMEOUT

Description: This error event is sent to indicate that an acquisition transfer request (ATR from STR to OBC) was not confirmed in time by the OBC.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 4

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x503

Name	Description	Offset (byte)	Format	Length (byte)	Range
nbDisc	accumulated number of bytes discarded	2	UI16	2	

9.19 EV_WATCHDOG_RESET – WATCHDOG RESET OCCURRED

Description: This error event is sent to indicate that a watchdog reset occurred in the STR (caused a reboot).
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 3
Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x504
opMode	STR operational mode in which the watchdog reset occurred	2	E	1	see OpMode enumeration

Note:

1. This event packet is generated by the STR after it has been rebooted and the APSW (or the boot mode handler) has been started.

9.20 EV_TRAP_MINOR – MINOR CPU TRAP OCCURRED

Description: This warning event is sent to indicate that a minor CPU trap occurred in the STR (did not cause a reboot).
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 2
Length of source data: 8
Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x482
trapNum	the trap number	2	UI8	1	
trapAddr	memory address where the trap occurred	3	UI32	4	
opMode	STR operational mode in which the trap occurred	7	E	1	see OpMode enumeration

Note:

1. Minor CPU traps are those which do not cause a reboot of the STR, i.e. the STR APSW (or the boot mode handler) will be continued where it was interrupted by the trap, possibly after some corrective action done by the trap handler.

9.21 EV_TRAP_MAJOR – MAJOR CPU TRAP OCCURRED

Description: This error event is sent to indicate that a major CPU trap occurred in the STR (caused a reboot).
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 8

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x505
trapNum	the trap number	2	UI8	1	
trapAddr	memory address where the trap occurred	3	UI32	4	
opMode	STR operational mode in which the trap occurred	7	E	1	see OpMode enumeration

Note:

1. This event packet is generated by the STR after it has been rebooted and the APSW (or the boot mode handler) has been started.

9.22 EV_RAM_ERROR – SINGLE-BIT RAM ERROR

Description: This error event is sent to indicate that one or more single bit RAM errors occurred in the STR (detected and corrected by the EDAC H/W) since the last generation of this event packet.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 10
Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x506
totErrs	total (accumulated) number of single bit RAM errors detected (and corrected) since last boot	2	UI32	4	
lastErrAddr	memory address of latest single bit error	6	UI32	4	

Note:

1. Only single bit errors are reported with this event type (double bit errors cause a major CPU trap and a reboot of the STR).

9.23 EV_BAD_APSW – CHECK OF APSW IN EEPROM FAILED

Description: This error event is sent during bootup to indicate that the check of the APSW in EEPROM failed.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 6
Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x507
expCs	expected EEPROM checksum (CRC16)	2	UI16	2	
calcCs	calculated EEPROM checksum (CRC16)	4	UI16	2	

Note:

1. This event may be generated if the requested startmode was either **startApsw** or **startApswForced** (see StartMode enumeration).
2. In case the APSW in EEPROM is detected as bad the boot loader will nevertheless load and start the APSW if the startMode **startApswForced** was requested. Otherwise the boot loader will load and start the boot mode handler S/W.

9.24 EV_BAD_TESTAPP – CHECK OF TEST APPLICATION S/W FAILED

Description: This error event is sent during bootup if activation of the test application S/W currently contained in the reserved RAM area was requested and the check of this test S/W failed.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 6

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x508
expCs	expected TestApp checksum (CRC16)	2	UI16	2	
calcCs	calculated TestApp checksum (CRC16)	4	UI16	2	

Note:

1. This event may only be generated if the requested startmode was **startTestapp** (see StartMode enumeration).
2. In case the test application S/W is detected as bad the boot loader loads and starts the boot mode handler S/W from ROM.

9.25 EV_TIME_SHIFT – SIGNIFICANT DEVIATION BETWEEN COBT AND LOBT

Description: This error event is sent when at the time of PPS synchronization, the LOBT deviates more than a configurable threshold (default: 4ms) from the received COBT
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 4

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x509
deviation	COBT–LOBT [in ms]	2	SI8	1	-128…-4 or +4…+127 (see foot note 2)
syncSrc	current sync source	3	E	1	see SyncSource enumeration

Note:

1. This event indicates that at the time of synchronization with the COBT, the LOBT had been adjusted by more than 4ms
2. The threshold of 4ms can be modified via TC_SELECT_SYNC_SOURCE

9.26 EV_SYNC_LOST – NO PPS WITHIN ACCEPTABLE TIME

Description: This error event is sent to indicate that the STR did not receive a PPS pulse within 1010 milliseconds since the last received pulse.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 3

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x50A
syncSrc	current sync source	2	E	1	see SyncSource enumeration

Note:

1. This event indicates that a timeout occurred in waiting for the next PPS. The STR will use the next pulse it receives from the OBC for a resynchronization.

9.27 EV_PPS_SPURIOUS – SPURIOUS PPS RECEIVED BY THE STR

Description: This error event is sent to indicate that the STR received a PPS pulse within 990ms after the last accepted (regular) PPS pulse.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 3

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x50B
syncSrc	current sync source	2	E	1	see SyncSource enumeration

Note:

1. Any PPS which is received more than 10ms before the next expected PPS will be ignored by the STR, i.e. will not be used for an LOBT correction.

9.28 EV_PPS_RESYNC – SUCCESSFUL RESYNCHRONIZATION AFTER SYNC WAS LOST

Description: This info event is sent to indicate that after synchronization was lost the STR is now synchronized again.

PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 3

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x426
syncSrc	current sync source	2	E	1	see SyncSource enumeration

9.29 EV_TRACK_LOSS – LOSS OF TRACKING

Description: This error event is sent in NAT mode to indicate that a loss of tracking occurred.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 5

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x50C
nTrk	number of tracked stars	2	UI8	1	
nVer	number of verified stars	3	UI8	1	
attRes	attitude result	4	E	1	see AttResult enumeration

9.30 EV_TM_OVERFLOW – TM QUEUE OVERFLOW

Description: This error event is sent to indicate an overflow of the TM output queue in the STR.

PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 4

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x50D
nbDisc	accumulated number of bytes discarded	2	UI16	2	

Note:

1. This type of event report is generated if during the last cycle at least 1 TM packet could not be sent because of not enough space in the TM output queue. Any TM packets which overflow the output queue are discarded.

2. This event packet will be sent in the next cycle after a TM overflow occurred.

9.31 EV_CYCLE_TOO_LONG – STR PROCESSING CYCLE EXCEEDED A SOFT LIMIT

Description: This warning event is sent to indicate that an STR processing cycle exceeded some soft limit.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 2
Length of source data: 2

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x481

9.32 EV_CYCLE_OVERRUN – STR PROCESSING CYCLE EXCEEDED A HARD LIMIT

Description: This error event is sent to indicate that an STR processing cycle exceeded some hard limit.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 2

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x50E

9.33 EV_TIME_SYNC_SUCCESS – THE VERIFY_TIME_SYNC COMMAND WAS SUCCESSFUL

Description: This info event is sent to indicate that the time synchronization requested by the TC_VERIFY_TIME_SYNC command was successful.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 1
Length of source data: 30

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x427
lobt_at tc_reception	the LOBT at TC reception	2	TS	7	
obt_for_next_sync_ref	received OBT for next synchronization reference	9	TS	7	

Name	Description	Offset (byte)	Format	Length (byte)	Range
new_lobt_at_sync_ref	new LOBT set at synchronization reference	16	TS	7	
orig_lobt_at_sync_ref	original LOBT at synchronization reference	23	TS	7	

9.34 EV_TIME_SYNC_FAILED – THE VERIFY_TIME_SYNC COMMAND FAILED

Description: This error event is sent to indicate that the time synchronization requested by the TC_VERIFY_TIME_SYNC command was not successful.
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 30

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x50F
lobt_at tc_reception	the LOBT at TC reception	2	TS	7	
obt_for_next_sync_ref	received OBT for next synchronization reference	9	TS	7	
new_lobt_at_sync_ref	new LOBT set at synchronization reference	16	TS	7	
orig_lobt_at_sync_ref	original LOBT at synchronization reference	23	TS	7	

9.35 EV_INVALID_GSC – THE BUILT-IN GUIDE STAR CATALOG IS INVALID

Description: This event is sent when the initial validity check of the guide star catalog reveals an error
PRID: see section 5.1
PCAT: 7 (event)
Service type: 5
Service Subtype: 3
Length of source data: 10

Structure of the source data field (offsets are wrt. The start of the source data field):

Name	Description	Offset (byte)	Format	Length (byte)	Range
eid	the event identifier of this event report	0	UI16	2	0x510
firstInvalidTileIndex	the index of the first invalid tile specification	2	UI32	4	
firstInvalidStarIndex	the index of the first invalid star specification	6	UI32	4	

10 TC ACCEPTANCE AND EXECUTION ERROR REPORTS

This chapter lists all FIDs (fault identifiers) used by the STR to report TC acceptance errors (by TM(1,2) packets) and to report TC execution errors (by TM(1,8) packets).

The following table provides an overview of all used FIDs. For all predefined (i.e. common) FIDs taken from AD1 a reference to AD1, §B2.9 is given in the table.

FID (dec)	FID (hex)	Description of detected error	Short Name	Param 1 (32-bit)	Param 2 (32 bit)	Report Type	Generating service	Reference
256	100	illegal CCSDS packet version number	FID_ILLEGAL_VERSION			TM(1,2)	all	AD1, §B2.9
257	101	illegal packet type	FID_ILLEGAL_P_TYPE			TM(1,2)	all	AD1, §B2.9
258	102	illegal data field header flag	FID_ILLEGAL_DFHF			TM(1,2)	all	AD1, §B2.9
259	103	PRID is not the STR's process identifier	FID_UNKNOWN_PRID			TM(1,2)	all	AD1, §B2.9
260	104	illegal packet category	FID_ILLEGAL_PCAT			TM(1,2)	all	AD1, §B2.9
261	105	illegal sequence flag	FID_ILLEGAL_SF			TM(1,2)	all	AD1, §B2.9
263	107	packet length out of allowed range	FID_INVALID_PLENGTH	received packet length		TM(1,2)	all	AD1, §B2.9
264	108	packet length not in line with given type and subtype	FID_LENGTH_DISCREP	received packet length	expected packet length	TM(1,8)	all	AD1, §B2.9
266	10A	illegal CCSDS secondary header flag	FID_ILLEGAL_SHF	data field header		TM(1,2)	all	AD1, §B2.9
267	10B	illegal TC packet PUS version number	FID_ILLEGAL_TC_PUS	data field header		TM(1,2)	all	AD1, §B2.9
268	10C	unknown service type	FID_UNKNOWN_S_TYPE	data field header	actual STR mode	TM(1,2)	all	AD1, §B2.9
269	10D	unknown service subtype	FID_UNKNOWN_S_SUBTYPE	data field header	actual STR mode	TM(1,2)	all	AD1, §B2.9
271	10F	calculated checksum not equal to received checksum	FID_CS_DISCREP	received checksum	calculated checksum	TM(1,2)	all	AD1, §B2.9
273	111	service requests a TM output with a logical structure larger than actually set MTU	FID_MTU_TOO_SMALL	size of MTU	size of requested logical structure	TM(1,8)	(6,5)	AD1, §B2.9
384	180	command not allowed in current mode	FID_CMD_NOT_ALLOWED	data field header	actual STR mode	TM(1,8)	all	
385	181	long duration command cannot be stored because the associated queue is full	FID_LONGDUR_QUEUE_FULL	data field header		TM(1,8)	(6,5); (6,9); (8,1); (221,23); (221,24); (223,10); (223,11)	

FID (dec)	FID (hex)	Description of detected error	Short Name	Param 1 (32-bit)	Param 2 (32 bit)	Report Type	Generating service	Reference
386	182	command temporarily inhibited due to blocking long duration command	FID_CMD_TEMPORARILY_INHIBITED	service of the blocking long duration command	subservice of the blocking long duration command	TM(1,8)	(220,1); (220,2); (220,3); (220,4)	
769	301	invalid collection interval	FID_INVALID_COLL_INT	received collection interval		TM(1,8)	(3,130); (3,131)	AD1, §B2.9
777	309	attempt to modify enabled (active) SID	FID_HK_ACTIVE	received SID		TM(1,8)	(3,130); (3,131)	AD1, §B2.9
778	30A	unknown structure identifier	FID_UNKNOWN_SID	received SID		TM(1,8)	(3,5); (3,6); (3,7); (3,8); (3,130); (3,131); (3,136)	AD1, §B2.9
1280	500	invalid number of EID's	FID_INVALID_NEID	received nEID parameter		TM(1,8)	(5,5); (5,6)	AD1, §B2.9
1281	501	amount of EID's not in line with nEID parameter	FID_NEID_LEN_DISCREP	received nEID parameter	received number of EID's	TM(1,8)	(5,5); (5,6)	AD1, §B2.9
1282	502	specified EID does not exist	FID_UNKNOWN_EID	index (1 to nEID) of unknown EID	received EID value	TM(1,8)	(5,5); (5,6)	AD1, §B2.9
1536	600	invalid (unknown) memory ID	FID_INVALID_MEM_ID	received memory ID		TM(1,8)	(6,2); (6,5); (6,9)	AD1, §B2.9
1537	601	invalid memory address	FID_INVALID_ADDRESS	received address value		TM(1,8)	(6,2); (6,5); (6,9)	AD1, §B2.9
1538	602	length out of allowed range	FID_INVALID_LENGTH	received length value		TM(1,8)	(6,2); (6,5); (6,9)	AD1, §B2.9
1539	603	amount of data not in line with length parameter	FID_DATA_LEN_DISCREP	received length value	received amout of data	TM(1,8)	(6,2); (221,12); (221,13)	AD1, §B2.9
1540	604	memory access failed	FID_FAILED_MEM_ACCESS	cause of failure (see MemoryAccessErrorId enumeration in §11.14)		TM(1,8)	(6,2); (6,5); (6,9); (223,10); (223,11)	AD1, §B2.9
1543	607	invalid *maxDumpSizeInBytes* parameter in TC_MAX_DUMPSIZE command (*maxDumpSizeInBytes*<SAU+10)	FID_INVALID_MXDMPSZ	received *maxDumpSizeInBytes* value		TM(1,8)	(224,10)	
2048	800	unknown function identifier	FID_UNKNOWN_FUNC_ID	received function ID		TM(1,8)	(8,1)	AD1, §B2.9
2060	80C	long duration TC execution aborted by TC(8,220) cmd request	FID_LONGDUR_TC_ABORTED	service of the aborted TC	subservice of the aborted TC	TM(1,8)	(8,220)	

FID (dec)	FID (hex)	Description of detected error	Short Name	Param 1 (32-bit)	Param 2 (32 bit)	Report Type	Generating service	Reference
2304	900	time sync verification failed	FID_TIME_SYNC_FAIL			TM(1,8)	(9,135)	AD1, §B2.9
2310	906	unknown sync source	FID_UNKNOWN_SYNC_SRC	received *syncSource* value		TM(1,8)	(9,136)	
2311	907	invalid shiftThresh threshold in TC_TIME_SHIFT_THRESHOLD	FID_INVALID_SHIFT_THRESHOLD	received *shiftThresh* value		TM(1,8)	(224,9)	
40960	A000	illegal a-priori quaternion in TC_ACQUIRE command (quaternion length is not 1)	FID_ILL_QUAT_IN_TCACQ			TM(1,8)	(220,2)	
40961	A001	illegal a-priori rate in TC_ACQUIRE command (rate magnitude exceeds a reasonable limit)	FID_ILL_RATE_IN_TCACQ			TM(1,8)	(220,2)	
40962	A002	TC_ACQUIRE failed due to invalid built-in guide star catalog	FID_INVALID_GSC	first invalid tile index	first invalid star index	TM(1,8)	(220,2)	
40976	A010	illegal *xMin* parameter in TC_PHOTO command (*xMin*+*xWidth*>1024)	FID_ILL_XRANGE_IN_TCPHOTO	received *xMin* value	received *xWidth* value	TM(1,8)	(220,3)	
40977	A011	illegal *yMin* parameter in TC_PHOTO command (*yMin*+*yWidth*>1024)	FID_ILL_YRANGE_IN_TCPHOTO	received *yMin* value	received *yHeight* value	TM(1,8)	(220,3)	
40978	A012	illegal *xWidth* parameter in TC_PHOTO command (*xWidth*>1020)	FID_ILL_XWIDTH_IN_TCPHOTO	received *xWidth* value		TM(1,8)	(220,3)	
40979	A013	illegal *yHeight* parameter in TC_PHOTO command (*yHeight*>1020)	FID_ILL_YHEIGHT_IN_TCPHOTO	received *yHeight* value		TM(1,8)	(220,3)	
40992	A020	illegal *startMode* parameter in TC_RESET command	FID_ILL_SMODE_IN_TCRESET	received *startMode* value		TM(1,8)	(220,4)	
41056	A060	illegal *n* value in TC_LOAD_GSC command (*n* not in range 1..4000)	FID_ILL_N_IN_TCLDGSC	received *n* value		TM(1,8)	(221,12)	
41057	A061	illegal GSC entry data in TC_LOAD_GSC command (*cnr* parameter of the star not in range 1..4000 or the star's vector not normalized)	FID_ILL_GSCDAT_IN_TCLDGSC	index of the bad entry in the *gscEntries* array		TM(1,8)	(221,12)	
41072	A070	illegal *start* value in TC_LOAD_GSC_INDICES command (*start*>215)	FID_ILL_START_IN_TCLDGSCIDX	received *start* value		TM(1,8)	(221,13)	
41073	A071	illegal *n* value in TC_LOAD_GSC_INDICES command (*n*>216-*start*)	FID_ILL_N_IN_TCLDGSCIDX	received *n* value	received *start* value	TM(1,8)	(221,13)	
41074	A072	illegal value in received *gscIndices* array in TC_LOAD_GSC_INDICES command	FID_ILL_IDXDAT_IN_TCLDGSCIDX	index of the bad entry in the *gscIndices* array	the bad value received	TM(1,8)	(221,13)	
41088	A080	illegal *start* value in TC_DUMP_CALIBRATION command (*start*>1147)	FID_ILL_START_IN_TCDMPCAL	received *start* value		TM(1,8)	(221,20)	
41089	A081	illegal *n* value in TC_DUMP_CALIBRATION command (*n*>1148-*start*)	FID_ILL_N_IN_TCDMPCAL	received *n* value	received *start* value	TM(1,8)	(221,20)	
41104	A090	illegal *start* value in TC_DUMP_GSC command (*start*>3999)	FID_ILL_START_IN_TCDMPGSC	received *start* value		TM(1,8)	(221,23)	

FID (dec)	FID (hex)	Description of detected error	Short Name	Param 1 (32-bit)	Param 2 (32 bit)	Report Type	Generating service	Reference
41105	A091	invalid *nToDump* value in TC_DUMP_GSC command (*nToDump*>4000-*start*)	FID_INV_NTODUMP_IN_TCDMPGSC	received *nToDump* value	received *start* value	TM(1,8)	(221,23)	
41106	A092	invalid *maxDumpSizeInBytes* value in TC_DUMP_GSC command (*maxDumpSizeInBytes*<16+4)	FID_INV_MXDMPSZ_IN_TCDMPGSC	received *maxDumpSizeIn Bytes* value		TM(1,8)	(221,23)	
41120	A0A0	illegal *start* value in TC_DUMP_GSC_INDICES command (*start*>215)	FID_ILL_START_IN_TCDMPGSCIDX	received *start* value		TM(1,8)	(221,24)	
41121	A0A1	invalid *nToDump* value in TC_DUMP_GSC_INDICES command (*nToDump*>216-*start*)	FID_INV_NTODUMP_IN_TCDMPGSCIDX	received *nToDump* value	received *start* value	TM(1,8)	(221,24)	
41122	A0A2	invalid *maxDumpSizeInBytes* value in TC_DUMP_GSC_INDICES command (*maxDumpSizeInBytes*<2+2)	FID_INV_MXDMPSZ_IN_TCDMPGSCIDX	received *maxDumpSizeIn Bytes* value		TM(1,8)	(221,24)	
41136	A0B0	illegal *MemId* in TC_ENABLE_MEMREAD command (received *MemId* is not allowed with the command)	FID_ILL_MEMID_IN_TCENMEMRD	received *MemID* value		TM(1,8)	(223,1)	
41137	A0B1	illegal *MemId* in TC_ENABLE_MEMWRITE command (received *MemId* is not allowed with the command)	FID_ILL_MEMID_IN_TCENMEMWR	received *MemID* value		TM(1,8)	(223,2)	
41138	A0B2	illegal *MemId* in TC_DISABLE_MEMREAD command (received *MemId* is not allowed with the command)	FID_ILL_MEMID_IN_TCDISMEMRD	received *MemID* value		TM(1,8)	(223,3)	
41139	A0B3	illegal *MemId* in TC_DISABLE_MEMWRITE command (received *MemId* is not allowed with the command)	FID_ILL_MEMID_IN_TCDISMEMWR	received *MemID* value		TM(1,8)	(223,4)	
41152	A0C0	command currently not enabled	FID_TC_NOT_ENABLED	service of the TC	subservice of the TC	TM(1,8)	(224,7)	
45055	AFFF	internal STR error	FID_INTERNAL_STR_ERROR	major error code	minor error code	TM(1,8)	all	

11 ENUMERATIONS

This section defines the enumerations used in command and telemetry data structures.

11.1 ENUMERATION 'APRIORIQUALITY'

Value	Label	Description
0	noApriori	No a-priori information specified
2	losApriori	Line of sight specified
3	completeApriori	Attitude and rate specified

11.2 ENUMERATION 'OPMODE'

Value	Label	Description
0	modeBoot	Boot mode
2	modeStandby	Standby mode
3	modePhoto	Photo mode
4	modeAadF	AAD mode with submode 'full frame'
5	modeAadW	AAD mode with submode 'window'
7	modeNat	NAT mode

11.3 ENUMERATION 'TECMODE'

Value	Label	Description
0	coolerOff	TEC switched off
1	coolerControlled	TEC operating in controlled mode to reach target temperature
2	coolerMaximum	TEC operating at maximum power

11.4 ENUMERATION 'SYNCSOURCE'

Value	Label	Description
0	syncSourceNone	no external synchronization
1	syncSourcePrimary	external synchronization via primary line
2	syncSourceSecondary	external synchronization via secondary line
3	syncSourceAny	external synchronization via XOR of both lines

11.5 ENUMERATION 'ATTRESULT'

Value	Label	Description
0	attitudeNotEnoughStars	Not enough stars suitable for attitude determination
+1	attitudeSuccess	Attitude calculated successfully
+2	attitudeRefined	Like attitudeSuccess but with additional a-posteriori refinement
< 0	N/A	Attitude determination failed

11.6 ENUMERATION 'IDRESULT'

Value	Label	Description
0	stidIdle	Star identification not yet started
+1	stidSuccess	Star identification succeeded
+2	stidRunning	Star identification is running
–1	stidNotEnoughStars	Not enough tracked stars for star identification
< –1	N/A	Star identification failed

11.7 ENUMERATION 'GAIN'

Value	Label	Description
0	sequencerGain1	Unity
1	sequencerGain2	Roughly ×2
2	sequencerGain4	Roughly ×4
3	sequencerGain8	Roughly ×8

11.8 ENUMERATION 'TRACKSTARSTATE'

Value	Label	Description
0	unused	Entry is not used
1	unprocessed	The track star has not been processed; x and y coordinates and dx and dy values are estimates
2	lost	The track star has been lost for this cycle; x and y coordinates and dx and dy values are estimates
3	detected	The track star has been detected in this cycle; x and y coordinates have maximum possible precision; other entries are derived from actual measurements
4	verified	Same as before ('detected'). Additionally: the track star has been verified

11.9 ENUMERATION 'ATTITUDEQUALITY'

Value	Label	Description
0	noAttitude	No attitude available
1	aPrioriAttitude	Attitude based solely on user input
5	coarseAttitude	Predicted from previous measurements
6	unconfirmedAttitude	Calculated from current measurements but not yet confirmed
7	validAttitude	Like unconfirmedAttitude but additionally validated

11.10 ENUMERATION 'RATEQUALITY'

Value	Label	Description
0	noRate	No rate information available
1	coarseRate	A-priori rate or rate derived from star coordinates
2	fineRate	Rate derived from attitude
3	filteredRate	Rate derived from attitude and filtered

11.11 ENUMERATION 'AUTOACQUISITIONMODE'

Value	Label	Description
0	autoAcquisitionOff	No automatic acquisition
1	autoAcquisitionWithDefaultsAfterReset	Automatic acquisition with default parameters after boot/reset
2	spare	Currently unused
3	spare	Currently unused

11.12 ENUMERATION 'STARTMODE'

Value	Label	Description
0	startBootHdl	copy the boot mode handler from ROM to the application runtime area and start it

Value	Label	Description
1	startApsw	if the check of the APSW in EEPROM is successful copy the APSW from EEPROM to the application runtime area in RAM and start it; otherwise copy the boot mode handler from ROM to the application runtime area and start it
2	startTestapp	perform a CRC check for the test application S/W in the reserved RAM area and if the check is successful copy the test app to the application runtime area and start it; otherwise copy the boot mode handler from ROM to the application runtime area and start it
3	startApswForced	regardless of the EEPROM check result copy the APSW from EEPROM to the application runtime area in RAM and start it

11.13 ENUMERATION 'FUNCTIONID'

Value	Label	Description
0	Selftest	identifies the selftest function

11.14 ENUMERATION 'MEMORYACCESSERRORID'

Value	Label	Description
1	MemoryWriteFailed	not all bytes have been written

Value	Label	Description
2	MemoryWriteProtected	the memory is write protected
3	MemoryReadProtected	the memory is read protected
4	MemoryNoBytesReadOrWritten	nothing has been read/written at all
5	MemoryAccessFailed	not all bytes have been read

12 INTERNAL STRUCTURE DEFINITIONS

This section defines sensor internal parameters which can be uploaded (i.e. modified) and downloaded using specific restricted commands. In principal all structures described here are located in RAM. Changes to the initial values are gone after a cold boot of the sensor.

12.1 PREPROCESSOR REGISTERS

This data structure is used in the TC_LOAD_PREPROCESSOR command and in the TM_PREPROCESSOR_DUMP TM packet. It contains the preprocessor register configuration and has a total size of 16 bytes.

The offsets given in the table are wrt. the start of the structure.

Parameter Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
inputMode	0: latched 1: normal	0	UI	1	0, 1	
spare		1	B	2		
compareMode	0: adaptive 1: fixed	3	UI	1	0, 1	
spare		4	B	3		
limitMode	0: none 1: fixed 2: positive criterion 3: positive and negative criterion	7	UI	2	0, 1, 2, 3	
smoothFactor a_i	0: 4x 1: 8x 2: 16x 3: 32x	9	UI	2	0, 1, 2, 3	
spare		11	B	1		

Parameter Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
smoothFactor b_i	0: 4x 1: 8x 2: 16x 3: 32x	12	UI	2	0, 1, 2, 3	
spare		14	B	2		
startValue	Start value for the ATF	16	UI	16		
fixedAverage		32	UI	16		
fixedLimit		48	UI	16		
positiveThreshold	Note: this is included only for completeness. It is overridden by the offset specified in the TC_ACQUIRE telecommand.	64	UI	12		
positiveHysteresis		76	UI	4		
negativeThreshold		80	UI	12		
negativeHysteresis		92	UI	4		
	spare	96	B	8		
verboseExtendedObjectReporting		104	B	1		
allowLineObjects		105	B	1		
objectChargeSum		106	UI	6		
	spare	112	B	2		
amplifierOffset	Offset of the programmable gain amplifier; value 0 corresponds to 0.8V; relevant for HAS detector only; ignored for others	114	SI	6		16mV
columnAmplifierOffset	Column amplifier offset; relevant for HAS detector only; ignored for others	120	SI	8		137µV

12.2 CALIBRATION DATA

This data structure contains the sensor calibration data of the STR. The structure has a total size of 1148 bytes.

The structure can be loaded with the TC_LOAD_CALIBRATION1, TC_LOAD_CALIBRATION2, TC_LOAD_CALIBRATION3, TC_LOAD_CALIBRATION4, TC_LOAD_CALIBRATION5, and TC_LOAD_CALIBRATION6 commands, each providing a specific part of the structure (note that partial load commands are necessary because the total length of this structure is 1148 bytes which exceeds the allowed TC packet length).

The TC_DUMP_CALIBRATION command dumps this structure in the form of a TM_CALIBRATION_DUMP packet.

Definition of the LSF Polynomial

(1) $\tan\alpha = f_x\left(x, y, T_{\deg C}\right) = -\left(\sum a_{i,j} x^i y^j + T_{\deg C} \sum a_{temp,m,n} x^m y^n\right)$ (X coordinate)

i,j= 0..3 and (i,j)= (0,4), (1,4), (4,0), (5,0); *m*,*n*= 0..1

(2) $\tan\beta = f_y\left(x, y, T_{\deg C}\right) = -\left(\sum b_{i,j} x^i y^j + T_{\deg C} \sum b_{temp,m,n} x^m y^n\right)$ (Y coordinate)

i,j= 0..3 and (i,j)= (0,4), (0,5), (4,0), (4,1); *m*,*n*= 0..1

The f_x and f_y polynomials are used to transform detector positions of found objects to the measurement reference frame including the correction of the low spatial frequency optical and temperature distortions.

Definition of the Decalibration (inverse) Polynomial

(3) $x = g_x\left(\tan\alpha, \tan\beta\right) = -\left(\sum c_{i,j} \left(\tan\alpha\right)^i \left(\tan\beta\right)^j\right)$ (X coordinate)

i,j= 0..3 and (i,j)=(0,4), (1,4), (4,0), (5,0)

(4) $y = g_y\left(\tan\alpha, \tan\beta\right) = -\left(\sum d_{i,j} \left(\tan\alpha\right)^i \left(\tan\beta\right)^j\right)$ (Y coordinate)

i,j= 0..3 and (i,j)= (0,4), (0,5), (4,0), (4,1)

The g_x and g_y polynomials are used to predict the detector positions for guide stars which are expected to be in the FOV.

The offsets given in the following table are wrt. the start of the structure.

Parameter Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
	Optics					
optMainPtX	Detector column coordinate (subpixel) of the optical main point	0	F	32		
optMainPtY	Detector column coordinate (subpixel) of the optical main point	32	F	32		
$a_{0,0}$	$a_{0,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	64	F	32		
$a_{0,1}$	$a_{0,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	96	F	32		
$a_{0,2}$	$a_{0,2}$ coefficient for transformation from detector(x,y) to tan(alpha)	128	F	32		
$a_{0,3}$	$a_{0,3}$ coefficient for transformation from detector(x,y) to tan(alpha)	160	F	32		
$a_{1,0}$	$a_{1,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	192	F	32		
$a_{1,1}$	$a_{1,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	224	F	32		
$a_{1,2}$	$a_{1,2}$ coefficient for transformation from detector(x,y) to tan(alpha)	256	F	32		
$a_{1,3}$	$a_{1,3}$ coefficient for transformation from detector(x,y) to tan(alpha)	288	F	32		
$a_{2,0}$	$a_{2,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	320	F	32		
$a_{2,1}$	$a_{2,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	352	F	32		
$a_{2,2}$	$a_{2,2}$ coefficient for transformation from detector(x,y) to tan(alpha)	384	F	32		
$a_{2,3}$	$a_{2,3}$ coefficient for transformation from detector(x,y) to tan(alpha)	416	F	32		
$a_{3,0}$	$a_{3,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	448	F	32		
$a_{3,1}$	$a_{3,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	480	F	32		
$a_{3,2}$	$a_{3,2}$ coefficient for transformation from detector(x,y) to tan(alpha)	512	F	32		
$a_{3,3}$	$a_{3,3}$ coefficient for transformation from detector(x,y) to tan(alpha)	544	F	32		
$a_{0,4}$	$a_{0,4}$ coefficient for transformation from detector(x,y) to tan(alpha)	576	F	32		
$a_{1,4}$	$a_{1,4}$ coefficient for transformation from detector(x,y) to tan(alpha)	608	F	32		
$a_{4,0}$	$a_{4,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	640	F	32		
$a_{5,0}$	$a_{5,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	672	F	32		
$a_{temp,0,0}$	$a_{temp,0,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	704	F	32		
$a_{temp,0,1}$	$a_{temp,0,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	736	F	32		
$a_{temp,1,0}$	$a_{temp,1,0}$ coefficient for transformation from detector(x,y) to tan(alpha)	768	F	32		
$a_{temp,1,1}$	$a_{temp,1,1}$ coefficient for transformation from detector(x,y) to tan(alpha)	800	F	32		
$b_{0,0}$	$b_{0,0}$ coefficient for transformation from detector(x,y) to tan(beta)	832	F	32		
$b_{0,1}$	$b_{0,1}$ coefficient for transformation from detector(x,y) to tan(beta)	864	F	32		
$b_{0,2}$	$b_{0,2}$ coefficient for transformation from detector(x,y) to tan(beta)	896	F	32		
$b_{0,3}$	$b_{0,3}$ coefficient for transformation from detector(x,y) to tan(beta)	928	F	32		
$b_{1,0}$	$b_{1,0}$ coefficient for transformation from detector(x,y) to tan(beta)	960	F	32		

Parameter Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
$b_{1,1}$	$b_{1,1}$ coefficient for transformation from detector(x,y) to tan(beta)	992	F	32		
$b_{1,2}$	$b_{1,2}$ coefficient for transformation from detector(x,y) to tan(beta)	1024	F	32		
$b_{1,3}$	$b_{1,3}$ coefficient for transformation from detector(x,y) to tan(beta)	1056	F	32		
$b_{2,0}$	$b_{2,0}$ coefficient for transformation from detector(x,y) to tan(beta)	1088	F	32		
$b_{2,1}$	$b_{2,1}$ coefficient for transformation from detector(x,y) to tan(beta)	1120	F	32		
$b_{2,2}$	$b_{2,2}$ coefficient for transformation from detector(x,y) to tan(beta)	1152	F	32		
$b_{2,3}$	$b_{2,3}$ coefficient for transformation from detector(x,y) to tan(beta)	1184	F	32		
$b_{3,0}$	$b_{3,0}$ coefficient for transformation from detector(x,y) to tan(beta)	1216	F	32		
$b_{3,1}$	$b_{3,1}$ coefficient for transformation from detector(x,y) to tan(beta)	1248	F	32		
$b_{3,2}$	$b_{3,2}$ coefficient for transformation from detector(x,y) to tan(beta)	1280	F	32		
$b_{3,3}$	$b_{3,3}$ coefficient for transformation from detector(x,y) to tan(beta)	1312	F	32		
$b_{0,4}$	$b_{0,4}$ coefficient for transformation from detector(x,y) to tan(beta)	1344	F	32		
$b_{0,5}$	$b_{0,5}$ coefficient for transformation from detector(x,y) to tan(beta)	1376	F	32		
$b_{4,0}$	$b_{4,0}$ coefficient for transformation from detector(x,y) to tan(beta)	1408	F	32		
$b_{4,1}$	$b_{4,1}$ coefficient for transformation from detector(x,y) to tan(beta)	1440	F	32		
$b_{temp,0,0}$	$b_{temp,0,0}$ coefficient for transformation from detector(x,y) to tan(beta)	1472	F	32		
$b_{temp,0,1}$	$b_{temp,0,1}$ coefficient for transformation from detector(x,y) to tan(beta)	1504	F	32		
$b_{temp,1,0}$	$b_{temp,1,0}$ coefficient for transformation from detector(x,y) to tan(beta)	1536	F	32		
$b_{temp,1,1}$	$b_{temp,1,1}$ coefficient for transformation from detector(x,y) to tan(beta)	1568	F	32		
$c_{0,0}$	$c_{0,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1600	F	32		
$c_{0,1}$	$c_{0,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1632	F	32		
$c_{0,2}$	$c_{0,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1664	F	32		
$c_{0,3}$	$c_{0,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1696	F	32		
$c_{1,0}$	$c_{1,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1728	F	32		
$c_{1,1}$	$c_{1,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1760	F	32		
$c_{1,2}$	$c_{1,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1792	F	32		
$c_{1,3}$	$c_{1,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1824	F	32		
$c_{2,0}$	$c_{2,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1856	F	32		
$c_{2,1}$	$c_{2,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1888	F	32		
$c_{2,2}$	$c_{2,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1920	F	32		
$c_{2,3}$	$c_{2,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1952	F	32		
$c_{3,0}$	$c_{3,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	1984	F	32		

Parameter Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
$c_{3,1}$	$c_{3,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	2016	F	32		
$c_{3,2}$	$c_{3,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	2048	F	32		
$c_{3,3}$	$c_{3,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	2080	F	32		
$c_{0,4}$	$c_{0,4}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	2112	F	32		
$c_{1,4}$	$c_{1,4}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	2144	F	32		
$c_{4,0}$	$c_{4,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	2176	F	32		
$c_{5,0}$	$c_{5,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(x)	2208	F	32		
$d_{0,0}$	$d_{0,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2240	F	32		
$d_{0,1}$	$d_{0,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2272	F	32		
$d_{0,2}$	$d_{0,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2304	F	32		
$d_{0,3}$	$d_{0,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2336	F	32		
$d_{1,0}$	$d_{1,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2368	F	32		
$d_{1,1}$	$d_{1,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2400	F	32		
$d_{1,2}$	$d_{1,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2432	F	32		
$d_{1,3}$	$d_{1,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2464	F	32		
$d_{2,0}$	$d_{2,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2496	F	32		
$d_{2,1}$	$d_{2,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2528	F	32		
$d_{2,2}$	$d_{2,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2560	F	32		
$d_{2,3}$	$d_{2,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2592	F	32		
$d_{3,0}$	$d_{3,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2624	F	32		
$d_{3,1}$	$d_{3,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2656	F	32		
$d_{3,2}$	$d_{3,2}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2688	F	32		
$d_{3,3}$	$d_{3,3}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2720	F	32		
$d_{0,4}$	$d_{0,4}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2752	F	32		
$d_{0,5}$	$d_{0,5}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2784	F	32		
$d_{4,0}$	$d_{4,0}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2816	F	32		
$d_{4,1}$	$d_{4,1}$ coefficient for transformation from tan(alpha),tan(beta) to detector(y)	2848	F	32		
s_x	s-curve correction factor for x-correction	2880	F	32		
s_y	s-curve correction factor for y-correction	2912	F	32		
tcRef0	reference total charge for t_{int}=1s and gain=1.0 and magnitude=0.0	2944	F	32		
sigmaX	size of star shape in x direction	2976	F	32		pixel
sigmaY	size of star shape in y direction	3008	F	32		pixel

Parameter Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
	Point Defect Correction Table					
x_1	x coordinate of first defect pixel	3040	UI	10	0..1023	
y_1	y coordinate of first defect pixel	3050	UI	10	0..1023	
$mode_1$	0: no correction 1: copy from (x_1-1,y_1) 2: copy from (x_1+1,y_1) 3: average	3060	UI	2	0..3	
$spare_1$	filler bits	3062	B	2		
x_2	x coordinate of second defect pixel	3064	UI	10	0..1023	
y_2	y coordinate of second defect pixel	3074	UI	10	0..1023	
$mode_2$	0: no correction 1: copy from (x_2-1,y_2) 2: copy from (x_2+1,y_2) 3: average	3084	UI	2	0..3	
$spare_2$	filler bits	3086	B	2		
...	...					
x_{256}	x coordinate of last defect pixel	9160	UI	10	0..1023	
y_{256}	y coordinate of last defect pixel	9170	UI	10	0..1023	
$mode_{256}$	0: no correction 1: copy from (x_{256}-1,y_{256}) 2: copy from (x_{256}+1,y_{256}) 3: average	9180	UI	2	0..3	
$spare_{256}$	filler bits	9182	B	2		

12.3 ALGORITHM PARAMETERS

This data structure is used in the TC_DUMP_ALGORITHM and TC_LOAD_ALGORITHM commands. It contains the algorithm control parameters and has a total size of 116 bytes.

The offsets given in the table are wrt. the start of the structure.

Parameter Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
temperatureGain	Gain for temperature measurement	0	F	32		K/LSB
temperatureOffset	Offset for temperature measurement	32	F	32		°C
tecGainA1	Gain A1 for temperature controller	64	F	32		
tecGainB0	Gain B0 for temperature controller	96	F	32		
tecGainB1	Gain B1 for temperature controller	128	F	32		
maximumRate	maximum supported rate	160	F	32		deg/s
fullframeUncertainty	uncertainty distance used during full frame mode	192	F	32		pixel
windowUncertainty	uncertainty distance used during window mode	224	F	32		pixel
minimumGaussSignal	minimum required signal in gauss based star detection	256	F	32		LSB
initialScalingFactor	initial value used for the focal length scaling filter	288	F	32		
aPosterioriMaximumDistance	the maximum allowed length of the difference vector between measured and reference coordinates for a star to be selected for attitude refinement	320	F	32		
minRotation	the minimum rotation from which non-zero dx/dy may be derived	352	F	32		rad
attitudeGain	Filter gain for attitude quaternion (used in rate filter)	384	F	32		
rotationGain	Filter gain for rotation quaternion (used in rate filter)	416	F	32		
exponentialMagnitudeWeightForQuest	exponent for magnitude (used in calc. of weights for attitude determination)	448	F	32		
$spare_1$	reserved	480	B	64		
dt	nominal cycle time of STR in ms	544	UI	16		1 ms
defaultTargetTemperature	the default target temperature after power-on/reset	560	SI	16	see TC_TEMPERATURE	10^{-1}°C
fixedOffset	if non-zero, this fixed offset above background will be used for binarization	576	UI	16		

Parameter Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
offsetMin	minimum offset in LSB above background for binarization	592	UI	16		
offsetWin	offset in LSB above background for binarization during window mode at 1s integration time; is overridden by fixedOffset (when non-zero)	608	UI	16		
offsetStart	start offset in LSB above background for binarization at 1s integration time	624	UI	16		
$spare_2$	reserved	640	B	32		
numFreeRunningBetweenSync	expected number of free-running cycles between cycles started by sync pulse	672	UI	8		
tecMax	maximum PWM value for the TEC	680	UI	8		
defaultRateLimit	the default rate limit used when no rate limit is specified (for instance in automatic acquisition); a value of 0 means: default is maximumRate	688	UI	8	see TC_RATE_LIMIT	2^{-5} deg/s
tracklistLength	maximum number of tracked stars	696	UI	8	16..64	
fixedIntegrationTime	if non-zero, this fixed integration time (in ms) will be used during acquisition	704	UI	8		
tIntMin	lower integration time limit in ms	712	UI	8		1 ms
tIntMax	upper integration time limit in ms	720	UI	8		1 ms
reservedTimeForReadOut	time span reserved for read-out in window mode	728	UI	8		1 ms
offsetDelta	offset change in LSB (at 1s integration time) for one cycle	736	UI	8		
maxStarPropagationCycles	maximum number of cycles a single star will be propagated until it will be removed from track list	744	UI	8		
minQualityForTracked	minimum quality of a star to be declared "tracked"	752	UI	8		
dxDyFilterLength	weight of previous dx and dy measurements when updating them with new values during tracklist update	760	UI	8	1,2,4,8	

Parameter Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
numStarsWindowEntry	minimum number of tracked stars needed for transition to window mode	768	UI	8		
numStarsWindowExit	limit of tracked stars for fallback from window into fullframe mode	776	UI	8		
minStarsRate	minimum number of stars needed for rate estimation	784	UI	8		
maxStarsForFirstAttitudeStage	the first stage in attitude determination will always be performed with no more than this many of the brightest stars; the attitude can later be refined when a-posteriori correction is enabled	792	UI	8		
minVerifiedForAttitudeValid	minimum number of verified stars necessary for declaring valid the attitude	800	UI	8		%
minIdStarsForVerify	number of identified stars necessary to verify and not identify anymore	808	UI	8		
allowedAcquisitionDurationSeconds	if time span (in seconds) from start of acquisition to reaching NAT mode exceeds this limit, an EV_LONG_ACQUISITION is issued; a value of 0 means: no restrictions	816	UI	8		s
starIdAngTol	angular tolerance in arcsec used in star identification	824	UI	8		
maximumGaussWindowSizeAad	maximum window size supported by Gauss based centroiding in AAD mode	832	UI	8		pixel
maximumGaussWindowSizeNat	maximum window size supported by Gauss based centroiding in NAT mode	840	UI	8		pixel
fpnFilterLength	weight of the previous background values when updating them with new samples	848	UI	8		

Parameter Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
allowedProcessingTimeInMs	if the processing time exceeds this limit, an EV_CYCLE_TOO_LONG event is issued; a value of 0 means: no limitations	856	UI	8		ms
minimumSpareTimeInMs	if the processing time extends into the interval of this length before the end of the cycle, an EV_CYCLE_OVERRUN event is issued; a value of 0 means: no limitation	864	UI	8		ms
minStarsSsva	the SSVA does not have any effect if the number of remaining stars after application of the algorithm would fall below this value; the SSVA will not be applied if minStarsSsva<4 or minStarsSsva>15	872	UI	8		
defaultSyncSource	default synchronization after reset	880	E	2	see SyncSource enumeration	
isInvertedSyncPolarity	If set, synchronization is performed on the *falling* edge of the synchronization source; otherwise on the rising edge	882	B	1		
autoAcquisition	automatic acquisition mode	883	E	2	see AutoAcquisitionMode enumeration	
enableScaleFilter	If set, the determined focus scaling factor will be applied	885	B	1		
enableAposterioriCorrection	If set, a-posteriori correction of the attitude measurement will be performed	886	B	1		
enableSmallestWindowSize	If set, window size 9 is enabled. If unset, the smallest window size is 15.	887	B	1		
enableRateFilter	If set, rate filtering is enabled.	888	B	1		

Parameter Name	Description	Offset (bit)	Format	Length (bit)	Range	Unit
fallbackToFullFrameAfterIdentificationFailure	If set, the sensor falls back to full frame mode after an identification failure. If unset and there are enough tracked stars, the identification is attempted again.	889	B	1		
enableFixedFullFrameMode	if set, the detector full frame read, transition to window mode inhibit	890	B	1		
enableStreakMode	if set, streak mode is enabled in full frame mode	891	B	1		
enableBlackLevelCorrection	if set, detector read-outs will be black-level corrected	892	B	1		
enableFpnFilter	if set, the fixed pattern noise filter will be enabled	893	B	1		
gain	ADC gain	894	E	2	see Gain enumeration	
enableFpnForNonGuideStars	if set, the fixed pattern noise filter will be used also for non-guide-stars	896	B	1		
enableTrackingAtFovBorder	if set, tracking is performed also at the border of the illuminated detector area	897	B	1		
$spare_3$	reserved	898	B	30		

12.4 GUIDE STAR CATALOGUE

This data structure is used in the TC_DUMP_GSC and TC_LOAD_GSC commands. It describes the Guide Star Catalogue (GSC).

The GSC of the STR can have up to 4000 entries each having a length of 16 bytes resulting in a total catalogue length of about 64 kilo bytes.

Each GSC entry describes one guide star and has an associated catalogue number which must be unique within the catalogue. The catalogue numbers range from 1 to 4000. The instrumental magnitude of each star is specified in LSB on the basis of an integration time of 1 second. The vector (v_x, v_y, v_z) represents the direction to the star in the geo-centered mean equatorial system of J2000 (ICRF) and must have norm 1.

For the TC_DUMP_GSC command the catalog entries are indexed from 0 to 3999. The *start* and *nToDump* parameters provided with a TC_DUMP_GSC command specify which part of the catalogue shall be dumped, where *start* specifies the 0-based index of the first GSC entry to be dumped and *nToDump* specifies the number of entries to be dumped.

Only complete GSC entries may be loaded or dumped, i.e. the load or dump of a partial GSC entry is not supported.

Parameter Name	Description	Offset (byte)	Format	Length (byte)	Range	Unit
cnr1	catalogue number of the 1st guide star (1st catalog entry)	0	UI16	2	1	n.a.
mag1	total charge of the 1st guide star in detector LSB	2	UI16	2		[LSB]
$v1_x$	unit vector of the 1st guide star	4	F	4	−1.0 … +1.0	n.a.
$v1_y$		8	F	4	−1.0 … +1.0	n.a.
$v1_z$		12	F	4	−1.0 … +1.0	n.a.
cnr2	catalogue number of the 2nd guide star (2nd catalog entry)	16	UI16	2	2	n.a.
mag2	total charge of the 2nd guide star in detector LSB	18	UI16	2		[LSB]
$v2_x$	unit vector of the 2nd guide star	20	F	4	−1.0 … +1.0	n.a.
$v2_y$		24	F	4	−1.0 … +1.0	n.a.
$v2_z$		28	F	4	−1.0 … +1.0	n.a.
…						
cnr4000	catalogue number of last guide star (last catalog entry)	40384	UI16	2	4000	n.a.
mag4000	total charge of last guide star in detector LSB	40386	UI16	2		[LSB]
$v4000_x$	unit vector of the last guide star	40388	F	4	−1.0 … +1.0	n.a.
$v4000_y$		40392	F	4	−1.0 … +1.0	n.a.
$v4000_z$		40396	F	4	−1.0 … +1.0	n.a.

12.5 GUIDE STAR CATALOGUE INDEX

The Guide Star Catalogue is tiled in 216 tiles which are numbered from 0 to 215. Each GSC tile covers a sequential group of guide stars and the catalogue number of the last guide star covered by each tile is stored in the Guide Star Catalogue Index, i.e. the GSC Index is an array of numbers, each specifying the last star covered by the dedicated GSC tile.

Parameter Name	Description	Offset (byte)	Format	Length (byte)	Range
gscIndex[0]	catalogue number of the last guide star covered by GSC tile 0.	0	UI16	2	1..4000
gscIndex[1]	catalogue number of the last guide star covered by GSC tile 1.	2	UI16	2	1..4000
…					
gscIndex[215]	catalogue number of the last guide star covered by GSC tile 215	430	UI16	2	1..4000

The Guide Star Catalogue Index may be dumped and loaded with TC_DUMP_GSC_INDICES and TC_LOAD_GSC_INDICES commands.

13 MEMORY MAP

This section describes the internal memory allocation and usage, helping to find the right addresses for upload, dumping and checking of memory content using TC_LOAD_MEMORY, TC_DUMP_MEMORY and TC_CHECK_MEMORY commands.

The most important MemID values have been selected according to the table given in AD1, section B2.10. However, according to a finer granulation required by the STR several additional MemID s have been defined. Note further that for the 4 I/O related areas only 32-bit access mode (incl. 32-bit address alignment) is supported. For all other memories no special alignment is required and the access is byte-wise.

Address Range (hex)	Type	Memory ID (hex)	Description	SAU [bytes]	MUL [bytes]	Read	Write
00000000-00001fff	ROM	1	Boot loader	1	1	X	–
00002000-0001ffbf			Boot Mode Handler				
0001ffc0-0001ffff			STR Device Identification: Model, SerNo Look-up table: RT Adress ➔ PRID				
00080000-000fefff	EEPROM	11	Application S/W (header+storage)[1]	1	1	X	X[2]
000ff000-000ffeff		51	Calibration Data storage				X[2]
000fff00-000fffff		61	EEPROM statistics				–
10000000-1000007f	I/O	A3	ASIC registers	4	4	X[3]	X[2]
20000000-2000003f		B3	RT controller registers				
20010000-2001ffff		C3	RT controller RAM				
40000000-4017efff	RAM	21	Application runtime	1	1	X	X[2]
4017f000-4017ffff		91	Communication failure buffer				
40180000-401fefff		71	Test Application upload area[1]				
401ff000-401ffeff		81	Calib Data upload area for TestApp				
401fff00-401fffff		31	SGM area in RAM				
80000000-ffffffff	I/O	D3	Processor registers	4	4	X[3]	X[2]

Table 13-1 Memory Map

[1] the last 64kByte contain the GSC
[2] write access via TC(6,2) is only accepted if currently enabled (see sections 5.12 and 6.9.2)
[3] read access via TC(6,5) and TC(6,9) is only accepted if currently enabled (see sections 5.12 and 6.9.1)

The MemID values given in Table 13-1 are valid for all STRs. Please see section 13.1 for details on memory ID 91.

13.1 ORGANIZATION OF THE COMMUNICATION FAILURE BUFFER

13.1.1 DATA STRUCTURE

The structure of the Communication Failure Buffer (memory ID 91) is defined in Table 13-2:

Name	Description	Offset (byte)	Format	Length (byte)	Range
errorTimeCode	The current time stamp is saved here	0	T	6	
spare	alignment bytes	6	B	2	
errorCode	This is the error code detected by the session layer	8	E	4	See Table 13-3
blockBufferPreviousSize	This is the size in bytes of the previously received block	12	UI32	4	1…1024
$blockBufferPrevious_0$	This is the data of the previously received block	16	UI8	1	
…		…	…	…	
$blockBufferPrevious_{1023}$		1039	UI8	1	
blockBufferCurrentSize	This is the size in bytes of the block currently in progress	1040	UI32	4	1…1024
$blockBufferCurrent_0$	This is the block currently received	1044	UI8	1	
…		…	…	…	
$blockBufferCurrent_{1023}$		2067	UI8	1	
rxErrorBufferSize	This is the size in bytes of the fragment currently decoded	2068	UI32	4	1…226
$rxErrorBuffer_0$	This is the data of the current fragment	2072	UI8	1	
…		…	…	…	
$rxErrorBuffer_{225}$		2297	UI8	1	
spare	alignment bytes	2298	B	2	
rxWrite	This is the pointer where the next write in the fragment will go to	2300	UI32	4	0…65535
rxWriteLeft	This is the amount of bytes left in the fragment before it is a PUS packet	2304	UI32	4	0…65535
blockBufferRead	This is the read pointer into the current received block at the point of the error	2308	UI32	4	0…1023

Table 13-2 Structure of the Communication Failure Buffer

13.1.2 Error Codes

These error codes are used in the Communication Failure Buffer

Value	Label	Description
0	milSessionLayerNoError	No Error detected (initial value)
1	milSessionLayerInvalidePaketHeaderLength	Invalid PUS packet header length field
2	milSessionLayerInvalideFirstHeaderPointer	Invalid first header pointer in the block header
3	milSessionLayerOutOfMemory	Not enough memory to reassemble the PUS packet
4	milSessionLayerNotEnoughBytes	Not enough bytes for a valid block received

Table 13-3 Error Codes in the Communication Failure Buffer

13.2 Numeric Software Version

Each software release is identified by an unsigned 16-bit number (UI16 format). This identifier can be accessed via service TC(6,5) at the following addresses:

Address (hex)	Type	Memory ID (hex)	Description
000d0030	EEPROM	11	Identifier of the application stored in EEPROM
40050000	RAM	21	Identifier of the currently running software (identical to the swVersion in §4.2)
401d0030	RAM	71	Identifier of the test application

Table 13-4 Location of Numeric Software Version Identifiers