MUPUS User Manual
& Onboard Software Description
- Version 7.0, 7.1, 7.2, 7.3, 7.4 -

RO-LMU-SP-3401-DLR

Version; 2.2

Revision date: 14.04.2017

Prepared by:

Dr. Jörg Knollenberg

Rolf Schrödter

Approval Sheet

	Name	Signature	Date
Prepared by:	Dr. Jörg Knollenberg		10.11.2005
	Rolf Schrödter		
Approved by:			
Approved by:			

Document Change Record

Vers.	Rev.	Date	Pages affected	Modification	DCR	Initials
A	a	11/2005	All	First draft	None	RoS
		02/2006		Document review by J.Knollenberg		JK
		06/2006		MUPUS software version 4.6b used as fallback		
		08/2006	all	Additions by J.Knollenberg		JK
		01/2007	56	Updates HK[26] Deployment Device Status flags (version 7.0 only)		RoS
		01/2007	42,49,64	Added software version 7.1 changes		RoS
		02/2007	42,59	Added HK values: TcmdReceived,TcmdExec,TcmdError		RoS
		02/2007	49	Added telecommands TestDelay & FuseHardware		RoS
		02/2008	31-32	Added MUPUS configuration flag values		RoS
		02/2008	33,44-45	Added software version 7.2 changes		RoS
		02/2008	75	Added missing BackupRAM telemetry frame description (type=0x7C)		RoS
		02/2008	54-	Added more detailed telecommand desription		RoS
				Updated Chapter 6 on operational modes. Minor corrections to all chapters		JK
		10/2009		New Mupus software version 7.3: - Added THC power measurements (+12V current for all heaters & voltage) - Added new TM frame type=0x75 - Added optional heating pause to reduce overall heating power		RoS
		01/2010		Further changes with version 7.3 - increased timeout in HAMMER mode to 12 seconds (to comply with hammer capacitor charge times) - update information to V7.3 in relevant chapters		JK
		04/2011		Added description and data frame structure of “time-stamped” ANCHOR mode, Added description of GEAR mode and added some details on the description of HAMMER mode		JK
		05/2013		Changed description of ANCHOR mode to reflect the actual baseline for SDL		JK
		12/2013		New Mupus software version 7.4: - changed HAMMER mode such that DD boom will always be retracted (even if configured depth no reached) - changed ARM mode to cope with		JK

				blocking during deployment - make constants controlling deployment/insertion configurable - added 5 s delay in ARM mode between 2nd release heating and start of motor (patch TC unnecessary) - added sending OPCL at end of successful deployment in ARM mode (patch TC unnecessary) - changed conversion coefficients for LG parameters according to latest LUM 5-1 (patch TC unnecessary) - keep CDMS interrupts enabled during ANCHOR mode (patch TC unnecessary) - added flag TM_OPS_HEATER_ON to TM status word		
		04/2014		Added h/k conversion coefficients to h/k packet description		JK
		08/2014		Added special TCs (patches) for deployment and retraction		JK
		06/2015		Added chapter 10, description of calibration		JK
		02/2017		Reworked chapter 10, added 10.4 for the DS calibration		JK
		04/2017		Corrected some typos in chapter 10.1/10.2, added note about DS operation on comet to 10.4		JK

Table of Contents

Table of Illustrations

List of Tables

Abbreviations

NCR	Non-Conformance Report
SCR	Space Research Centre, Polish Academy of Sciences, Warsaw
RCS, CS-RCS	Revision Control System, keeping track of software modifications
BIOS	Basic Input and Output system used by the Common-DPU
Common-DPU	A universal processor unit used for five Rosetta Lander instruments
ROLIS/CIVA	Camera experiment at the Rosetta Lander
PENEL	Penetrator Electronics
TM	Thermal Mapper (infrared sensors)
PENTS	Penetrator electronics temperature sensor
STCB	Stored Telecommand Buffer for each instrument in CDMS (version 4.6b/6.1 only)
CDMS	Rosetta Lander Command and Data Management System
BRAM	Backup-RAM buffers in CDMS
HK	Housekeeping data
TBC	To Be Confirmed (later)
TBD	To Be Defined (later)
DPU	Data Processing Unit, RTX microprocessor board
RTX, RTX-2010	Intersil/Harris 16-bit microprocessor, RTX=Real-Time-eXpress
SR	CDMS Service Request
LOBT	Lander onboard time send by CDMS

Table of Relevant Documents

Table 1. Table of relevant documents.

[1]	RO-LMU-SP-3401-SRC	Flight Software Document, Ver. Final Rev. 3.1
[2]	Rosetta Lander Common-DPU	Flight Model (FM), Users Manual (July 2000)
[3]	LMI UR/Forth	LMI UR/FORTH manual
[4]	RO-LCD-SP-3101	Command and Data Management System (CDMS) – Subsystem Specification

1 Introduction

This document serves as a User Manual and describes the MUPUS instrument software *version 7.ff* architecture, the MUPUS working modes and a complete reference of telecommands for instrument control and of MUPUS housekeeping and science data frames. It contains all necessary information to operate the instrument and understand the generated MUPUS data.

Chapter 2 describes the main ideas behind the new MUPUS software version 7.0.

Chapter 3 describes the MUPUS hardware resources from a software point of view

Chapter 4 describes the general software architecture, the multitasking RTX kernel, the way of handling telecommands and executing MUPUS working modes and telemetry data buffering.

Chapter 0 gives a detailed description of the MUPUS operational modes.

Chapter 8 is a complete reference of MUPUS telecommands.

Chapter 9 is a complete reference of MUPUS housekeeping and telemetry data.

2 MUPUS software version 7.0

2.1 MUPUS software versions 4.6 to 6.1

Software version 4.6b (fallback version)

MUPUS software *version 4.6* is originally installed on the flight model (status of 2005).

Although it has a few open NCR's (30357,30358,30359,30369,30376) the most critical mechanical part (deployment, retraction, hammering) has been tested most intensively in laboratory as well as in the thermal-vacuum chamber.

For these formal reasons it has been decided to use MUPUS software *version 4.6* as the fallback software, for the case that the new *version 7.0* fails for the mechanical part.

To keep the software upload small *version 4.6* is reduced to the mechanical part only (17 kBytes instead of 40 kBytes), leaving EEPROM space for future patches of the new flight software *version 7.0*.

☝ ***Software version 4.6b serves as the fallback software. It is reduced to the mechanical part only (deployment, retraction, hammering) and the ANCHOR mode. For a complete documentation of version 4.6, see the original MUPUS instrument documentation.***

The following table briefly shows the main MUPUS software *version 4.6b* characteristics. For details refer to the original MUPUS instrument documentation.

Table 2: MUPUS software version 4.6 characteristics

Mupus Software version identifier (MUPUS boot frame):		
	"***MUPUS-FM Ver.46b***", followed by the compilation date & time	
Supported telecommands:		
	A422 0000 5BDE	ANCHOR mode
	A433 0000 0000 0000 0000 0000 (5BCD) **A433** parm1 parm2 parm3 parm4 chksum	ARM mode
	A444 0000 0000 0000 0000 0000 (5BBC) **A444** parm1 parm2 parm3 parm4 chksum	HAMMER mode
	B588 0000 4A78	HARPOON mode (2^{nd} anchor shot)
Telemetry frames:		
	TBD	
Housekeeping data:		
	The same housekeeping data structure as for the original *version 4.6*. Obviously only values affected by HAMMER,ARM,ANCHOR and HARPOON telecommands are filled with meaningful values.	

Software version 4.6 to 6.1 history

MUPUS onboard software *version 6.1* is based on the original flight software *version 4.6*. This software version has been refined by the original authors (Marek Hlond, Marek Banaszkiewicz, SRC Warsaw) to *version 5.8* with the following changes (brief overview):

Version 5.5	Fixes addressing NCR's 30357,30358,30359,30369,30376.
Version 5.7	Implementation of a new "triad" method for PEN sensor measurements for

	TPPROBE mode (file "TPPROBE57.4TH")
Version 5.8	Implementation of the same new measurement even for LONGTERM (file "HAMM25B.SCR") Added M.Hlond's version 5.8 patches (replacing files MUPUS58.4TH, PENEL58.4TH)

☝ ***Software version 5.8 is the last official onboard software delivered by SRC Warsaw.***

As some of the above NCR's related to CDMS communication could not be reproduced with the ground support equipment, but were only be observed in flight configuration, it has still to be shown, that version 5.8 solves all above NCR's. However from the applied changes (adding timeouts to CDMS services) it can be assumed, that CDMS communication and scientific measurements of version 5.8 work better than the current flight version 4.6.

Without modification of software functionality the following changes have been made leading to onboard software *version 6.1*:

Version 5.7	Put software source under revision control (RCS) using text files "*.4TH" instead of binary .SCR files
	Remove unused code and variables
	Move larger buffers (>16 bytes) to end-of-memory, not occupying target space; Initialize memory buffers at startup
	Replace CONSTANT's by EQU's, not occupying target space
Version 6.0	Replace FMOVE (dangerous for stack usage) by WMOVE
	Analyzing and commenting the mechanical part (HAMMER.4TH) together with the author M.Banaszkiewicz, RSC Warsaw.
	Removed internal hammer test code (hammer mode=6) not applicable for flight
Version 6.1	Few minor bug fixes

☝ ***Software version 6.1 is completely functional identical to version 58 and could alternatively serve as the fallback software. For a complete documentation of version 6.1, see [1]: "Flight Software Document, Ver. Final Rev. 3.1"***

2.2 Software version 7.0

The actual development branch *version 7.0* is a complete rewrite of the MUPUS onboard software based on the BIOS & CDMS kernel successfully running on the ROLIS/CIVA instrument observing the following rules:

- Keep the critical mechanical part (deployment device, release procedures, hammering) almost unchanged.
- Implement a single PEN sensor measurement procedure used by all MUPUS modes.
- Replace multiple heating procedures by a single heating method used before PENEL, Thermal Mapper (TM) are powered on.
- Replace the PENEL continuous heating method by a procedure controlled by real temperature measurements using HK4 sensor (PENTS_ASSIST).
- Keep all MUPUS configuration settings in EEPROM, instead of CDMS-STCB, making MUPUS configuration more straight and avoiding conflicts with parallel existing MUPUS software version 6.1.
- Keep Backup-RAM used for data exchange with other instruments (SESAME/PP, Landing GEAR) compatible.

- Add more housekeeping (HK) values (64 instead of 32) by decreasing the HK frequency (1 block per 2 minutes instead of 1 block per minute).

2.3 MUPUS dual-boot management

Because MUPUS software version 7.0 is a complete rewrite it seems to be a good idea to have a fallback software onboard. By applying the changes described in 2, version 4.6 has been compressed to <17 Kbytes to make room for another software version 7.0 in the 64 Kbytes EEPROM. The following EEPROM memory map illustrates the location of the software images:

Address (hex)	Code / Data
0000…ACFF	Software version 7.0 and optional patches, max. 43 Kbytes
AD00…AFFF	Configuration data for version 7.x
B000…FFFF	Software version 4.6b and patches, max. 20 Kbytes

1. The software upload to the EEPROM is performed by Common-DPU debug-monitor telecommands using the following procedure (ref. to document [2]):
2. Upload software version 7.0 to e.g. memory page=1, address=0 by sending ~500 DEB0/DEB1/DEB2 telecommands
3. Burn the uploaded code too EEPROM address=0000

    telecommand: DEB3 0000 214D

4. Upload software version 4.6b to e.g. memory page=1, address=0 by sending ~500 DEB0/DEB1/DEB2 telecommands
5. Burn the uploaded code too EEPROM address=B000

    telecommand: DEB3 B000 714D

MUPUS dual boot procedure

The following options are applicable for booting the MUPUS instrument software:

1. Power on MUPUS and wait. After one minute the default software at EEPROM address=0000 (version 7.0) is booted automatically.
2. Power on MUPUS and send the following telecommand within one minute after power on to boot software version 7.0:

    telecommand: DEBD 0000 2143

3. Power on MUPUS and send the following telecommand within one minute after power on to boot software version 4.6b:

    telecommand: DEBD B000 7143

3 MUPUS hardware resources

3.1 DPU memory usage

Table 3: EEPROM memory map

Addr		**Name**	**Usage**
Mupus Software version 7.0			
	0000...0001	Length (1 word)	Number of Code words
	0002…0003	Checksum (1 word)	Negated Sum of all code words
	0004…→ max ACFF	Code (Length words)	Software code and initialized data
	AD00.. →max ADFF	Config	Software v.7.0 & v7.1 configuration
	AE00…→max AFFF	Config	Software v.7.0 & v7.1 configuration
Mupus Software version 4.6b			
	B000...B001	Length (1 word)	Number of Code words
	B002...B003	Checksum (1 word)	Negated Sum of all code words
	B004…→ max FFFF	Code (Length words)	Software code and initialized data

RAM-Page & Addr			**Name**	**Usage**
Mupus Software version 7.0				
	RAM page=0			
		0000…→ max. ADFB	Code	Software code and initialized data
		min. ADFC ←…FFFF	Buffers	Buffers, initialized to ZERO
	Upper memory pages (can be configured)			
		Page=1	TM	Telemetry frame buffer, max. 255 frames
		Page=2	DATA1	Raw ADC mode data Main ANCHOR measurements
		Page=3	DATA2	Redundant ANCHOR measurements

3.2 DPU interrupts

The following table lists all used and unused RTX-2010 interrupts for the MUPUS software version 7.0. Interrupt are listed with increasing priority, lowest priority first. Highest priority is defined by Common-DPU hardware for the CDMS interrupt.

Table 4: MUPUS interrupt usage

Interrupt	**Procedure**	**Usage**
swi	NOOP	<not used>
ei5	NOOP	FM: <not used> / EM: optional UART interrupt
ei4	TBD	Touch down interrupt
ei3	NOOP	<not used>

Interrupt	Procedure	Usage
timer2	hw$isrTimer2	Square wave generation for powering PENEL & DSB2
timer1	isr$timer1	Mupus software clock, 1 millisecond resolution
timer0	isr$timer0	Hard (no multitasking) delay's and timeouts Routines: $USEC, $MSEC, TOUT$MSEC
ei2	isr$adc	Common-DPU ADC interrupt
rovw, povw, runw, punw	isr$stack	Multitasker: Stack underflow/overrun interrupt, kill liable task
ei1	isr$cdms	CDMS interrupt (highest priority)
nmi	NOOP	<not used> Non-maskable interrupt
noint	NOOP	<not used>

3.3 ADC channels

The following table lists all ADC channels available for the MUPUS instrument. From the software and connection point of view there are three types of ADC channels:

1. Channels directly connected to the Common-DPU multiplexer, Common-DPU ADC
2. Channels connected to the external MUPUS multiplexer (DSB10 board), cascaded to Common-DPU channel #13
3. PENEL temperature sensors measured by the PENEL-ADC, accessed via G-BUS

Table 5: Common-DPU ADC channels

MUPUS Channel number	ComDPU connector	MUPUS name	Description
0	INR1	TM0	Thermal mapper thermopile channel A
1	INR2	TM1	Thermal mapper thermopile channel B
2	INR3	TM2	Thermal mapper thermopile channel C
3	INR4	TM3	Thermal mapper thermopile channel D
4	INR5	TM4	Thermal mapper blackbody Pt-100
5	INR6	TM5	Thermal mapper reference channel A Pt-1000
6	INR7	TM6	Thermal mapper reference channel B Pt-1000
7	INR8	TM7	Thermal mapper reference channel C Pt-1000
8	INR9	TM8	Thermal mapper reference channel D Pt-1000
9	INR10	ANCM1	Anchor 1 accelerometer
10	INR11	ANCM2	Anchor 2 accelerometer
11	INR12	ANCT1	Anchor 1 temperature sensor Pt-100
12	INR13	ANCT2	Anchor 2 temperature sensor Pt-100
13	INR14	MUX	External MUX (DSB10 board), see Table 6
14	INR15	PENTS	PENEL temperature sensor Pt-100
15	<none>	DPU+5V	Common-DPU voltage +5V (on ComDPU board)

MUPUS external multiplexer (MUX) is located on the DSB10 board. It is addressed through the G-BUS port G0 (see Table 8) and is accessed through the Common-DPU ADC channel #13 (MUX).

Table 6: MUPUS external MUX channels

MUPUS Channel number	G0 address (see Table 8)	MUPUS name	Description
16	0x0000	SI-12	-12V current
17	0x1000	DS	Depth sensor
18	0x2000	SV-12	-12V voltage
19	0x3000	RES1	Thermal Mapper h/k-temperature sensor Pt-1000
20	0x4000	SI-5	-5V current
21	0x5000	RES2	Thermal Mapper 2.5Volt reference voltage
22	0x6000	SV-5	-5V voltage
24	0x8000	SI+5	+5V current
26	0xA000	SV+5	+5V voltage
28	0xC000	SI+12	+12V current
30	0xE000	SV+12	+12V voltage

PENEL temperature sensors are addressed by 11-bit addresses through G-BUS port G0. They are written using ADDA, SCLK, SDAT0 bits of the G0 port. The most significant address bit is written first.

PENEL sensor addresses are composed of a 5-bit channel number=1..24 referring to temperature sensors R1..R24 in the following way:

ChanNo=1..24	5 bits = b4,b3,b2,b1,b0	R1..R24
Sensor address	11 bits = 0,1,b0,b1,b2,b3,b4,1,1,0,0	MSB first

This leads to the following temperature sensor address table:

Table 7: PENEL temperature sensors

MUPUS Channel number	Address (hex, 11 bits)	MUPUS name(s)	Description
32	030C	R1	PEN sensor 1, d=4.57 mm* (top)
33	028C	R2	PEN sensor 2, d=14.01 mm
34	038C	R3	PEN sensor 3, d=24.39 mm
35	024C	R4	PEN sensor 4, d=36.04 mm
36	034C	R5	PEN sensor 5, d=48.70 mm
37	02CC	R6	PEN sensor 6, d=62.73 mm
38	03CC	R7	PEN sensor 7, d=78.33 mm
39	02CC	R8	PEN sensor 8, d=95.14 mm
40	032C	R9	PEN sensor 9, d=113.30 mm
41	02AC	R10	PEN sensor 10, d=133.27 mm
42	03AC	R11	PEN sensor 11, d=155.40 mm
43	026C	R12	PEN sensor 12, d=180.24 mm
44	036C	R13	PEN sensor 13, d=207.47 mm
45	02EC	R14	PEN sensor 14, d=236.45 mm
46	03EC	R15	PEN sensor 15, d=268.50 mm
47	021C	R16	PEN sensor 16, d=305.14 mm (bottom)
48	031C	HK1, R01	Short circuit/0 Ohm reference #1 for R1-R8
49	029C	HK2	PT100 holder inset (outside)**
50	039C	HK3	PT100 holder ring (outside)
51	025C	HK4, PENTS_ASSIST	PT100 detector, temperature control sensor inside PENEL
52	035C	HK5	TT100 holder inside
53	02DC	HK6, R100	100 Ohm reference resistor
54	03DC	HK7, R02	Short circuit/0 Ohm reference #2 for R9-R16
55	023C	HK8, R20	20 Ohm reference resistor

*d=distance to centre of sensor measured from top of sensor foil

**sensor inside holder for Flight Spare

3.4 MUPUS internal G-Bus resources

Three G-BUS ports are used for MUPUS instrument hardware: G0/G1 are write-only, G2 is read-only. MUPUS version 7.0 uses shadow variables, holding the last written G0/G1 values ($regStatG0/G1) and the last read G2 value ($regStatG2). These shadow values are written to HK data for analysis on ground.

Table 8: MUPUS G-BUS resources

	Bit	MUPUS name	Description
G0 port, write only			**Instruction: HEX 019 G!**
	0 (LSB)	PWRPENCBD	PENEL power: 1=on, 0=off
	1	PWRDB2	DSB2 board power: 1=on, 0=off
	2	PWRTM	Thermal Mapper power: 1=on, 0=off
	3	PWRANC1	First Anchor power: 1=on, 0=off
	4	PWRANC2	Second Anchor power: 1=on, 0=off
	5	ADDA	PENEL address/data, 0=send address, 1=send data
		ADDA=0	Write PENEL sensor/heater address using SCLK,SDAT0
			Switch off sensor heating
		ADDA=1	Perform ADC measurement (close MUX), Read ADC using SCLK,SDAT
			Write sensor heater energy using SCLK,SDAT0 Perform heating
	6	SCLK	Serial address/data clock
	7	SDAT0	Serial address/data sent from DPU
	8..10	CML1..3	Command bits for Thermal Mapper
	11	MUX	Internal MUX selection
	12	D	Depth sensor selection (see Table 6)
	13..15	A,B,C	External MUX address (see Table 6)
G1 port, write only			**Instruction: HEX 01A G!**
	0	PHO	Hammer action: 1=0n, 0=off
	1..2	DPO10..DPO11	Hammer energy level, 2 Bits: lo,hi
	3	EL1	Clamping electromagnets: 1=on, 0=0ff
	4	ELMB	Electromagnet current level
	5	MON	Stepper motor: 1=on, 0=0ff
	6	MDIR	Stepper motor direction: 1=backward, 0=forward
	7	MCLK	Motor clock pulse
	8..9	MM0..MM1	Motor energy level, 2 bits: lo,hi
	10	HEAT	PENEL heating: 1=on, 0=off
	11	HE	Thermal Mapper heating: 1=on, 0=off
	12	RES4	Deployment device release: 1=burn string
	13	DSR	Depth sensor release: 1=burn string
G2 port, read only			**Instruction: HEX 01C G@**
	0	MTL	Motor temperature level: 1=Motor is blocking
	1	MPS	Motor position sensor: 1=next motor rotation finished
	2	RES5	Sensor confirming Deployment device release: 1=released
	3	DPPO	1=Next hammer stroke is done
	4	DSS	Sensor confirming Depth sensor release: 1=released

	Bit	MUPUS name	Description
	7	SDAT	PENEL serial data input (PENEL temperature sensors)

4 Software Architecture

4.1 MUPUS software architecture

Architecture Overview

The figure below gives an overview on the general MUPUS software architecture. The RTX-2010 kernel, the multitasker and the basic I/O (BIOS) are ROLIS heritage, while the MUPUS software modules are instrument specific.

The main connection to the outer world is the CDMS interface. From the software point of view the MUPUS instrument can be thought as a machine receiving telecommands and producing telemetry:

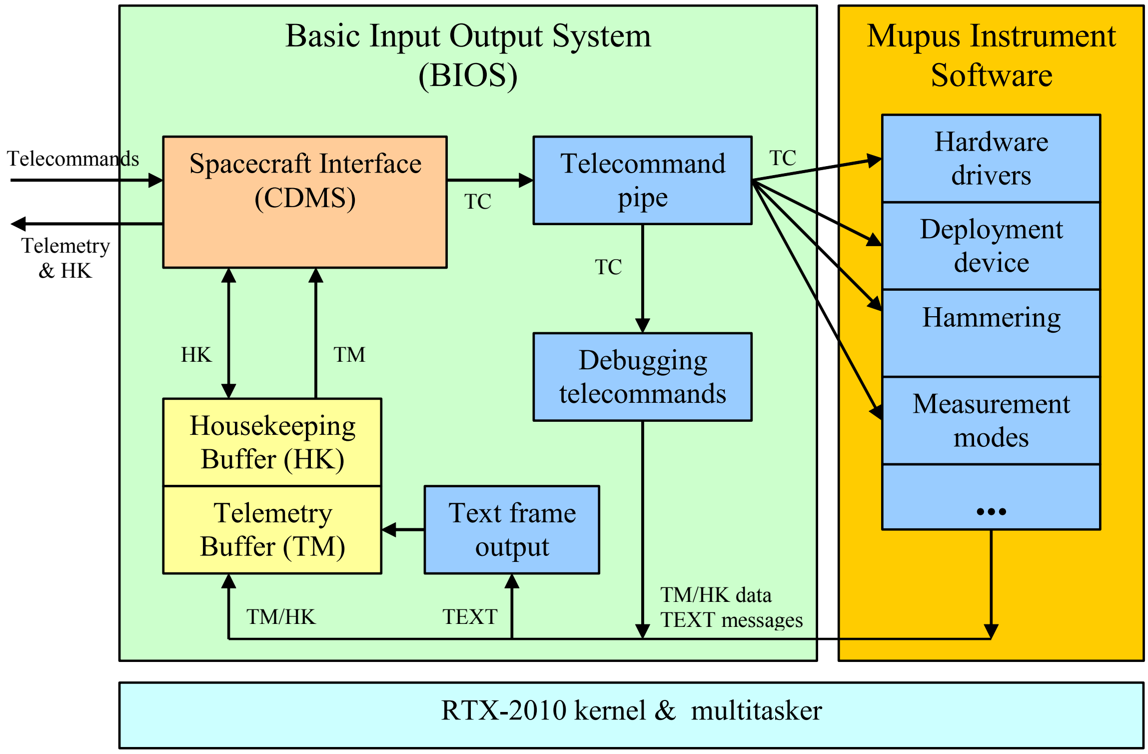


Figure 4-1: Onboard Software Overview

The Basic Input and Output System (BIOS) is application independent and provide a general communication layer (TC, TM, HK) with the outer world.

Housekeeping (HK) and telemetry data (TM) are buffered for CDMS output. The telecommand pipe buffers and executes incoming telecommands.

The instrument software modules provide drivers for the MUPUS hardware, performs instrument control and science data processing.

4.2 RTX kernel and Multitasker

RTX-2010 kernel

The RTX-2010 kernel is the same for all Common-DPU units and provides definitions for FORTH-83 standard words implemented for the RTX-2000/2010 target. For a description of these words refer to the LMI UR/FORTH manual [3].

RTX cooperative multitasker

MUPUS version 7.0 uses a cooperative multitasker based on a standard Forth-83 ***PAUSE*** scheduling supplemented by ***sleep*** and ***timeout*** functions. Dividing the MUPUS software into different tasks makes it easier to keep control of complex situations, e.g. Execute a MUPUS measurement mode while buffering received telecommands, sending science data frames and controlling the PENEL temperature in background.

The following rules apply to the multitasking implementation:

- The RTX-2000/2010 data and return stacks (256 words each) are divided into four data and four return stacks (64 words each), thus allowing a maximum number of four tasks.
- The stacks should be used carefully, because interrupt services use the stack of the currently running task. The maximum necessary data stack capacity is +9 words, for the return stack +8 words.
- No automatic task switching, e.g. driven by hardware timer.
- Every task decides by its own, when to perform task switching by *"manual round-robin"*, observing some typical latency of maximum ~10 milliseconds (TBD).
- On task switching the current RTX stack pointers (SPR) and configuration register (CR) are saved / restored to / from the stack of their own task. To reduce the task switching overhead, no other processor resources (DPR, multiplier registers, etc.) or hardware resources (G-BUS status, MUX channels, etc.) are saved and restored during task switching.
- A consequence is that waiting for ADC measurements should not be performed using multitasking timeout (function: ***timeout***), but using ***TOUT$MSEC*** (timer0, see 3.2).

Stack interrupts

The RTX-2010 processor supports stack interrupts, initiated on stack overflow or underflow, when a ***push*** or ***pop*** operation crosses a limit of the current stack region. The RTX-2000 processor used for the laboratory model doesn't support stack interrupts.

A stack overflow or underflow has to be considered as a serious software crash. That can happen during software development, but should not happen at flight. Therefore a very simple mechanism is used for this case:

- If a stack interrupt occurs, then the current "liable" task is killed.
- The MAIN task (task#0) regularly checks for a killed task and restarts it if necessary, trying to notify about the failure by sending a telemetry TEXT frame.
- The MAIN task itself cannot be restarted, it should be the simplest task doing *idle operations* (e.g. PENEL heating).
- A killed task will be restarted from scratch, loosing the complete context. Stopped operations cannot be continued.

Task switching

Task switching is performed by three basic procedures:

PAUSE (--)

- A standard FORTH-83 word used for IDLE-tasks.
- Delay the current task for a manual task switching (round-robin).
- All tasks are restored and checked for continuation in a fixed order:

  Task#0 → Task#1→ Task#2→ Task#3

- When the round-robin is complete the current task immediately continuous.

timeout? (cond? msec -- flag)

- Cond? .is the address of a procedure returning a flag.
- Return immediately (without task switching), if the condition is TRUE.
- Otherwise remember the current MUPUS time (see 0) on task stack.
- Delay the current task (***PAUSE***) until a condition becomes TRUE.
- Timeout after the specified number of milliseconds has gone.
- Return TRUE if timeout, otherwise return FALSE.

sleep (msec --)

- Return to current task after a number of milliseconds.
- Implemented by calling ***timeout?*** with a FALSE condition, ignoring the result:

: sleep (msec --) ['] FALSE SWAP timeout? DROP ;

4.3 Basic I/O system

Spacecraft interface (CDMS)

The low-level interface based on CDMS interrupts (RTX external interrupt ei1) is part of the Common-DPU BIOS and provides callback functions for different CDMS message types: Status, Command, Data and Broadcast messages. A detailed description of the CDMS callback mechanism can be found in the "Common-DPU User Manual" [2].

High-Level spacecraft interface is implemented as a jump table for all CDMS action codes, branching to a set of handlers one for each of the implemented CDMS action codes. The following tables give an overview of the CDMS action codes, request codes and status flag served by the MUPUS software version 7.0. For a complete description of CDMS services, refer to [4].

Table 9: CDMS Action Codes serviced by MUPUS

Action Code		Supported ?	MUPUS function
TRSW	Transmit status word	yes	Status word transmitted by Common-DPU hardware
TRQC	Trm Serv Request Code	yes	Service request machine: Send Request Code word; - Mark a single-step service request finished, or - Wait for the next step of multi-step SR's (BRAM)
STBY	Standby / Power down	no	Ignored
RMOD	Rcv Current CDMS Mode	yes	Store CDMS mode to HK variable **Hk.Cdms.RMOD**
RTIM	Rcv Onboard Time	yes	Store the current Lander time together with synchronized MUPUS instrument time to HK (see 0)
RSST	Rcv Service System Status	yes	Not used ???; Store Service system Status
RAXT	Rcv Action Code Extension	no	Ignored
RHFM	Rcv HK Data Format Count	yes	Prepare next HK channel (modulo 64) value for transmission;
THKD	Trm HK Data word	yes	Transmit the HK data value; Initiate a new HK measurement cycle after transmitting channel 63.
RCMD	Rcv Telecommand Sequence	yes	Store the received telecommand to a buffer; CDMS task then moves this telecommand to the telecommand pipe for execution.
TCMO	Trm Ofs/Len StoredTcmd	yes	Not used in MUPUS version 7.0ff
RCMS	Rcv Stored Tcmd Buf	yes	Not used in MUPUS version 7.0ff
RASV	Rcv Alloc Science Data Vol	yes	Not used; Store the allocated data volume
TSCR	Trm Science data burst	yes	Transmit 4/32 data words from current telemetry frame output buffer after sending SRDY request; otherwise transmit zeros
RSCS	Rcv Science Data Pack Checksum	yes	MUPUS science data frames contain a checksum so that RSCS should always be = FFFF; otherwise increment error counter in HK

Action Code		Supported ?	MUPUS function
RBUS	Rcv Alloc BackRAM Buf Size	yes	Not used; Store the Backup RAM buffer size
TBUP	Trm Pointer Backup RAM	yes	Transmit the Backup RAM pointer after WRBF request; otherwise transmit zeros; Mark the SR finished (next SR can be started)
TBUF	Trm BackupRAM record	yes	Transmit the Backup RAM record after WRBF request; otherwise transmit zeros; Mark the BRAM cycle finished.
RBUF	Rcv BackupRAM Buf Record	Yes	Store the Backup RAM record; Mark the BRAM cycle finished.
TTRG	Trm Trigger word	no	TrsTrigWord/1
RTRG	Rcv Trigger Word	no	RecTrigWord/1
RERC	Rcv Error Code Word	yes	Increment error counter in HK; Drop or repeat the service request (see)

The Service Request (SR) machine is implemented on top of the callback mechanism of the CDMS interface. High-level user routines may initiate any supported SR and optionally wait for the SR completion using multitasking timeouts. The SR machine supports SR buffering, error repetition and SR timeouts with the following features and conditions:

- Maximum 8 different SR's can be queued.
- Only one SR of the same type (request code) can be buffered at the same time.
- If a timeout value is specified and the SR is not finished with the specified number of milliseconds then the same SR can optionally be retried.
- If the CDMS responds with an error message (RERC) then the same SR can optionally be retried.
- If an error condition occurs (timeout or RERC) and a retry value is specified, then the same SR will be retried (appended to the SR queue). If no retry value is specified or the number of retries is exhausted then the SR is dropped.

The following table lists the SR code used by MUPUS version 7.0 together with their error handling.

Table 10: CDMS Service Request Codes used by MUPUS

Request Code		Supported ?	Timout ?	Retry ?	Description
SSST	Send service system Status	yes	0	0	Issued once at MUPUS startup, Value not used ???
SCMD	Send stored Tcmd buffer section	yes	3 sec	0	Not used in MUPUS version 7.0
SASV	Send allocated science data volume	yes	0	0	Issued once at MUPUS startup, Value not used ???
SRDY	Science data ready	yes	3 sec	forever	Issued if there is buffered telemetry data available.
SBUS	Send allocated backup RAM buffer size	yes	0	0	Not used in MUPUS version 7.0
WRBF	Write backup RAM buffer record	yes	3 sec	0	Used in several MUPUS modes
RDBF	Read backup RAM buffer record	yes	3 sec	0	Used in several MUPUS modes
STRG	Pass trigger word	no			Not implemented in MUPUS version 7.0
FLSP	Flush last science data packet	yes	0	0	Issued once before sending each science data frame.
OPCL	Operation completed	yes	0	0	Used by telecommand WaitDataComplete

The following table lists the CDMS status word bits and their usage in MUPUS version 7.0.

Table 11: CDMS status word flags supported by MUPUS

Status flag		Supported ?	Buffer/Size(word)
BSY	BUSY flag	no	
CE	Count Error flag	yes	Supported by Com-DPU BIOS: Issued if the number of received command word does not match the message WRDC field.
ME	Message Error flag	no	Supported by Com-DPU BIOS; Not used in MUPUS version 7.0
SR	Service Request flag	yes	Supported by Com-DPU BIOS: Issued, when a SR request is pending.
SM	Sleep Mode flag	yes	Supported by Com-DPU BIOS: Common-DPU is switched to low power mode; Not used in MUPUS version 7.0

MUPUS instrument time

MUPUS uses an internal software clock. It is not a real-time clock, but operates at the basis of the processor clock. It counts the time since MUPUS boot with a resolution of 1 millisecond, much better than the CDMS Lander onboard time (LOBT). The MUPUS time value is a 32-bit integer value overrunning after 49.7 days (2^{32} milliseconds).

☝ ***MUPUS science measurements always refer to the MUPUS with 1 millisecond resolution.***

☝ ***Each MUPUS HK data record contains the last received LOBT together with the corresponding MUPUS time to be used for time synchronisation on ground.***

Telecommand pipe

The telecommand pipe is composed of

- A 1 Kbytes ring-buffer storing incoming telecommands;
- A telecommand interpreter using a lookup-table
- A separate task executing MUPUS non-mode telecommands.

Multitasking aspects of the telecommand pipe are discussed in section 0.

Text frame output

The most simple science data frame format is TEXT. Beside a frame header those frames are composed of human readable plain ASCII characters. TEXT frames are very useful for debugging during onboard software development as well as during flight, e.g. for sending a simple notification that the Deployment Device has successfully been released.

However TEXT frame output differs is bit from ROLIS heritage:

TEXT frame output	Rationale
ROLIS TEXT frames are sent directly bypassing frame buffering	ROLIS/CIVA may have megabytes of image data buffered for output, while still continuing imaging. Bypassing the frame buffer makes TEXT messages being visible “in real-time”.
MUPUS TEXT frames are buffered as normal science data frames	MUPUS does not produce so much data. On the other hand during MUPUS motor control multitasking needs to be disabled, preventing direct TEXT output.

Telemetry frame output

MUPUS science data frames including text frames are usually buffered in a 255 frame FIFO buffer (64 kBytes). This buffer might fill up only if the CDMS does not serve Mupus telemetry. To avoid stop of MUPUS operation (e.g. endless heating could damage hardware !!!) software version 70 rejects produced science data frames if the telemetry output buffer is full. As this would be a non-nominal case, a housekeeping counter (RejectedFrames) reflects that fact.

Error handler

A very simple error handler has been implemented on top of TEXT frame output:

- There are 16 different error codes=0..15
- Each of the error codes has a separate byte counter in housekeeping data (MupusErrors, HK#56-63).
- A simple routine ***mupError(code n --)*** increments the error counter and send an error message via a TEXT frame. In order to avoid flooding the telemetry with the same error message again and again, only the first N occurrences of the same error code are send (typical N=3).

The following table shows the implemented error codes:

Table 12: Mupus error codes

Number	Name	Description
0	MUP_ERR_TCMD	Invalid telecommand received
1	MUP_ERR_HAM_HW	Hammer hardware does not respond
2	MUP_ERR_ADC	ComDPU ADC timeout
3	MUP_ERR_ADC_EXT	External (DSB2) ADC timeout
4	MUP_ERR_4	Unused
15	MUP_ERR_15	Unused

4.4 MUPUS tasks

The cooperative multitasker used for MUPUS version 7.0 (see 0) is configured to run four independent tasks. Each task has its own DATA and RETURN stack (64 words each).

☝ ***Note that interrupt services are using the stack of the currently active task with a maximum (CDMS-ISR) required stack depth for DATA/RETURN stack of 9/8 words respectively.***

The following figure gives an overview of the different MUPUS tasks:

MUPUS tasks

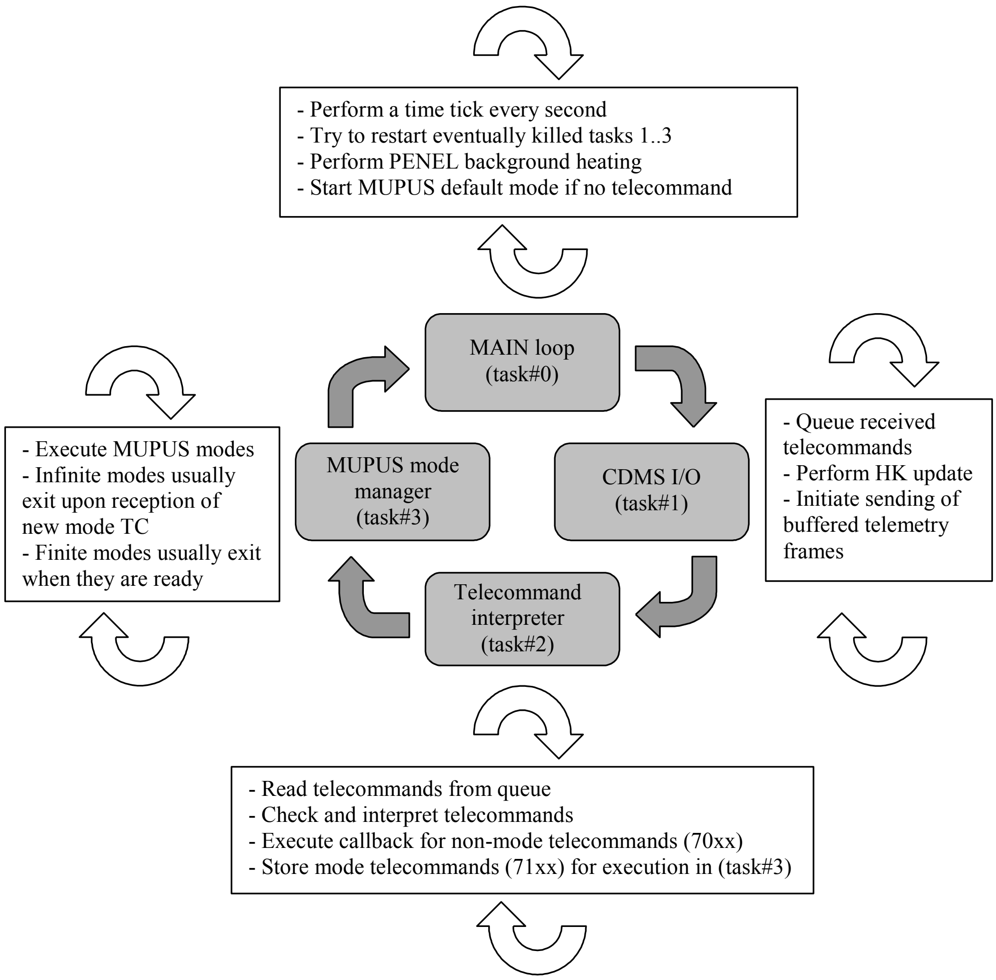


Figure 4-2: Mupus tasks overview

MAIN loop (task#0)

The MAIN task is the simplest of all MUPUS task. It performs the following actions:

- Sleep 10 seconds (TBD).
- Check it there is a task killed by stack failure, restart it.
- Perform other idle operations in background:
  - Check PENEL heating if PENEL is powered, by measuring
  - – to be continued –

CDMS input and output (task#1)

CDMS message handling is performed in real-time, by interrupt services. But there is only a limited buffer for incoming CDMS messages (including telecommand), a single telemetry frame (128 words) output buffer and a single housekeeping block.

The CDMS task is responsive for accepting incoming telecommand and for feeding telemetry and housekeeping buffers with fresh data at regular intervals:

- If a telecommand has been received, move it to the telecommand pipe (task#2) capable of storing more telecommands until they are executed.

- If there is a telemetry frame in the global frame buffer available, then move it to the CDMS telemetry output buffer and initiate the data output by issuing FLSP, SRDY service requests (see 0).
- If the last housekeeping value of a block has been sent (see 9.1), then perform a housekeeping update including ADC measurements and move the completed housekeeping block as a system snapshot to the CDMS HK output buffer.

Telecommand interpreter (task#2)

Incoming telecommands are moved to the telecommand pipe, a 1 Kbytes (TBC) ring-buffer storing telecommands for execution. The telecommand interpreter is responsive for reading and interpreting telecommands from the pipe and for executing them. The telecommand execution time varies for different telecommands. E.g. a patching telecommand copies a few data words to memory returning after a few microseconds, while burning the EEPROM may take seconds, or a large memory dump may stall until there is some place in the frame output buffer.

The telecommand interpreter

- Reads a telecommand;
- Interprets it by searching in a telecommand lookup-table;
- Executes the registered telecommand callback.

However there is one exclusion – **MUPUS mode telecommands**. A MUPUS mode is a procedure running for a long time (e.g. hammering or deployment) or even running forever (e.g. LONGTERM mode). Running these procedures from the telecommand interpreter task#2 would block all other incoming telecommands until the mode is finished. Therefore mode telecommands are stored into a global variable and marked for late execution in the MUPUS task#3 (see 0). This approach frees the telecommand interpreter immediately and gives an eventually currently running mode some time e.g. to exit by finishing the current measurement cycle.

MUPUS mode manager (task#4)

The main MUPUS tasks (hammering, deployment, longterm measurements) are performed as MUPUS modes independently from the execution of other MUPUS telecommands. This approach allows to patch variables or to dump memory, while a MUPUS mode is running.

If a new MUPUS mode telecommand is received it is marked for late execution. The currently running MUPUS mode may check for a new mode and decide to exit for executing (e.g. LONGTERM measurement) or ignore new modes (e.g. during hammering or deployment).

- ***If a MUPUS mode procedure is running, then only one new mode can be marked for later execution, another MUPUS mode telecommand would override the previous one.***
- ***Every MUPUS mode may decide whether to exit for new a mode or whether to ignore it.***

5 MUPUS instrument software

MUPUS instrument software is mainly a set of independent working modes (hammering, deployment, measurement modes) supplemented by a set of tools for powering MUPUS components on/off, instrument configuration and debugging aids.

This part of onboard software has mostly been rewritten, with one exception:

☝ ***The "mechanical" part of the onboard software (hammering deployment) has been adopted the new onboard software kernel, but keeping the functionality unchanged.***

5.1 MUPUS configuration (EEPROM instead of STCB)

To avoid conflicts with the coexisting software version 4.6b/6.1, MUPUS version 7.0 does not make use of the CDMS stored telecommand buffer (STCB) for instrument configuration. Version 7.0 even does not use CDMS Backup-RAM for storing configuration to avoid duplications and conflicts.

For a better transparency the complete MUPUS configuration is kept in a single place (EEPROM starting at address xAD00). A small set of telecommands allow changing, saving and restoring configuration.

From the software point of view, the MUPUS configuration is a block of variables initialized with default values (default configuration). At MUPUS software start-up the configuration is restored from EEPROM overwriting the default configuration. The current instrument configuration can be changed by telecommand at any time. Optionally the current configuration can be saved to EEPROM to be used for future MUPUS operations.

During software development the contents of the configuration block changes frequently. To avoid random misconfiguration the EEPROM configuration is checksum protected. Additionally a timestamp is added during software compilation to avoid misconfiguration due to software changes. When the timestamp of the stored EEPROM configuration does not match the onboard software timestamp, then the EEPROM configuration is not used.

☝ ***The EEPROM configuration is checksum protected.***

☝ ***Recompilation of the onboard software resets the EEPROM configuration.***

The following table shows the configuration block of MUPUS software version 7.0 with default values and memory offsets needed for the configuration telecommand:

Table 13: MUPUS configuration

Word Nº	Name	Default	Range	Description
General software configuration				
0	config	0xC1C0		Software flags (bit masks) 0x8000 Text frames enabled 0x4000 EGSE=1: Flush text frame at /TXT FLIGHT=0: Collect data until frame is full 0x2000 Direct output of text frames; 0: Buffered output 0x0300 0..3: Diag level via TXT-frames 0x0080 Enable heating output for WARM-UP heating 0x0040 Enable heating output for CONTINUOUS heating phase 0x0020 Enable heating output for DSB/ENB heating (disable/enable

Word Nº	Name	Default	Range	Description
				heating information during measurements, only for debugging)
1	framePage	1	=0..3	Memory page used for science data frame buffer =0: Use code page, buffer max. 63 telemetry frames =1..3: Use upper memory page, max. 255 telemetry frames
2	dataPage1, dataPage2	0x0203		Two different data pages (≠framePage): dataPage1: Used for fast storage of raw ADC measurements and for ANCHOR mode (main anchor) dataPage2: Used for ANCHOR mode (redundant anchor)
3	ancChan1, ancChan2	0x090a		ADC channels for ANCHOR measurement mode: ancChan1: Main anchor ADC channel ancChan2: Redundant anchor ADC channel
Heating section				
4	Heat.PenOnLimit	352 (-40°C)	(signed)	PENEL power-on temperature limit; Measured with PENTS sensor
5	Heat.PenLowLimit	8100 -48.2°C	(unsigned)	Lower limit of PENEL temperature in 1/100 Ohms (Pt-100) when powered ON Measured with PENTS_ASSIST HK4 sensor
6	Heat.PenHiLimit	8230 -45.7°C	(unsigned)	Upper limit of PENEL temperature in 1/100 Ohms (Pt-100) when powered ON Measured with PENTS_ASSIST (HK4) sensor
7	Heat.PenCheck	10	=0..65535	Penel temperature check interval when PENEL is powered ON (PENTS_ASSIST, HK4 sensor) =0: Don't check PENEL temperature >0: Interval in seconds
8	Heat.TmOnLimit	-2641 (-90°C)	(signed)	MAPPER power-on temperature limit; Measured with RES1 sensor (Pt-1000)
9	Heat.TmCalLimit	-4697 (-110°C)	(signed)	MAPPER calibration temperature limit; Measured with TM4 sensor (Blackbody Pt-100) **Should NOT BE decreased because of danger to destroy Pt-100 heater by inrush current !**
10	Heat.OnTime0	10	(unsigned)	PENEL & MAPPER pre-heating loop: Starting Heat=ON time in milliseconds
11	Heat.OffTime0	990	(unsigned)	Starting Heat=OFF time in milliseconds
12	Heat.IncrTime	10	(unsigned)	Time increment in milliseconds
13	Heat.MaxLoops	6000 (100 min)	(unsigned)	Max. pre-heating loops (1 sec for 10+990msec)
14	Heat.DataFlag (version ≥7.2 only)	1	(unsigned)	Heating TM data production flag: 0= No data; >0= Send every N'th value Note interaction with config (word#0) flags !

Word Nº	Name	Default	Range	Description
15	Heat.Spare	0		<unused> 2 words
ADC measurement section				
16	ADC.Delay.Timer2	100	(unsigned)	>0: Microseconds wait after Timer2-square wave before ADC measurement
17	ADC.Delay.Mux	40	(unsigned)	>0: Microseconds wait after setting up MUX address
18	ADC.Avrg.Num	256	=1..1000	Number of measurements to average, for all measurements calling hwAdcMean (V7.2, only used for ANC-T measurements after accelerometer sampling in Anchor mode) (V7.3, used for measurement of PENEL THC heater current !)
19	ADC.Avrg.Mapper	160	=1..TBD	Averaging of MAPPER & ANCHOR temperature measurements (separate parameter)
20	Pen.Delay.Timer2	0	(unsigned)	>0: Microseconds wait after Timer2-square wave before ADC measurement
21	Pen.Delay.MuxAddr	0 (v7.1) 256 (v7.2)	(unsigned)	>0: Microseconds wait after setting up MUX address, Should always be > 255 µs ! Delays < 256 µs could cause disturbances in PENTS_ASSIST readings during execution of THC mode with high power settings
22	Pen.Delay.Mux	64	(unsigned)	>0: Microseconds to wait after 1st ADDA pulse (close MUX, settle MUX); **MUST BE > 0 for reliable resistance measurements !** =0: Don't set ADDA=low,high (method used up to Version 46)
23	Pen.Delay.Adc	30	(unsigned)	Microseconds to wait after 2nd ADDA pulse (PENEL A/D conversion time, should not be changed !)
24	Pen.Delay.Meas	3	(unsigned)	>0: Milliseconds delay between two consecutive Penel measurements
25	Pen.Avrg.Num	10	1..TBD	Number of values used to average for Penel measurements (TEM,THC,Longterm)
26	Pen.Dummy.Num (version ≥7.2 only)	0	(unsigned)	Number of dummy measurements of all 24 PENEL channels before averaging (0=none)
27	Pen.Spare	0		<unused> word
ANCHOR mode section				
28	ANC.PreTriggerLimit	476	1..30000	Number of accelerator measurements to buffer before triggering anchor-fire Changed with version V7.2 from originally 3000 (until V7.1) => Trigger point = first sample in ANC frame 5
29	ANC. Wait2ndAnchor	0	0,-1	TRUE: Wait until 2 Anchors fired, even after ***safe anchoring*** from Anc1. Exit after *unsafe*

Word Nº	Name	Default	Range	Description
				anchoring (CDMS SST) FALSE: Exit only if Safe anchoring (CDMS SST, introduced change for version 7.3 ff, for earlier versions also the *unsafe anchoring* flag in the SST causes MUPUS to exit Anchor Mode)
30	ANC.Spare	0,0		<unused> 2 words
HAMMER/ARM insert mode section (hammer.4th version 57 heritage)				
32	INS.PDIn	0x0600	(unsigned)	Parameter description table (not used in V7.x)
33	INS.#Puls	0x0306	(unsigned)	Number of DD pulses during deployment
34	INS.#PulB	0x02EE	(unsigned)	Number of DD pulses during retracking
35	INS.Fr0	0x00C8	(unsigned)	Double initial frequency of DD motor in Hz
36	INS.Fr1	0x0960	(unsigned)	Double final frequency of DD motor in Hz
37	INS.FrR	0x00C8	(unsigned)	Default Deploy Freq in Res Mode [256 Hz] (TC-parameter)
38	INS.CMR	0x0300	(unsigned)	Default Deploy/Retract CMS in Res Mode (TC-parameter)
39	INS.Fr0VT	0x00C8	(unsigned)	Default Deploy Freq in Thermal-Vacuum mode TC-parameter)
40	INS.Fr1VT	0x0960	(unsigned)	Default Deploy Freq in Thermal-Vacuum mode (TC-parameter)
41	INS.#PulsVT	0x0050	(unsigned)	Default Deploy/Retract CMS in Res Mode (TC-parameter)
42	INS.ELR	0x0000	(unsigned)	Reserve mode HAMMER energy (changed for V7.3, use lowest energy level per default instead of level 2)
43	INS.#4StrokeL	0x0064	(unsigned)	Reserve mode default number of HAMMER 4-strokes
44	INS.DepthOff	336	(unsigned)	`CONSTANT Depth offset`
45	INS.SafL	200	(unsigned)	Max. number of 4-Strokes for each energy level=200
46	INS.SafTab	 2, 4, 8, 10	(unsigned)	Limit the number of hammer cycles, then switch from energy level 0 to level 1 switch from energy level 1 to level 2 switch from energy level 2 to level 3 Switch from energy level 3 to STOP (only if no progress was made with last stroke !)
50	INS.DV250	128	(unsigned)	Insertion depth for retracting (V7.4*)
51	INS.DV495	37	(unsigned)	Maximum insertion depth (V7.4)
52	INS.DV007	5	(unsigned)	minimum depth increment between 2 strokes (if progress less => increase energy level, V7.4)
53	INS.DVLow	20	(unsigned)	minimum depth for Lander lowering (not effective), V7.4
54a	INS.ArmBlockChkMode	1	(byte)	Mode for checking blocked deployment: 0=none / 1=check & exit / 2=check & retract (V7.4)
54b	INS.ArmBlockChkRo	5	(byte)	Number of rotations expected in check interval

Word Nº	Name	Default	Range	Description
	t			(def. 5 rot), V7.4
55a	INS.ArmBlockChkMin	3	(byte)	Minimum rotations in check interval, otherwise DD is considered as blocked (V7.4)
55b	INS.ArmBlockRetract	25	(byte)	Number of rotations to retract blocked Arm if INS.ArmBlockChkMode=2 (V7.4)
56	INS_ArmMinDeploy	125	(unsigned)	Minimum number of rotations needed to be away from Balcony (to consider deployment as successful), V7.4
Longterm TEM/THC/MAPPER section				
60	LT.Default	600 (10 min)	(unsigned)	Run default longterm after N seconds
61	LT.Tick	100	(unsigned)	Tick interval in msec / Heating time of a single sensor
62	LT.MapperMode	1	=0..2	Mapper mode 0=off, 1=normal power, 2=low power
63	LT.IdxHeat	-1	=-1..31	Starting heater index: -1: IdxHeat/NumHeat from BRAM; >=0: Reconfiguration: Don't use BRAM
64	LT.NumHeat	1	=1..16	Number of sensors to heat quasi-simultaneously
65	LT.THC.Interval	88	(unsigned)	THC (active measurement) interval in TICKS, normally a multiple of NumHeat (Note: Measurement time added to interval, ref. 0) => 1 scan every 10 sec
66	LT.THC.Num	200	(unsigned)	Number of THC measurement records (Longterm active THC phase)
67	LT.TEM.Interval	288	(unsigned)	TEM (passive measurement) interval in TICKS (Note: Measurement time added to interval, ref. 0) => 1 scan every 30 sec
68	LT.TEM.Num	200	(unsigned)	Number of TEM measurement records (Longterm passive TEM phase)
69	LT.MAP.Interval	300	(unsigned)	Default MAPPER interval in TICKS, default longterm if not deployed (Note: Measurement time added to interval, ref. 0)
70	LT. ANC.AlwaysOn	0	(unsigned)	Anchor 1,2 powering during longterm mapper/anchor measurements: AlwaysOn: Flag whether both anchors should be continuously powered
71	LT.ANC.PwrOnDelay	50	(unsigned)	Milliseconds delay after powering both anchors on before ANC-T temperature reading is taken
Longterm THC sensor heating				
72	LT.HeatTab	0x0087, 0x008F, 0x0083, 0x008B, 0x0086, 0x008E, 0x0082,	(unsigned)	heater current applied 000:FFF, No of heated sensor 0:F (low byte**), heating current should NOT be changed during cruise phase**

Word Nº	Name	Default	Range	Description
		0x008A, 0x0085, 0x008D, 0x0081, 0x0089, 0x0084, 0x008C, 0x0080, 0x0088,		
88		0x0000	=0	End of heating table
89	LT.THC.HeatPause (version ≥7.3 only)	0	(unsigned)	Number of ticks to wait after heating each separate sensor (reduces the overall heating power, introduced with V7.3)
90	LT.Spare-2	0,0		<unused> 2 words
Landing gear mode				
92	LG.Height	170	(unsigned)	Minimum LG height in millimetres for deployment, Value changed with V7.4 to cope with new LUM 5-1 values
93	LG.Tilt1	5	(unsigned)	Max. absolute LG tilt-1 allowed for deployment in [deg]
94	LG.Tilt2	5	(unsigned)	Max. absolute LG tilt-2 allowed for deployment in [deg]
95	LG.Spare	0,0		<unused> 2 words
97	LG. AzimuthExclude	28,56, 145,178, 268,296, 0,0	(unsigned)	Excluded deployment regions, from<=degree<=to 0,0 ➔ end of table Default: exclude azimuths of Lander legs (max. total 22 words) Values changed with V7.4 to cope with new LUM 5-1 values
119	chksum		(unsigned)	Config checksum (16-bit sum of all config bytes)

*parameters marked by V7.4 were introduced with s/w version V7.4 and are meaningless (but harmless since these values were not used) for earlier s/w versions !

5.2 MUPUS Backup-RAM

MUPUS version 7.0 uses CDMS Backup-RAM for instrument status and for data exchange with other instruments (SESAME/PP and Landing GEAR).

MUPUS version 7.0 instrument status is saved/restored to/from Backup-RAM address 3800(hex) to avoid conflicts with MUPUS version 4.6b/6.1 status stored at address 3804(hex).

Obviously the existing Backup-RAM structures for data exchange with SESAME/PP and Landing GEAR should be kept compatible between MUPUS versions 4.6b/6.1 and 7.0, unless there are other agreements with the SESAME/PP or Landing GEAR teams.

Table 14: Backup RAM (address 3800), MUPUS instrument status

The instrument status is saved whenever a MUPUS mode is finished. Additionally it is saved in LONGTERM mode after each THC/TEM cycle.

Word Nº	Description	Value	Default
0,1	MUPUS SW compilation date (fixed) compilation month,day	Year=2006 256*month+day	

Word Nº	Description	Value	Default
2,3	MUPUS SW compilation time (fixed)	256*hour+min 256*sec+sec100	
4	MUPUS software version	0x0700	
5	MUPUS status flags, copy of HK value MUPHK31		
6	LONGTERM mode heating table index (-1➔ use CONFIG)	=-1,0…16	-1
7	LONGTERM mode: Number of heaters (-1➔ use CONFIG)	=-1,0…16	-1
8-31	NA	0	0

Table 15: Backup RAM (address 3805), designated to data exchange with SESAME/PP.

Word Nº	Description	Value (hex)	Remarks
0A	PEN deployed\stowed	FF\00	FF=PENEL is deployed (high byte)
0B	HAMMER active\silent	FF\00	FF=Hammering in progress (low byte)
1A	PEN electronics ON\OFF	FF\00	FF=PENEL powered
1B	Hammer stroke level	00:03	Hammer stroke energy level
2	PEN insertion depth	0000…FFFF	PEN depth sensor measurement after performing four hammer strokes and releasing the electromagnetic clamps
3	Record number	0000..FFFF	Counter of hammering records (4 strokes each).
4	LOBT, low	0000…FFFF	Last received Lander onboard time from CDMS, low word
5	LOBT, medium	0000…FFFF	*Ditto, medium word*
6	PEN reference depth	0000…FFFF	PEN depth measured before hammering (same value for all measurement records)
7	**Mupus SW version**	**0000\0700…07FF**	**Words 3,8-20 are valid only for SW version >=0700(hex)**
8A	DS released	FF\00	FF=Depth sensor (penetrator) mechanically released from the deployment mechanism (high byte)
8B	N.A.	00	Spare (low byte)
9	Mupus LOBT-Reference, high	0000…FFFF	Mupus reference time (milliseconds since boot), when the last LOBT (words 4,5) has been received. Useful for time synchronization with LOBT.
10	Mupus LOBT-Reference, low	0000.FFFF	*Ditto, low word*
11	Hammer stroke start time, high	0000.FFFF	Mupus time (milliseconds since boot), when current hammering 4-stroke has been started. Words 13-16 are measured relative to this time.
12	Hammer stroke start time, low	0000.FFFF	*Ditto, low word*
13	Time of 1st hammer stroke	0000.FFFF	Millisecond difference to hammer stroke start time (words 11,12)
14	Time of 2nd hammer stroke	0000.FFFF	*Ditto*
15	Time of 3rd hammer stroke	0000.FFFF	*Ditto*
16	Time of 4th hammer stroke	0000.FFFF	*Ditto*
17-30	N.A.	0000	Spare
31	MUPUS BRAM properly installed	BDEF	

Table 16: Backup RAM (address 8000), reading data from Landing GEAR

Word Nº	Description	Digital value	Scaling applied	Physical value
HK#13	Motor 1 position (Height)	8192…15019	{ 2500*(X-8192)/6827 + 115 } / 10	0…220 mm
HK#15	Motor 3 position (Tilt1)	8192…15019	{ 803*(X-8192)/6827 - 394 } / 10	-30…+30°
HK#16	Motor 4 position (Tilt4)	8192…15019	{ 788*(X-8192)/6827 - 388 } / 10	-30…+30°
HK#17	Motor 5 position (Azimuth)	8192…15019	{ 3814*(X-8192)/6827 - 53 } / 10	-5…+376°

5.3 MUPUS heating procedures

Two MUPUS hardware modules have a minimum required switch-on temperature: the penetrator electronics (PENEL, ~-40°C) and the thermal mapper (TM, ~-90°C). They need heating before power-on. Furthermore when powered PENEL needs to kept in working temperature range (~-40..-50°C). Temperature thresholds are defined in the MUPUS configuration (see 5.1) and can be changed during flight.

PENEL heating

The PENEL heating procedure is different for two phases: the warm-up phase before power-on and the continuous phase preserving the working temperature after power-on.

PENEL warm-up phase

During PENEL warm-up the PENTS sensor is used for temperature measurements. The PENTS temperature is checked in intervals of ~1 second. Measured PENTS temperature values are sent as heating telemetry frames (type=0x71, see 0). To avoid temperature shocks of the PEN electronics, heating starts slowly, by switching PENEL heating on for only 10 milliseconds, and then switching it off for 990 milliseconds. At the every next step the heating time is incremented by 10 milliseconds and the off-time is decremented by the same value.

When the PENTS sensor reaches the configured power-on temperature level, then the PEN electronics is powered on and heating is continued in the next phase.

PENEL warm-up heating is performed as a MUPUS mode from the power-on telecommand (see 0) or from other MUPUS modes using the PEN electronics.

PENEL continuous heating phase

After PENEL has been switched on, the PENEL temperature is measured by the PENTS_ASSIST (HK4) sensor. Continuous heating is performed to keep the PENEL temperature in the configured working temperature range. PENEL heating is switched on, if the measured value skips below the lower limit, and switched off, if it skips above the upper limit of the working temperature range.

The current PENEL temperature is included in PENEL-housekeeping data (HK4). The HK4 values are stored in the general heating frame type 0x71, where each record includes an individual sensor ID to distinguish between warm-up heating based on PENTS and continuous heating based on HK4 (see 0) .

Continuous heating is performed in procedure *pwrPenHeatCheck* as a background job from the MAIN MUPUS task#0.

Background heating distorts PENEL temperature measurement, it is temporarily disabled during those measurements by calling procedures *pwrPenHeatDisable/Enable*. These procedure should be called in pairs, they save and restore the current heating status. Calling *pwrPenHeatCheck* in the disabled phase waits max. 1 second for heating to be enabled, otherwise it returns without any action, even without measuring the PENTS_ASSIST sensor.

☝ ***During temperature scans background heating is temporarily disabled.***

Thermal Mapper heating

For the Thermal Mapper only a warm-up heating is necessary. After power-on it keeps the required temperature by hardware regulation.

Thermal Mapper warm-up phase

The thermal mapper warm-up heating procedure is exactly the same as for PENEL warm-up, except that it uses another temperature sensor (RES1).

The same general warm-up heating procedure allows also to warm-up both (PENEL and TM) at the same time. The power-on telecommand (see 0) defines a “virtual MUPUS module” (PENEL+TM) for that purpose. The generated data is again stored in the heating frames of type 0x71.

5.4 MUPUS measurement procedures

Passive PENEL temperature measurements

From November 2005 to January 2006 a series of test have been performed to find the optimum measurement procedure for PENEL temperature measurements. The telecommands implemented for these tests (RawADC-Mode, AverageADC-Mode) are still maintained in the MUPUS software version 7.0, as well as the corresponding telemetry data (0x7Axx) frames.

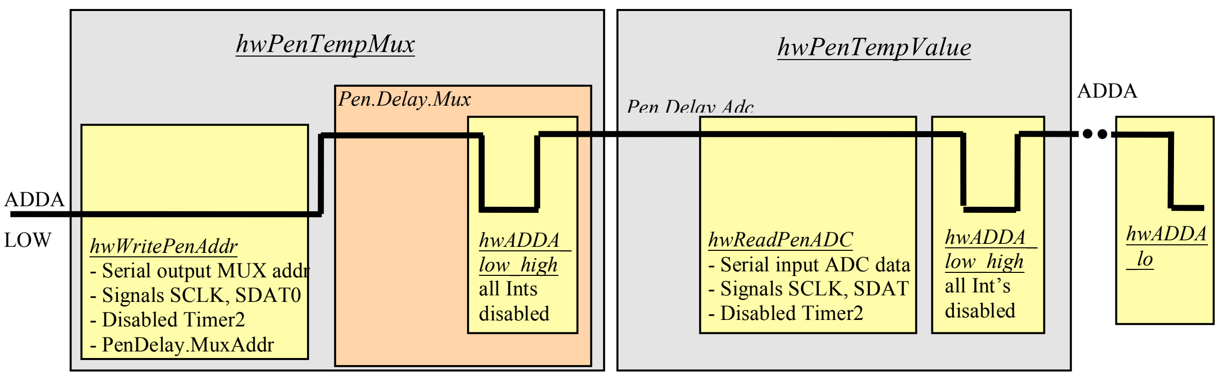
The following constraints have been observed during these tests:

1. The measurement and multiplexer (MUX) is controlled by the ADDA signal on MUPUS G-Bus-0 (G-Bus port 25). The ADDA signal is rather complex.
2. Switching the ADDA signal from logical LOW to HIGH closes the MUX for the measurement current (MUPUS uses 20mA), closes the MUX channel for ADC measurement, and initiates the ADC conversion.
3. The MUX needs some time to swing-in, therefore it is mandatory to perform a dummy ADC measurement after changing the MUX address . The corresponding configuration value is ***Pen.Delay.Mux*** which should be chosen to at least 1 µsec (default = 64 µsec).
4. To perform a PENEL temperature measurement ADDA should again be switched from logical LOW to HIGH.
5. ***ADDA should not be LOW for longer than ~20 µsec, otherwise the measured ADC value will be corrupted !***
6. Multiple ADC measurements can be performed after setting up a PENEL sensor MUX address, considering condition 5.
7. For real scientific measurements it is necessary to wait between two consecutive PENEL measurements of the same or of another channel with ADDA=low in order to cool down the current source (config parameter Pen.Delay.Meas).

Software implementation

The MUPUS software implements PENEL temperature measurement routines at three levels: serial I/O and helper routines, low-level measurement, high-level measurement.

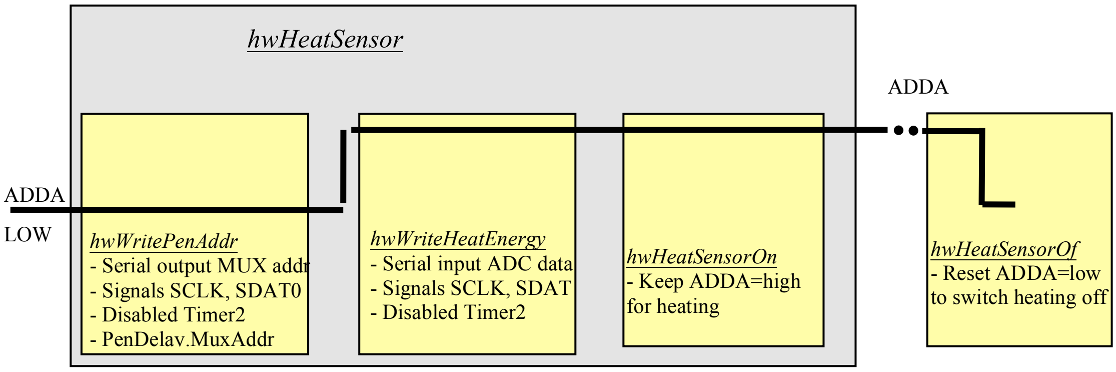
The following graphics illustrates the implementation of the measurement of a single PENEL temperature channel with respect to the ADDA signal:



PENEL sensor heating

A similar procedure is used for PENEL sensor heating.

The following graphics illustrates the implementation of heating a single PENEL sensor with respect to the ADDA signal:



Active PENEL temperature measurements (with sensor heating)

Only a single sensor can be heated or measured at a time, because different current sources are used for heating and temperature measurement. Therefore sensor heating is disrupted during temperature measurement. For THC measurements quasi-parallel heating of multiple sensors is simulated by switching between all heated sensors every 100 msec (config LT.Tick).

6 Operational modes

Main MUPUS tasks are executed by the ***mode manager*** in task#4 (see 0). Only one MUPUS mode can be executed at the same time. Another MUPUS mode can be in wait-state to be executed after the current mode has finished. When a mode-telecommand is received (0x71xx) then the mode is put to wait-state by saving the telecommand parameters, eventually overwriting an older wait-state mode. When there is active mode, the mode manager starts executing the wait-state mode thus freeing the saved mode parameters.

Each mode decides by its own, when it finishes. However a general rule can be formulated:

- ***Finite modes producing a fixed number of data sets (e.g. parameter nLoops>0) exit after complete data production. Inifinite modes, (e.g. parameter nLoops=0 or default modes) are exiting when a new mode telecommand (0x71xx) is received.***
- ***A new received mode telecommand eventually overwrites an older mode in wait-state.***

Table 17: MUPUS operational modes

Mode	Main Purpose
Power ON	Switch MUPUS devices on and performs warm-up heating, if necessary
Power OFF	Switch MUPUS devices off
MAPPER	Perform TM surface temperature and ANC-T measurements
CMAPPER	Perform TM inflight calibration as dedicated mode
TEM	Perform temperature measurements with all MUPUS sensors (PEN, TM, ANC-T)
THC	Perform thermal properties measurements by actively heating one or more PEN sensors and simultaneous scanning of all MUPUS temperature sensors
LONGTERM	Macromode containing of TEM and THC phases
Default LONGTERM	Default mode after 10 min (CFG) idle time Perform MAPPER and PENEL (if deployed) measurements
Raw ADC	Perform raw (single samples)ADC measurements with configurable channels and timings
Average ADC	Perform averaged ADC measurements with configurable channels and timings
ANCHOR mode	Fast anchor accelerator measurements during touch-down
ARM	PEN deployment
HAMMER	PEN insertion

6.1 Power ON/OFF modes

The power on/off modes are used to switch power to MUPUS devices (e.g.DSB-2, PENEL,…). In case that the device has a minimum switch-on temperature, like PENEL and TM, first a pre-heating phase is executed until the necessary temperature is reached. The power off command is used to explicitly switch a MUPUS device off. Note that MUPUS devices are not switched off automatically when measurement modes are finished. Furthermore, the devices which need warm-up heating cannot be switched off immediately during the preheating phase. In case a switch-off command is issued during the warm-up phase, the heating is continued until the switch-on temperature is reached and the device is switched ON first before it is switched off again. It should further be noted that a change of the configured switch-on temperature using the appropriate telecommand is immediately applied (while the warm-up heating is executed).

☝ ***MUPUS devices are not switched off implicitly when a mode is finished.***

☝ ***MUPUS devices cannot be immediately switched off during pre-heating***

Table 18: MUPUS devices

Device	Device number	Remarks
PENEL	1	PEN front end electronics, needs pre-heating to -40°C
DSB-2	2	
TM	3	IR sensorhead front-end-electronics, needs pre-heating to -90°C
ANC-1	4	
ANC-2	5	
PENEL + TM	6	Virtual device of PENEL + TM, allows simultaneous pre-heating of both devices

6.2 Raw ADC measurement mode

The RAW ADC measurement mode is intended for basic tests of system performance and measurement methodology. Measurement channels from both MUPUS ADC's (located on PENEL and ComDPU) can be configured arbitrarily to take readings with varying parameter settings. Every "raw" sample taken is put unprocessed to the output data packet.

6.3 Average ADC measurement mode

The Average ADC measurement mode is similar to the RAW ADC mode except that the output packest contain already averaged values. The number of samples used for averaging can be commanded.

6.4 Longterm-type measurement modes

The Longterm-type measurement modes are the main MUPUS scientific modes. In these modes all temperature sensors of the active devices (PENEL, TM, ANC-T) are scanned in regular time intervals. Longterm modes are a set of telecommands (0x71Bx) providing

- Passive measurements of PENEL temperature sensors (TEM mode).
- Active measurements of PENEL temperature sensors with heating some of the sensors (THC mode).
- Measurements of TM and ANC-T sensors (MAPPER mode).
- TM calibration (CMAPPER mode).
- A combination of the above measurements (LONGTERM mode).

Usually LONGTERM, TEM, THC modes are combined with MAPPER measurements, depending on the ***LT.MapperMode*** configuration parameter (see 5.1). To keep the telemetry data structure simple, Mapper and

Penel measurements data are written separately to telemetry with separate time stamps and power flags (see 0, 0).

Timing parameters of longterm measurements

The basic time interval of LONGTERM, TEM, THC, MAPPER measurements is a tick. The tick length is a configuration parameter (LT.Tick, default=100 msec). For THC measurements quasi-parallel heating of multiple sensors is simulated by switching between all heated sensors at every tick.

Although the measurement intervals (telecommand & config parameter) are defined in ticks, it has to be known, that the total measurement period is longer by the time needed for the measurement and averaging of 24 PENEL sensors and optionally 9 MAPPER and 2 ANCHOR temperature sensors.

The time needed for MAPPER and TEM/THC measurements is estimated by the formulas below. Furthermore, at least one tick is always applied (even when zero delay is commanded) which adds another 100 ms (with the default configuration) to the minimum time needed between two subsequent temperature scans.

☝ ***It should be noted that the exact timing is different between MUPUS Flight Model and Reference Models (RM and GRM) because of the FM ComDPU working at only ½of the nominal speed on all instructions except timer controlled wait states.***

Mapper measurement time

The nominal time (for a nominally working ComDPU, e.g. for RM and GRM) needed for averaged measurement of 9 MAPPER and 2 ANCHOR channels can be estimated by

$$t_{MAP} \approx 1ms + ANC.Delay.PwrON + 11 \times \left\{ ADC.Delay.Mux + ADC.Avrg.Mapper \times \left[ADC.Delay.Timer2 + 26\mu s + t_{Conv} \right] \right\}$$

Formula 1: Duration of a single MAPPER temperature scan (parameters in config Table 1)

Here, the configuration parameters correspond to timer waits and should be the same for the FM as well as the ADC conversion time t_{Conv}=12.5 µs , but the 26 µs are probably mostly code execution times which are expected to be twice as long for the FM.

With the default configuration settings the mapper measurement time is

$$t_{MAP} \approx 1ms + 50ms + 11 \times \left\{ 40\mu s + 160 \times \left[100\mu s + 26\mu s + 12.5\mu s \right] \right\} \approx 295ms$$

After averaging of every single MAPPER or ANCHOR channel PAUSE is called to keep multitasking alive.

PENEL measurement time

The time needed for averaged measurement of 24 PENEL sensors can be estimated by

$$t_{PENEL} \approx 24 \times \begin{Bmatrix} Pen.Avrg.Num \\ + Pen.Dummy.Num^{(*)} \end{Bmatrix} \times \begin{Bmatrix} Pen.Delay.Timer2 + Pen.Delay.MuxAddr + Pen.Delay.Mux \\ + Pen.Delay.Adc + 250\mu s + (Pen.Delay.Meas + 400\mu s) \end{Bmatrix}$$

Formula 2: Duration of a single PENEL temperature scan (parameters in config Table 1)

(*) Dummy measurements introduced in Mupus software version 7.2

Experiments with the MUPUS RM have shown that approximately 250 µs are needed for taking a single sample, this time includes the time needed for writing the MUX address and reading the ADC value to/from the serial PENEL interface.

With the default configuration settings the PENEL measurement time is

$$t_{PENEL} \approx 24 \times 10 \times \begin{Bmatrix} 0 + 256\mu s + 64\mu s \\ + 30\mu s + 250\mu s + (3 + 0.4)\, ms \end{Bmatrix} \approx 24 \times 10 \times 4.0\, ms \approx 960ms$$

Two notes:

- Microsecond delays are hard waits, while the 3 ms delay is executed as a multitasking wait. The latter adds the additional overhead of 0.4 ms to the measurement time. In case of Pen.Delay.meas=0 this additional time is also zero.

- The Mupus-FM has a reduced processor performance, thus for the Flight Model an additional overhead of approximately $24x10x250\mu s = 60\ ms$ should be added per measurement record.

TEM/THC measurement time

Two possibilities exist for the execution of TEM and THC mode:

- The configuration parameter LT.MAPPER.MODE is 0. In this case the TM is OFF and only the 24 PENEL channels are taken during each temperature scan. Therefore, the measurement time is directly given by Formula 2.
- The configuration parameter LT.MAPPER.MODE is > 0. This is the default case where each temperature scan comprises of the 24 PENEL channels plus the 9 TM and 2 ANC-T channels. The scan time is then the sum of T_{PENEL} and T_{MAPPER}.

$$t_{TEM/THC} = t_{PENEL} + t_{MAPPER}$$

During t_{PENEL} the active heating of sensors in THC mode (and in the THC phases of LONGTERM mode) is interrupted. This must be taken into account when estimating the average heating power applied to the sensors during active thermal properties measurements. Furthermore, it should be kept in mind that the background heating of the PENEL sensor head is also temporarily disabled but only during t_{PENEL}.

☝ ***Sensor heating is temporarily disabled during the temperature scan !***

☝ ***PENEL background heating is temporarily disabled during the PENEL part of the temperature scan !***

The time interval between two subsequent temperature scans is given by:

$$\Delta t = t_{TEM/THC} + N_{Ticks}\left(LT.Tick + 1\right)$$

Where LT.Tick is the configurable length of one tick (the elementary time unit used by the MUPUS software, default LT.Tick=100 ms) and N_{Ticks} is the commanded delay which is applied after each scan in multiples of one tick. Because of the multitasking architecture of the MUPUS software the interval computed by Formula 3 is a lower limit, in real operational conditions additional delays could occur occasionally.

For s/w version 7.3 and higher simultaneous measurements of the +12V supply current (using the ComDPU ADC) have been introduced into all THC activities (THC mode and THC subphases of Longterm mode). This was necessary because the power calibration performed with the RM turned out to be invalid for the FM with the consequence that measurements of the +12V current provide the only means to derive the real heating current used by the FM (by using an equivalent electrical circuit model).

These additional measurements were implemented into the existing software by prolonging each THC tick by the time required for measuring the configured number of +12V current samples (controlled by the configuration parameter ADC.Avrg.Num). The length of a THC tick in [s] can be approximately expressed by:

$$Tick_{THC} = 10^{-3}(LT.Tick + 1)[ms] + \frac{ADC.AvrgNum}{f_s}$$

where f_s is the effective sampling rate (~5000 Hz for the FM, 7000 Hz for the RM)

It should also be noted that in order not to disturb the current measurement PENEL background heating is disabled while the +12V current is sampled. A side effect is here that the available average PENEL heating power is reduced, accordingly.

☝ ***PENEL background heating is temporarily disabled during sampling of the +12V current in THC***

phases !

6.5 Default longterm mode

When there is no active mode running and no mode in wait-state, then after the configured period ***LT.Default*** a default longterm measurement is started, taking parameters from the LONGTERM configuration section (see 5.1).

The default longterm term is an infinite mode, running until a new mode telecommand is received. It performs active (THC) and passive (TEM) PENEL measurements complemented by MAPPER measurements if ***LT.MapperMode*** is non-zero.

At the start Default Longterm checks the MUPUS section of the CDMS-BRAM (x3800) which is used by MUPUS to store some context information (e.g. PEN deployed, PEN inserted,..) if the PEN has already been deployed or not. If not, Default Longterm calls MAPPER mode, otherwise phases of TEM and THC submodes are executed according to the LT configuration values. Longterm always starts with a TEM submode.

Note that Default Longterm mode can also be started by issuing the telecommand 71B0 without parameters. It is even possible to perform implicit MAPPER calibration or to change the Mapper mode during LONGTERM measurements by issuing the ***SwitchMapper*** telecommand.

6.6 Anchor mode

MUPUS Anchor mode will be used only during the descent and landing phase of PHILAE and shall mainly perform fast (~47.8 kHz sampling frequency) measurements of the accelerometers mounted inside the anchoring harpoons. The measured deceleration of the harpoon(s) when penetrating the cometary soil will provide clues about the mechanical properties of the upper cometary layers. In addition, after the accelerometer (ANC-M) sampling is finished, an averaged value of the ANC-T temperatures will be taken which is output by a text frame. The two accelerometers are to be measured sequentially, e.g. the second anchor should not be fired before MUPUS has finished the accelerometer measurements of the first anchor and is prepared to record the second shot.

MUPUS s/w supports two different versions of ANC-mode, default ANC-mode and “time-stamped” ANC-mode. The difference is that in the default variant only accelerometer values are stored and that the sampling time is assumed to be equidistant between the first and the last (post-trigger) sample whereas in “time-stamped” mode together with each sample a time stamp is stored. This halves the length of the time series (from nominally 676 ms to 338 ms) but should still be more than sufficient to measure the whole penetration process of the harpoon. Nevertheless, it is now foreseen to use the “time-stamped” variant during SDL because in the meantime tests at the GRM had revealed that the disabling of CDMS interrupts during post-trigger sampling (default in the s/w V7.ff) could lead occasionally to “hang-ups” in the communication with CDMS and should be overridden.

☝ ***CDMS interrupts should be enabled during all phases of ANC-mode! The default of disabled CDMS interrupts during post-trigger phase needs to be overridden by special command !***

Then, the implemented sequence for the anchor measurements is as follows:

1) Override disabling of CDMS interrupts by sending the telecommand: 70E9 0000 4B66 A000 A3B1
2) Configure MUPUS for usage of “time-stamped” ANC-mode
3) MUPUS receives the ANC-mode Tcmd
4) MUPUS checks the actual SST for the selected anchor for firing. Note that the SST word is requested by MUPUS after boot by SRQ and is updated during operation whenever CDMS broadcasts a new SST.
5) MUPUS starts the pre-trigger phase. This is implemented as a ring-buffer where the last N_{pre}=476 samples before the trigger are stored (N_{pre} is configurable and the new default introduced with version 7.2 is N_{pre}=476). The ADC is sampled at highest possible speed under the condition that all interrupts are enabled. This results in a sampling frequency of about 40 kHz but with some unavoidable jitter caused by the interrupts. This is acceptable because pre-trigger data are needed only to evaluate the offset and the noise of the measurements. During this phase MUPUS can react on a switch of the anchor provided that the CDMS sends

another SST message at least 100 ms before the anchor is actually fired. During this phase MUPUS would react properly even on multiple switches of the “active” anchor.

- ***Mupus needs to be informed by SST about the selected anchor at least 100 ms before firing !***
- ***It should be noted that the sampling frequency is different between MUPUS Flight Model and Reference Models (RM and GRM) because of the FM ComDPU working at only ½of the nominal speed on all instructions except timer controlled wait states.***

6) MUPUS receives INT4 (hardware trigger indicating that the harpoon has left its storage container) and starts the post-trigger phase.

7) In the post-trigger phase the ADC is sampled at the highest possible speed of 47.8 kHz (for the FM !) until a RAM page of 64 kByte is completely filled. This corresponds to 16384 samples in total (and 16384 time stamps), subdivided into 238 pre-trigger (default configuration) and 16146 post-trigger samples. The post-trigger phase in the nominal configuration has a duration of 338 ms. Because of step 1) CDMS interrupts are still enabled during this phase (without sending the command they would be disabled !) which causes some jitter in the duration of the sampling intervals. After the accelerometer measurement is finished the temperature of the anchors is measured using the ANC-T sensors and the result is output as a text frame together with some auxiliary information as, for example, the exact MUPUS time of INT4 generation, the stop time of the sampling and the SST.

- ***Note that the SST contained in the text message which is output at the end of sampling could be wrong, in case the SST was updated during the pre-trigger phase without changing the harpoon selected for firing, because the software interpretes only the bit dedicated to identification of the anchor ! This bug affects only the informational text output and has no influence on the correct reaction of the program to the SST !***

6) MUPUS waits for a change in the SST indicating either

a) safe anchoring (irrespective of an eventual change in the “active” anchor flag, for Version 7.3ff the *unsafe anchoring* flag is ignored with the default configuration whereas in earlier versions also *unsafe anchoring* causes MUPUS to leave ANC-mode), in this case ANC-mode is finished. (for Version 7.3ff no automatic transmission of data to the CDMS is done before exit of the Anchor mode in contrast to previous flight software versions, instead an explicit TC x70D3 needs to be sent to initiate the data transfer.

- ***The Safe(and Unsafe forVers. 7.2 and less) anchoring flag MUST NOT be set in the SST if the second anchor shall be shot shortly after the first one!.***

b) switch to second anchor, in this case MUPUS starts the pre-trigger phase for the second anchor. Here, the RAM page is changed to ensure not to overwrite an existing measurement. As above, this is followed by detection of INT4 and the post-trigger sampling of the accelerometer of the second anchor. In case INT4 was detected two times since start of ANC-mode (both anchors fired) the mode is exited immediately, otherwise (if the shot of one harpoon was not succesful, e.g. no INT4 was generated although the anchor firing was commanded by CDMS) MUPUS again waits for a change of the SST (note that in this case MUPUS stays in pre-trigger mode and, thus, can react on SST changes). Only one acquisition of post-trigger data per anchor is possible (only one INT4 per anchor can be generated).

7) Anchor mode may be finished also (in case no CDMS-SST about Safe/unsafe anchoring is received) by Tcmd. Therefore, each procedure using ANC-mode should include the AnchorStop Tcmd some time (e.g. 1 to a few minutes) after the nominal touchdown.

- ***For MUPUS s/w versions 7.3 or higher the AnchorStop Tcmd MUST BE used to initiate the data transfer to the CDMS !***

8) The MUPUS software is compatible with eventual failures of the anchor firings provided that the SST is changed in between (but NOT setting the safe or unsafe anchoring flags !), e.g. the following (emergency) scenario would be handled correctly:

a) CDMS initiates first anchor (1) shot without success (pyro not ignited, no INT4 generated)

b) CDMS switches to second anchor and generates corresponding SST

c) CDMS initiates second anchor shot without success (again no INT4 generated)

d) CDMS switches back to first anchor and generates corresponding SST

e) CDMS initiates shot of first anchor with second pyro (INT4 generated, ok)

f) CDMS switches to second anchor and generates corresponding SST

g) CDMS initiates shot of second anchor with second pyro (INT4 generated, ok)

☝ ***To guarantee that the background "square wave" does not disturb the accelerometer ADC measurements taken at maximum speed PENEL and DSB-2 are switched OFF at the start of the mode!***

6.7 ARM mode

ARM mode is dedicated to the deployment of the MUPUS PEN to its chosen location. Before starting this mode it is required that the PHILAE lander has already rotated into the desired azimuthal position for deployment. After calling this mode first the launch locks are released by burning of Dyneema strings. Thereafter PEN is deployed by the tubular boom to its configured length. An important exception from all other MUPUS modes should be noted here. During the time interval where the actual deployment of the boom takes place (e.g. the DD motor is working) multitasking is disabled, because it could disturb the motor operation. The consequence is that during this time span of a few minutes no controlled background heating of PENEL is performed (background heating is stuck in the situation when the multitasking was disabled => either continous heating or no heating at all is performed during this period of time) and also no transfer of science data frames is made. After successful deployment multitasking is enabled again and the MUPUS Status byte flag "PEN deployed" is set and also written into the MUPUS-BRAM at address x3800 and, furthermore, the x3805 BRAM (for MUPUS-SESAME interaction) content is updated indicating now that PEN is deployed by setting the first byte to xFF.

☝ ***Multitasking is disabled for the time interval where the DD motor works => no controlled PENEL background heating, no data frames !.***

Furthermore, after successful completion of the deployment an OPCL message is send to the CDMS. Based on receiving this message (or not) the CDMS makes the decision whether the nominal MUPUS FSS procedure shall be continued (OPCL received) or a switch to the alternative branch should be performed (no OPCL). This treatment should prevent the Lander from MUPUS hammering while still on the balcony. This feature has been introduced into the software with version V7.4, in earlier versions an additional patch TC needs to be issued before calling ARM in order to send the expected OPCL.

In addition, with V7.4 another safety mechanism was introduced which is aimed to cope with a possible failure mode during deployment, namely the blocking of the deployment by a hard obstacle before the configured deployment length is reached. The blocking of the deployment can be detected by analysis of the rotation counter which is also used to measure the deployed length of the boom. Since the stepper motor is driven with constant frequency, the expected number of rotations in a certain time interval can be calculated, and thus, the blocking can be detected.

The reaction of MUPUS in this failure case depends on the value of the configuration parameter INS.Arm.BlockChkmMode:

- INS.Arm.BlockChkmMode = 0: nothing done (as in V7.3)
- INS.Arm.BlockChkmMode = 1: DD motor is stopped, ARM is exited, no OPCL is send
- INS.Arm.BlockChkmMode = 2:
    i. Deployment stuck close to the balcony (at less than INS.ArmMinDeploy rotations): DD motor is stopped, ARM is exited, NO OPCL is send
    ii. Deployment stuck further away: DD motor is stopped, switched ON again and then retracts by INS_ArmBlockRetract rotations, OPCL is send

The detailed implementation to check whether the deployment device is considered as stuck is done in flight s/w V7.4 in the following way. Firstly, the check interval in [s] is computed using the formula:

$$dt_{check} = \frac{2000\ \mathrm{INS.ArmBlockChkRot}}{3\ INS.Fr1}$$

where INS.Fr1 is the configured final deployment motor frequency in [Hz] (Config offset 36) and INS.ArmBlockChkRot (offset 54b) is the configured number of rotations expected for the check interval. Note that for the default INS.Fr1=1000 Hz one expects 1.5 rotation counts /s. For example, the default values for INS.Fr1 and INS.ArmBlockChkRot=5 result in a check interval of dt_{check}=3.33 s. Every dt_{check} the actual value of the rotation counter is read and compared to the readout performed at the previous timestep. If the actual number of rotations counted during the last check interval is less than INS.ArmBlockChkMin (=3, default, Config offset 55a) the deployment is considered as stuck, and the DD stepper motor is stopped.

6.8 GEAR mode

GEAR mode shall be used for the actual deployment of the MUPUS PEN on the comet. It differs from ARM mode described above only by adding a check whether the Lander is in an allowed position for the deployment of the PEN at the start of the procedure. For this purpose MUPUS first reads out the Landing Gear (LG) section of the BRAM (at address x8000) and compares the actual position values of LG azimuth, tilt1, tilt2, and height with the corresponding parameters of the MUPUS configuration (at offsets 92ff…). Only if all required conditions are fulfilled the deployment is started by calling ARM mode with the default parameters. In case at least one of the conditions on the allowed deployment position is not fulfilled, only a text frame with an error message is issued and no deployment is performed.

6.9 Hammer mode

HAMMER mode is used for inserting the PEN into the ground after successful deployment (during the First Science Sequence) or for additional hammering (using HAMMER Reserve mode) to either compensate possible erosion of the cometary surface while approaching the Sun or to serve as an active seismic source for SESAME.

The default parameter setting of HAMMER mode which is intended to be used for the initial deployment starts an automated sequence which is finished when either the full insertion depth has been reached or the (configurable) maximum number of hammer strokes has been performed. Immediately after HAMMER mode starts the "HammerActive" byte is set to xFF in the x3805 BRAM. This byte remains then unchanged until hammering is completed, then it is reset to x00.

Note that PENEL is switched on at the start of the insertion procedure using the usual method of checking PENTS and, if below the threshold, start the necessary warm-up heating. In contrast, to the original implementation by SRC (e.g. s/w version V4.6) PENEL is now (V7.ff) kept on at the end of the procedure which allows directly starting with temperature measurements after the PEN insertion has been finished.

Table 19 Timing of hammer strokes

Energy level	Charging time [s] (time between individual strokes)	Time between 4-stroke-sequences [s]	Remarks
0	0.8	11.7	Measured with RM
1	3.0	20.5	
2	4.0	24.5	
3	9.5	46.5	

At the start of the sequence the PEN is successfully deployed but still connected to the deployment boom (e.g. the boom has not yet been retracted). Firstly, a depth sensor measurement at the DS reference position (at z^*_{REF}=282.5 mm, z* measured from the top of the visible part of the tube to the center of DS) is acquired. All hammering activity is done in sequences of so called 4–strokes, which means that 4 individual strokes are performed in one row, where the individual strokes follow each other as fast as possible. The time interval between the individual strokes is of the order of 1 s to 10 s and is determined by the time needed to charge the capacitor of the electromagnetic hammering mechanism (see Table 19). The capacitor can be charged with 4 different voltage levels, thereby providing four different energy levels for the stroke. Between two subsequent 4-strokes the electromagnet is operated twice (taking about 8.5 s) which is aimed at supporting the DS movement), thus, the interval between two 4-strokes is given by 4 x charging time + 8.5 s. The sequence begins with the lowest hammering energy (Level 0). After each 4-stroke another DS measurement is taken and the insertion progress evaluated. Furthermore, the content of the x3805 BRAM is updated to transfer the actual depth and timing of the hammer strokes to the SESAME experiment (see Table 15 for a detailed description of the contents of BRAM x3805). In case after a (configurable) number of 4-strokes the insertion progress is not sufficient, an automatic switch to the next higher energy level is performed. This cycle of hammering and DS measurements is repeated until about 2/3 of the maximum insertion depth has been reached. Then the retraction of the deployment boom is initialized with burning the launch locks (dyneema strings) of the DS. Afterwards the boom is retracted the insertion process continues until the maximum insertion depth has been reached.

The critical depth sensor readings (14 Bit digital values) for initiating the deployment device retraction and the stopping of the insertion process are given by:

$$V_{Retract} = INS.DV250*V_{Ref}/890 + Ins.DepthOff$$

$$V_{max} = INS.DV495*V_{Ref}/890 + Ins.DepthOff$$

Where V_{Ref} is the depth sensor reading acquired at the reference position (DS attached to DD before insertion starts) and Ins.DepthOff, INS.DV250, INS.DV495 are configuration parameters (located at offsets 44,50 and 51, respctively) which can be used to control the insertion depth where these actions are started. The parameters INS.DV250 and INS.DV495 were put into the configuration with s/w version V7.4 (default values INS.DV250=128, INS.DV495=37), before they were hard coded with the values 256 (INS.DV250) and 37 (INS.DV495) and could only be changed by dedicated special patch TCs. With the configured default values the DS position at retraction is about 50 mm below the upper end of the PEN rod. Note that an eventual change in the resistance of the conductive paint at the outside of the PEN tube over the years is taken into account by using the actual DS reading at the reference position instead of a constant in the equations above.

Furthermore the energy level of the strokes is increased when the minimum number of 4-strokes configured for each energy level have been made (Config offsets 46-48) and, in addition, the progress between the last two 4-strokes (subsequent DS readings) is less than $5*V_{Ref}/890$ which corresponds to approximately 1.7 mm.

7 Software patches 7.1 - 7.4

7.1 Version 7.1 changes

MUPUS software version 7.1 is a small (~2 kByte) patch of version 7.0 fixing version 7.0 bugs and addressing hardware problems observed with the MUPUS flight model (ComDPU#01 - FM).:

Compilation Date of release version: 7-MAR-2007 15:29:43

2006/12/11	Bug fix#004: RawADC: No PENEL background heating if blocking TM output - adcSoftLoop: pwrHeatDisable...pwrHeatEnable for each separate measurement !
2006/12/14	Bug fix#005: CDMS STAT sync can be lost - no execution of USER$DATA - The CDMS interrupt service should not rely on the STAT SYNC (IVEC=4 flag) quickly following a SSADR-SYNC in DATA(T/R=1) messages or a CDM/DAT-SYNC in COMMAND (T/R=0) messages. - Thus for CDMS COMMAND messages: * Count the incoming CMD words. * Call USER$CMD after reception of the last CMD word. - For CDMS DATA messages: * Call USER$DATA after reception of the SSADR-SYNC (IVEC=1) * Ignore the following IVEC=4 interrupt. - For CDMS STAT messages: * Bind USER$STAT callback to IVEC=1 (earlier)
2006/12/14	Bug fix#006: DumpRAM produces only half of the requested data
2007/01/02	Added TEST: interrupt counting (Tcmd = 7071 sec)
	Added ANCHOR TEST mode including timestamps for each anchor measurement
2007/01/29	Fixed Power Limitation Tables (PwrLimitTbl) - was HEX instead of DECIMAL
2007/01/30	Added memory & performance test at MUPUS startup
2007/02/01	Added HK values: TcmdReceived,TcmdExec,TcmdError

7.2 Version 7.2 changes

MUPUS software version 7.2 is provided as a patch against version 7.0. It includes the complete 7.1 patch added by a few software changes according to ***"ECR-v72.doc" (12-Feb-2008)***:

Compilation Date of release version: 11-MAR-2008 14:44:27

2008/02/18	ANC mode sampling frequency too low (33 kHZ) due to reduced MUPUS-FM performance. - Modified post-trigger Anchor ISR to get a sampling rate close to 50 kHz
2008/02/18	ANC mode temperature measurements - Inserted ANC-T1 & ANC-T2 measurement (average) into TEXT frame issued after each anchor measurement : "ANC stop: Status= ... ANC-T1= ... ANC-T2=..."
2008/02/18	ANC mode power switching - Do not switch off anchor electronics at the end of ANC mode to allow continues ANCHOR temperature measurements after ANC mode.
2008/02/19	Disturbed PENTS-Assist measurements (Penel HK#4) during THC active heating - Penel hardware (G-Bus): when PENTS-Assist is measured, heating is switched off/on physically ==> Possible solution: Configure Pen.Delay.MuxAddr = 256
	Added ANCHOR TEST mode including timestamps for each anchor measurement
2008/02/19	THC mode - Added MAPPER/ANCHOR measurements at THC startup
2008/02/19	Perform dummy PENEL measurements of all 24 channels before THC/TEM measurements.

	- Added Config parameter Pen.Dummy.Num = Number of dummy measurements of all 24 PENEL channels before averaging (0=none)
2008/02/19	Reduction of auxiliary data from pre- and background heating of sensor heads - Added config value Heat.DataFlag to send only every N'th heating value, or no data (=0)

7.3 Version 7.3 changes

MUPUS software version 7.3 is provided as a e patch against version 7.2. It includes the complete 7.2 functionality plus:

Compilation date of release version: 12-FEB-2010 12:25:43

2009/10/21	Measure THC-Power (current for each heated sensor + 12V voltage)
2009/10/21	Add an optional delay tick after each heat sensor (configuration LT.THC.Pause)
2010/01/25	Increase the timeout in HAMMER mode (PEN insertion) to 12 s (from 3s in V7.2) to be compliant with hammer capacitor charging times also for energy levels 2 and 3
2010/01/25	Change ANCHOR mode such that no automatic transfer of accelerometer data to the CDMS is carried out (as in versions 7.0-7.2), instead an explicit telecommand “AnchorStop” is always required to initiate the data transfer.
2010/01/27	Fix bug in writing of MUPUS-SESAME BRAM (address x3805)

7.4 Version 7.4 changes

MUPUS software version 7.4 is provided as a patch against version 7.3. It includes the complete 7.3 functionality plus:

Compilation date of release version: 12-DEC-2013 12:25:43

2013/11/12	- *** Starting implementation of Mupus 7.4 patches *** - ECR#7.4: Problems solved in 7.3 by direct code execution or Load-RAM TC's - 4B) Patch GEAR & ARM-Modes to send OPCL status when finished successfully * Side effect: DSB2 is not powered OFF at the end of armDeploy ! - 4A) Add a 5sec delay in ARM mode before DD motor start - 4D) Introduce "new" conversion factors for interpretation of LG HK - 4C) Condition for retracting & Max. insertion depth can be configured in INS.DV250 & INS.DV495 - 1B, 1C) Added INS.SafTab table entry for energy level=3 to limit the number of max. energy strokes if there is no insertion progress. - 1A) DD retraction shall be automatically performed after hammering. In v7.3 PEN may be half-inserted when max. number of strokes have been done. Missing DD retraction would block the Lander !!!
2013/11/13	- Mupus 7.4 patches - 3) Changed INS.Fr1 from 2400 (0x0960) ==> 1000 (0x03E8) - 2) Check if FORWARD deployment blocked by barrier, then STOP or RETRACT - 1B, 1C) The number of level=2 strokes was 500 (unlimited), changed to =8
2013/11/14	- Mupus 7.4 patches - 1C) Few minor fixes

	- 5) Remove "dead code" of compton measurements at 2-6 depth levels during hammering - 1C) Make the number of rotations (10) an the limit (5) configurable in INS.ArmBlockChkRot/Min - 1B, 1C) STOP hammering at energy level=3 by INS.SafTab[3] only if we are already retracted
2013/11/18	- Mupus 7.4 patches - modified ARM deployment to send OPCL only if we are away from Balcony (~125 rotations).
2013/11/25	- Mupus 7.4 patches - Keep CDMS-INT1 enabled during fast ADC sampling after INT4 - Added INS.DV007=5 and INS.DVLow=20 to Config - Check motor blocking even for retraction (exit if blocked) - Added StackPointer-output to end of ARM
2013/11/27	- Mupus 7.4 patches - Add Thermal-Mapper operational heater flag to TM.Status bit7=1: Internal heater ON
2013/12/12	- Mupus 7.4 final delivery - Fixed bug in Landing Gear Height (invalid sign) - Changed Landing Gear defaults in Config

8 Telecommands

8.1 Overview of all MUPUS telecommands

General telecommand structure

Table 20: General telecommand structure

1 Word	= DEBx	Common-DPU Debug Monitor	Only in Common-DPU mode: First telecommand must be received within 1 min after power-on, otherwise the MUPUSS-SW is booted.
	= 7fcc	MUPUS telecommand f = command flag(s): 1=MODE telecommand cc = command code	Mode and non-mode telecommands are executed in different context, see 0.
max. 30 Words		Telecommand parameters	Note that there are some telecommands with a variable number of parameters,. The command length is known from the WRDC field in the CDMS SSADR word.
1 Word		Telecommand checksum	Sum of the whole telecommand (incl. checksum) = 0000

Repeated telecommands are ignored.

MUPUS telecommands

Table 21: MUPUS telecommands

Code (hex)	Name	Parameters	Description	Remarks
7100	NoMode	<any>	Perform no Mupus MODE operation	Wait until a new Mupus MODE telecommand is received. Prevents Mupus DEFAULT mode from being executed.
7001	Config	Ofs val1 val2 … valN	Set global configuration	
700A	ConfigSave	<none>	Save configuration to EEPROM	The current configuration becomes the default and will be restored at the next power on.
700B	ConfigUnsave	<none>	Unsave default configuration from EEPROM	Reset the current configuration to the default.
700D	ConfigDump	<none>	Dump current MUPUS configuration	Produce a single configuration dump frame (type=0x7D00)
7110	PowerOff-Mode	dev1 … devN	Power-On Mupus subdevices	MUPUS subdevices 1 = PENEL 2 = DSB-2 3 = MAPPER 4 = ANC-1 5 = ANC-2 6 = PENEL + MAPPER
7111	PowerOn-Mode	dev1 … devN	Power-On Mupus subdevices	Implemented as a MODE telecommand, because PENEL & Thermal MAPPER need pre-heating before power on. Optionally produces Mupus heating frames (type=0x7100)
7018	SwitchMapper	Mode	Switch MAPPER power nominal/low/calibration	Implemented as a no-mode telecommand to be used during Longterm/THC/TEM measurements.

Code (hex)	Name	Parameters	Description	Remarks
				Mode=1 => nominal power Mode=2 => low power Mode=3 => low power Mode=4 => calibration
7024	DumpBRAM	adr1 … adrN	Dump (multiple) Backup-RAM records	Reads Backup-RAM records from any Rosetta-Lander instrument and dumps them with a time stamp to telemetry frames (type=0x7C00). Implemented as a no-mode telecommand to be used for monitoring MUPUS-modes.
7025	UploadBRAM	adr data1 … dataN	Upload MUPUS Backup-RAM record	Upload a single Backup-RAM ord (max. 29 words, all other words =0). Other than MUPUS addresses ofs=0x38xx produce errors. Implemented as a no-mode telecommand to be used for testing MUPUS-modes.
71A0	RawADC-Mode	mode delay1 delay2 num chan1…chanN	Raw ADC measurements	Produces a single ConfigDump (type=0x7D00) telemetry frame and a number of ADC data frames (type=0x7Axx)
71A1	AverageADC-Mode	mode avrg delay1 delay2 num chan1…chanN	Averaged ADC measurements	Produces a single ConfigDump (type=0x7D00) telemetry frame and a number of ADC data frames (type=0x7Axx)
71B0	Longterm-Mode	[heatIdx numHeat intervalTHC numTHC intervalTEM numTEM] [nLoops]	Longterm measurements: active (THC), passive (TEM) PENEL and MAPPER measurements	Without parameters: Default longterm 6 parameters: Infinite longterm measurements 7 parameters: nLoops longterm measurement cycles of numTHC+numTEM measurements Produces a single ConfigDump (0x7D00) telemetry frame and a number of Penel/Mapper data frames (type=0x7300/0x7400)
71B1	TEM-Mode	interval [numTEM]	Passive PENEL and MAPPER (optional) measurements	1 parameter: Infinite TEM measurements 2 parameters: numTEM TEM measurements Produces a single ConfigDump (0x7D00) telemetry frame and a number of

Code (hex)	Name	Parameters	Description	Remarks
				Penel/Mapper data frames (type=0x7301/0x7401)
71B2	THC-Mode	idxHeat numHeat interval [numTHC]	Active (heating) PENEL and MAPPER (optional) measurements	1 parameter: Infinite THC measurements 2 parameters: numTHC THC measurements Produces a single ConfigDump (type=0x7D00) telemetry frame and a number of Penel/Mapper data frames (type=0x7302/0x7402)
71B3	Mapper-Mode	interval [numMAP]	Mapper measurements	1 parameter: Infinite MAPPER measurements 2 parameters: numMAP MAPPER measurements Produces a single ConfigDump (type=0x7D00) telemetry frame and a number of Mapper data frames (type=0x7403)
71B4	CMapper-Mode	interval preMAP calMAP postMAP	Mapper calibration	Produces a single ConfigDump (type=0x7D00) telemetry frame and a number of Mapper data frames (type=0x7404)
71C0	Arm-Mode	mode parm1 parm2 parm3 parm4	ARM release and deployment	All 5 TC parameters (mode, parm1…parm4) MUST BE provided, otherwise only the message “Invalid telecommand length” is returned parm4 is a dummy parameter in all cases If one of the parameters parm1…parm3 is 0 then the corresponding value is taken from the MUPUS Config. mode=0 Standard mode - Release the PEN from the balcony - Deploys it ~1 meter from the Lander parm1= Initial Motor Frequency parm2= Final Motor Frequency parm3= Number of motor rotations mode=1 Reserve Deploy (including release burning)

Code (hex)	Name	Parameters	Description	Remarks
				parm1= Initial Motor Frequency parm2= Number of motor rotations mode=2 Only Release mode=3 Only Deploy in the Vacuum Thermal Chamber - with different parameters parm1= Initial Motor Frequency parm2= Final Motor Frequency parm3= Number of motor rotations
71C8	Hammer-Mode	mode parm1 parm2 parm3 parm4		All 5 TC parameters (mode, parm1…parm4) MUST BE provided, otherwise only the message “Invalid telecommand length” is returned parm4 is a dummy parameter in all cases If one of the parameters parm1…parm3 is 0 then the corresponding value is taken from the MUPUS Config. mode=0 Standard FLIGHT mode Insertion & Rectracting mode=1 Reserve Insertion - Fixed hammer energy and number of hammer strokes parm1= Hammer energy level 0..3, default=2 parm2= Number of hammer 4-strokes, default=100 mode=2 Reserve retracting - Penetrator retracting with constant motor frequency parm1= Motor frequency 200..299, default=200 parm2= Number of motor rotations, default=750

Code (hex)	Name	Parameters	Description	Remarks
				mode=3 Release (TV-chamber) mode=4 Insertion (TV-chamber) - Fixed hammer energy and number of hammer strokes parm1= Hammer energy level 0..3, default=2 parm2= Number of hammer 4-strokes, default=1 mode=5 Standalone default retracting (originally designated as TV mode) parm1= Initial motor frequency 200..299, default=200 parm2= Final motor frequency 200..2400, default=2400 parm3= Number of motor rotations, default=1033 mode=6 Hammer test mode (not implemented) mode=7 Reserve heating (not implemented)
71D0	Anchor-Mode	<none>		Anchor mode
70D3	AnchorStop	flag	Stop anchor mode / simulate int4	flag=0 Stop anchor mode, optionally (if data is available) send data of both anchors to CDMS flag=1 Simulate Anchor fire (int4), don't stop anchor mode flag=2 Stop anchor mode, don't send data flag=3 Stop anchor mode, send data of ANC-1 (RAM page 2) flag=4 Stop anchor mode, send data of ANC-2 (RAM page 3) **Note: for versions 7.3ff AnchorStop MUST BE send to initiate the transfer of accelerometry data !**
71E0	Gear-Mode	<none>	Check Landing gear position and deploy	If the Gear-Simulation TC was used before to simulate LG BRAM contents no actual deployment is started if all conditions are ok, instead a text message

Code (hex)	Name	Parameters	Description	Remarks
				"Simulated ARM mode started" is issued.
70E3	GearSimulate	hk13 hk14 hk15 hk16 hk17	Simulate landing gear BRAM data	Only intended to use for checks of GEAR mode on the GRM or FM **without** starting ARM mode ! For tests of Gear mode at the RM load the necessary contents into the CDMS simulator LG BRAM at address x8000
70E8	ExecCode	code code ... 116A A020 1171 A020	Execute short (<=30 words) code	Compiled machine *code* should finish with a RETURN statement Switch TM (warmup) heater ON explicitly (can be used during TM operation) Switch TM (warmup) heater OFF explicitly
70E9	LoadRAM	page adr data data ...	Load code/data into RAM page	Upload *data* into RAM *page:adr*
70EA	DumpRAM	page adr len	Dump code/data from RAM	Dump min. *len* bytes from code page to type=0x7F frames. Frames are filled up.
70EB	CopyRAM	page1 adr1 page2 adr2 size	Copy code/data from/to RAM	Copy *size* data words from *page1:adr1* to *page2:adr2*
70EC	FillRAM	page adr times pat1 ... patN	Fill RAM with repeated pattern	Fill RAM at page:adr with pattern=(pat1...patN), repeat times
70ED	BurnEEPROM	eadr page adr size {chksum}	Burn EEPROM, optionally after checksum validation	Burn *size* words from *page:adr* to EEPROM *eadr* If *chksum* is specified, it must match chksum = - Sum(size words) % 65536
70EE	BootRAM	page adr size	Boot code from RAM	Boot code from *page:adr*
70EF	BootEEPROM	eadr	Boot code from EEPROM	Boot from EEPROM *eadr* after checksum validation
70F0	Sleep	sec	Sleep sec seconds	May be used to wait for completion of some background operation (e.g. DMA)
70F4	WaitDataComplete	sec	Wait until instrument operation complete - no more telemetry data	Sec – Poll interval in seconds Conditions for OCPL-operation complete: 1. No TM data in frame buffer 2. No TM data sent during the last poll period
70F8	TcmdLog	<none>	Dump telecommand log	Dump last received telecommands to a single telemetry frame (type=0x7E)
70FF	Noop	<none>	Do nothing	Valid telecommand without operation

Code (hex)	Name	Parameters	Description	Remarks
Version 7.1 technological telecommands				
7071	TestCountISR	sec	Mupus interrupt counting	Sec > 0: Redirect all interrupt vectors to perform interrupt counting Send a TM frame every ***sec*** seconds Sec = 0: Stop interrupt counting
7072	TestDelay	ticks	Simulate some delayed ISR execution (testing new CDMS ISR's)	Ticks > 0: Delay (heavily disable interrupts) in MAIN task Note: Delay>2500 (500µsec) will definitely disrupt CDMS communication !!! Ticks = 0: Stop delaying
707D	TestAnchorMode	<none>	Anchor mode with timestamp	Anchor mode testing: • Patch ANCHOR mode to store RTX-TC0 together with ANCHOR measurements. • Every anchor measurement is accompanied by a 16-bit timer counter value (1 DU = 0.2 microseconds), running backwards. • Test anchor mode frames types = 0x7611/0x7612 for Main/Redundant anchor. • Note: Even indexed values are frame counters, while odd indexed values are anchor measurements. • Reduces the number of anchor mode measurements by a factor of 2 (total of 16384 measurements) **Limitation:** There is **NO WAY** to run the "normal" anchor mode once this telecommand is issued until **MUPUS is REBOOTED!**

Code (hex)	Name	Parameters	Description	Remarks
707F	FuseHardware	mask=0x701F	“Fuse”MUPUS hardware (G-Bus)	"Fuse" GBus access, which is originally “defused” at startup. The mask parameter is a XOR mask for obfuscating code. It must be the same as used for “defusing”.

Table 22 Selected special telecommands using LoadRAM or Execute Code (V7.4 only !)

Code	**Description**	**Remarks**
70E9 0000 3AD4 A000 B443 71C8 0005 0000 0000 0300 0000 8B33	Disbable release (burning) of Depth sensor in arm retract procedures Then retract boom fully with PEN still attached using HAMMER mode 5 (“TV retracting”, uses PatchArmotor Work of V7.4)	Can be used to retract boom without releasing PEN in case of deployment failure, to be used in conjunction with HAMMER retracting Tcs
70e9 0000 3aa8 a000 b46f 71c0 0001 00c8 0005 0000 0000 8d72 7110 0002 8eee	Disable release in ARM deployment routines Deploy 5 rotations Switch DSB2 OFF	Should be used ONLY if first release already successfully performed ! Can be used to remove the overcurrent condition on the +5V line asscociated with prior DD retraction
70E8 1F17 1F14 A020 B0CD 70E8 1F25 1F14 A020 B0BF	Operate electromagnet once Operate electromagnet twice	

8.2 Detailed telecommand description

Mupus configuration telecommands

Config

The *Config* telecommand writes a single or multiple parameters into the global Mupus configuration table. Limited only by the telecommand size, a maximum number of 29 configuration parameters can be set in a single telecommand.

For some parameters a change will have an immediate effect of a Mupus mode running in background (e.g. the heating temperature setings). Other parameters will need a mode restart to take effect. Therefore it is always recommended to change configuration settings before starting a Mupus mode.

The configuration parameters and their address offsets are described in Table 13**.**

7001 ofs val1 … valN		
Parameters:		
ofs	=0.. 119	The configuration table offset in words (Word Nº column in Table 13**)**
val1..valN	(unsigned)	N=1..29 configuration parameters to write.
Notes:		
- Maximum 29 unsigned 16-bit integer parameters can be set in a single telecommand. - Due to the general character of the *Config* telecommand no parameter checking is performed. - Some parameter changes will have an immediate effect, others will need a Mupus mode restart.		
Telemetry:		
<none>		No telemetry frames are produced

ConfigSave

The *ConfigSave* telecommand is used to save the complete configuration table into EEPROM. The current configuration becomes the default and will be restored after when the instrument software is started.

To avoid configuration mismatch between different Mupus software versions, a software version & time stamp is stored together with the configuration table. A default configuration is loaded from only if the software version & time matches the configuration record.

The configuration parameters and their address offsets are described in Table 13. The EEPROM addresses for stored default configurations are listed in Table 3.

700A		
Parameters:		
<none>		
Notes:		
- The EEPROM location of the default configuration is hardcoded in the Mupus software and cannot be specified. - If several software versions are used in parallel, it is recommended to use different EEPROM		

		locations for storing the default configuration.		
	Telemetry:			
		<none>		No telemetry frames are produced

ConfigUnsave

The *ConfigUnsave* telecommand immediately restores the default configuration from EEPROM.

For some parameters a change will have an immediate effect of a Mupus mode running in background. Other parameters will need a mode restart to take effect. Therefore its always a good idea to change configuration settings before starting a Mupus mode.

The configuration parameters and their address offsets are described in Table 13. The EEPROM addresses for stored default configurations are listed in Table 3.

700B				
	Parameters:			
		<none>		
	Notes:			
		- The EEPROM location of the default configuration is hardcoded in the Mupus software and cannot be specified. - If several software versions are used in parallel, it is recommended to use different EEPROM locations for storing the default configuration. - Some parameter changes will have an immediate effect, others will need a Mupus mode restart.		
	Telemetry:			
		<none>		No telemetry frames are produced

ConfigDump

The *ConfigDump* telecommand sends the currently active configuration including the current software version & time stamp as a single telemetry frame type=0x7D (see Table 44). Note that a ConfigDump will also be issued automatically at the beginning of every Longterm-type mode.

700D				
	Parameters:			
		<none>		
	Notes:			
		- Mupus science modes implicitly dump the configuration at mode startup.		
	Telemetry:			
		1 frame	Type=0x7D	MUPUS configuration dump (see Table 44)

Mupus power related telecommands

PowerOn-Mode

The *PowerOn-Mode* telecommand switches on one ore more Mupus subunits in the order as they appear in the telecommand parameter list.

Some subunits are immediately switched on, others (PENEL and MAPPER) may need preheating which can take a quite long time. That's why this telecommand is implemented as a separate Mupus mode. To speedup preheating the pseudo-unit 6 (PENEL+MAPPER) allows parallel preheating of both subunits.

The *PowerOn-Mode* stops automatically, when all units are switched on. While the PENEL & MAPPER subunits are on, they will be kept at a configured temperature range by means of continuous background heating, optionally producing heating frames from time to time.

7111 dev1 … devN		
Parameters:		
dev1…devN	1..6	Mupus subunit identifier: 1 = PENEL 2 = DSB-2 3 = MAPPER 4 = ANC-1 5 = ANC-2 6 = PENEL + MAPPER
Notes:		
- Subunits 1=PENEL and 3=MAPPER need preheating. - The pseudo-unit 6 allows PENEL and MAPPER preheating in parallel. - Heating frame output is affected by several configuration parameters.		
Telemetry:		
? frames	Type=0x71	A number of Mupus heating frames (see Table 36)

PowerOff-Mode

The *PowerOff* telecommand switches off one ore more Mupus subunits in the order as they appear in the telecommand parameter list.

Switching off the PENEL and/or MAPPER subunits also stops background heating and heating frame production.

7110 dev1 … devN		
Parameters:		
dev1…devN	1..6	Mupus subunit identifier: 1 = PENEL 2 = DSB-2 3 = MAPPER 4 = ANC-1 5 = ANC-2 6 = PENEL + MAPPER
Notes:		

	Telemetry:			
		<none>		No telemetry frames are produced

SwitchMapper

The *SwitchMapper* telecommand changes the current MAPPER mode to use low, nominal or calibration power. *SwitchMapper* is implemented as a none-mode telecommand used to change the MAPPER power during a running Longterm, THC, TEM measurements without having to stop and restart the measurement mode.

7018 mode				
	Parameters:			
		mode	1..4	MAPPER mode: 1 = nominal power 2 = low power 3 = low power (same as mode=2) 4 = calibration
	Notes:			
	Telemetry:			
		<none>		No telemetry frames are produced

foobar

The foobar telecommand

71A0 mode				
	Parameters:			
		mode	=0	foobar
	Notes:			
		<none>		
	Telemetry:			
		<none>		No telemetry frames are produced

Mupus science mode telecommands

NoMode

The *NoMode* telecommand does not perform any active Mupus operation, but puts Mupus into an idle state. Mupus *NoMode* runs until a new Mupus MODE telecommand ist received. The main purpose of the *NoMode* telecommand is to prevent the execution of the Mupus DEFAULT mode (longterm), it can also be used as a telecommand placeholder.

7100 <any>			
Parameters:			
	<any>		Any number and type of parameters are accepted.
Notes:			
	<none>		
Telemetry:			
	<none>		No telemetry frames are produced

RawADC-Mode

The RawADC telecommand is intended for test purposes but may also be useful for flight operations. It performs a specified number of raw ADC measurements cycles of up to 26 different MUPUS ADC channels. Channels are numbered 0..55 (0x00..0x3F) according to Table 5 - Table 7. In a measurement cycle maximum 26 channels are scanned as fast as possible. An additional delay between two consecutive cycles can be specified in milliseconds or microseconds. The produced telemetry data contains the channel number, the measurement time with microsecond resolution (with some limitations) and the measured raw ADC value.

71A0 mode delay1 delay2 num chan1…chanN			
Parameters:			
	mode	=0	"Soft" mode with complete background services (multitasking); Delay in milliseconds; Multiple ADC channels allowed.
		=1	"Hard" single channel mode; No background services (telecommands, frames, HK update, Penel heating, etc.), only interrupts are serviced; Delay in microseconds. No MUX reset between measurements. MUX is closed during complete measurement (20 mA current !).
		=2	"Hard" multi-channel mode; No background services (telecommands, frames, HK update, Penel heating, etc.), only interrupts are serviced; Delay in microseconds.
	delay1	=0..65535	Delay in milliseconds (mode=0) or microseconds (mode=1,2) between each two measurements (channels), additional to the time used for measurements.
	delay2	=0..65535	Delay in milliseconds (mode=0) or microseconds (mode=1,2) between two measurement cycles, additional to delay1.
	num	=0..65535	Number of measurement cycles.
	chan1		The MUPUS channel number according to Table 5 - Table 7
		=0..15	Common-DPU ADC & MUX channel (see Table 5); Channel 13 does not

				make sense, it is the external MUPUS MUX.
			=16..31	Common-DPU ADC & external MUX (see Table 6)
			=32..55	Penel MUX with serial connection to G-BUS (see Table 5). Channels 32..47 are resistors R1..R16. Channels 48..55 are housekeeping channels HK1..HK8.
		…		Next channel (mode=1,3 only)
		chanN		Maximum of 26 channels can be specified in a single telecommand
	Notes:			
		The measurement time is recorded after the ADC measurement is ready and the value is transmitted. Channels 0..31 are measured by the Common-DPU by ADC interrupt, while the PENEL ADC has no interrupt line, but the DPU waits a certain fixed time for the measurement to be finished. It must be noted, that other interrupts (CDMS) are allowed during measurements. Also the CPU performance of 5 MHz is of the same order of magnitude, leading to some limited response time. In "hard" mode the maximum number of samples (=number of channels x number of scans) is limited to 8192 (0x2000 for single channel mode=1, just filling one memory page)		
	Telemetry:			
		1 frame	Type=0x7D	MUPUS configuration dump (see Table 44)
		? frames	Type=0x7A	MUPUS ADC measurement frames (see Table 43)

AverageADC-Mode

The AverageADC telecommand is intended for test purposes but may also be useful for flight operations. It performs a specified number of averaged ADC measurements cycles of up to 25 different MUPUS ADC channels. Channels are numbered 0..55 (0x00..0x3F) according to Table 5 - Table 7. An additional delay between two consecutive cycles can be specified in milliseconds. The produced telemetry data contains the channel number, the measurement time with microsecond resolution (with some limitations) and the measured raw ADC value.

71A1 mode avrg delay1 delay2 num chan1…chanN				
	Parameters:			
		mode	=0	"Soft" mode with complete background services (multitasking); Delay in milliseconds; Multiple ADC channels allowed.
			=1	"Hard" single channel mode; No background services (telecommands, frames, HK update, Penel heating, etc.), only interrupts are serviced; Delay in microseconds. No MUX reset between measurements.
			=2	"Hard" multi-channel mode; No background services (telecommands, frames, HK update, Penel heating, etc.), only interrupts are serviced; Delay in microseconds.
		avrg	=1..32767	Number of measurements to average
		delay1	=0..65535	Delay in milliseconds (mode=0) or microseconds (mode=1,2) between each two measurements (channels), additional to the time used for measurements.
		delay2	=0..65535	Delay in milliseconds (mode=0) or microseconds (mode=1,2)

				between two measurement cycles, additional to delay1.
		num	=0..65535	Number of measurement cycles.
		chan1		The MUPUS channel number according to Table 5 - Table 7
			=0..15	Common-DPU ADC & MUX channel (see Table 5); Channel 13 does not make sense, it is the external MUPUS MUX.
			=16..31	Common-DPU ADC & external MUX (see Table 6)
			=32..55	Penel MUX with serial connection to G-BUS (see Table 5). Channels 32..47 are resistors R1..R16. Channels 48..55 are housekeeping channels HK1..HK8.
		…		Next channel (mode=1,3 only)
		chanN		Maximum of 25 channels can be specified in a single telecommand
	Notes:			
		The measurement time is recorded after the ADC measurement is ready and the value is transmitted. Channels 0..31 are measured by the Common-DPU by ADC interrupt, while the PENEL ADC has no interrupt line, but the DPU waits a certain fixed time for the measurement to be finished. It must be noted, that other interrupts (CDMS) are allowed during measurements. Also the CPU performance of 5 MHz is of the same order of magnitude, leading to some limited response time. In “hard” mode the maximum number of samples (=number of channels x number of scans) is limited to 8192 (0x2000 for single channel mode=1, just filling one memory page)		
	Telemetry:			
	1 frame	Type=0x7D	MUPUS configuration dump (see Table 44)	
	? frames	Type=0x7A	MUPUS ADC measurement frames (see Table 43)	

Longterm-Mode

The *Longterm-Mode* is the main measurement mode of the Mupus instrument. It performs active and passive temperature measurement of the PENEL and the Thermal MAPPER sensors.

There are three ways to start longterm measurements:

1. ***[Default Longterm-Mode]:*** Without parameters the *Longterm-Mode* telecommand executes the default LONGTERM mode. Longterm measurement parameters are taken from the default configuration (LT.*.* section of Table 13) at mode initialization During *Longterm-Mode* LT.*.* parameter changes do not have any impact on the current measurements, but need a *Longterm-Mode* restart. The *Default Longterm-Mode* runs until any other Mupus-Mode telecommand is received. Note that the the *Default Longterm-Mode* first checks the BRAM at address x3800 to find out if the PEN is already deployed or not. In case the PEN is not yet deployed it automatically calls MAPPER mode and executes this mode using the default parameters from the configuration.
2. ***[Infinite Longterm-Mode]:*** When all six *Longterm-Mode* measurement parameters are specified in the telecommand, then these settings are used instead of the default configuration parameters. The *Infinite Longterm-Mode* runs until another Mupus-Mode telecommand is received.
3. ***[Finite Longterm-Mode]:*** Additionally the number of loops can be specified together with all *Longterm-Mode* measurement parameters. The *Longterm-Mode* runs until the specified number of longterm cycles is finished. The execution of another received Mupus-Mode telecommand is postponed until all measurements are completed.

The heating during active THC measurements is performed in longterm-ticks of 100 milliseconds by default (LT.Tick). Quasi-parallel heating of LT.NumHeat sensors means:

- During each THC cycle the same group of LT.NumHeat PENEL sensors from the heating table LT.HeatTab is heated.
- Each sensor is heated for a single longterm-tick.
- At the next tick the next sensor of this sensor group is heated.
- The next THC cycle of Default Longterm mode continues with heating the next sensor group of LT.HeatTab, writing the status to the CDMS BackupRAM for later continuation.

The duration of a longterm measurement cycle and the number of produced PENEL and MAPPER measurement telemetry frames per cycle depends on the telecommand parameters or default LT.*.* configuration parameters as well as the general measurement configuration ADC.*.* and Pen.*.* Table 13). The following formulas help to calculate the data rate and measurement duration:

Formula 3: Longterm-Mode cycle duration and data rate

$$T_{CycleLT} \approx \left\{ \begin{array}{l} numTHC \times (intervalTHC \times LT.Tick + T_{TEM/THC} + T_{MAP}) \\ + numTEM \times (intervalTEM \times LT.Tick + T_{TEM/THC} + T_{MAP}) \end{array} \right\} \approx \text{(default) } 2h54min$$

$$N_{PENEL} = (numTEM + numTHC)/4 \quad \text{[frames per cycle]}$$

$$N_{MAPPER} = (numTEM + numTHC)/8 \quad \text{[frames per cycle]}$$

71B0 **71B0 heatIdx numHeat intervalTHC numTHC intervalTEM numTEM** **71B0 heatIdx numHeat intervalTHC numTHC intervalTEM numTEM nLoops**		
Parameters:		
heatIdx	Default: LT.IdxHeat	Starting index to the heating table LT.HeatTab. If -1 (0xFFFF) is specified, then the starting index is taken from the previously stored CDMS BackupRAM record.
numHeat	Default: LT.NumHeat	Number of PENEL sensors to heat quasi-simultaneously.
intervalTHC	Default: LT.THC.Interval	Active THC measurement interval in longterm ticks (LT.Ticks, 100 milliseconds by default).
numTHC	Default: LT.THC.Num	Number of measurement records during the active THC phase.
intervalTEM	Default: LT.TEM.Interval	Passive TEM measurement interval in longterm ticks (LT.Ticks, 100 milliseconds by default).
numTEM	Default: LT.TEM.Num	Number of measurement records during the passive TEM phase.
nLoops	Default: <infinite>	Total number of longterm measurement cycles to be performed. Each cycle consists of an active THC phase with numTHC measurements and a passive TEM phase with numTEM measurements.
Notes:		
- See the formula above to estimate the duration of a longterm measurement cycle and the number of		

		produced PENEL and MAPPER measurement frames.		
	Telemetry:			
		1 frame	Type=0x7D	MUPUS configuration dump (see Table 44)
		N_{PENEL} frames	Type=0x73	PENEL measurement frames (see Table 38) per longterm cycle
		N_{MAPPER} frames	Type=0x74	MAPPER measurement frames (see Table 39) per longterm cycle

TEM-Mode

The *TEM-Mode* performs passive temperature measurement of the PENEL and the Thermal MAPPER sensors. For the duration of a TEM cycle and the rate of produced passive TEM measurement frames see Formula 4.

There are two ways to start passive TEM measurements:

1. ***[Infinite TEM-Mode]:*** If no number of measurements is specified in the telecommand, then the *Infinite TEM-Mode* runs until any other Mupus-Mode telecommand is received.
2. ***[Finite TEM-Mode]:*** If the number of measurements is specified, then the *TEM-Mode* runs until the specified number of measurements is finished. The execution of another received Mupus-Mode telecommand is postponed until all measurements are completed.

Formula 4: Duration of a single TEM temperature measurement cycle:

$$T_{CycleTEM} \approx \{(interval \times LT.Tick + T_{PENEL} + T_{MAP})\} \approx interval \times 0.1 + 1.3\,\mathrm{sec}(default)$$

$$N_{PENEL} = numTEM / 4 \ \ [\mathrm{frames}]$$

$$N_{MAPPER} = numTEM / 8 \ \ [\mathrm{frames}]$$

71B1 interval [numTEM]				
	Parameters:			
		interval		Passive TEM measurement interval in longterm ticks (LT.Ticks, 100 milliseconds by default).
		numTEM [optional]	Default: <infinite>	Number of passive TEM measurements.
	Notes:			
		- See the formula above to estimate the duration of a TEM measurement cycle and the number of produced PENEL and MAPPER measurement frames.		
	Telemetry:			
		1 frame	Type=0x7D	MUPUS configuration dump (see Table 44)
		N_{PENEL} frames	Type=0x73	PENEL measurement frames (see Table 38)
		N_{MAPPER}	Type=0x74	MAPPER measurement frames (see Table 39)

		frames		

THC-Mode

The *THC-Mode* performs active (with heating) temperature measurement of the PENEL and the Thermal MAPPER sensors. For the duration of a THC cycle and the rate of produced active THC measurement frames see Formula 5.

The same rules for THC quasi-parrallel sensor heating apply as discussed in the longterm mode 0.

There are two ways to start active THC measurements:

1. ***[Infinite THC-Mode]:*** If no number of measurements is specified in the telecommand, then the *Infinite TEM-Mode* runs until any other Mupus-Mode telecommand is received.
2. ***[Finite THC-Mode]:*** If the number of measurements is specified, then the *THC-Mode* runs until the specified number of measurements is finished. The execution of another received Mupus-Mode telecommand is postponed until all measurements are completed.

Formula 5: Duration of a single THC temperature measurement cycle:

$$T_{CycleTEM} \approx interval \times LT.Tick\_THC + T_{PENEL} + T_{MAP} \approx interval \times 0.15 + 1.3\sec$$

$$N_{PENEL} = numTHC/4 \ \ [\text{frames}]$$

$$N_{MAPPER} = numTHC/8 \ [\text{frames}]$$

Note that the tick length for THC is extended by the additional sampling of the +12V current compared to the tick length used in all other modes !

71B2 heatIdx numHeat interval [numTHC]				
	Parameters:			
		heatIdx		Starting index to the heating table LT.HeatTab. If -1 (0xFFFF) is specified, then the starting index is taken from the previously stored CDMS BackupRAM record.
		numHeat		Number of PENEL sensors to heat quasi-simultaneously.
		interval		Active THC measurement interval in longterm ticks (LT.Ticks, 100 milliseconds by default).
		numTHC [optional]	Default: <infinite>	Number of active THC measurements.
	Notes:			
		- See the formula above to estimate the duration of a THC measurement cycle and the number of produced PENEL and MAPPER measurement frames.		
	Telemetry:			
		1 frame	Type=0x7D	MUPUS configuration dump (see Table 44)
		N_{PENEL} frames	Type=0x73	PENEL measurement frames (see Table 38)
		N_{MAPPER} frames	Type=0x74	MAPPER measurement frames (see Table 39)

Mapper-Mode

The *Mapper-Mode* performs temperature measurements of the thermal MAPPER and ANCHOR sensors. For the duration of a MAPPER cycle and the rate of produced MAPPER measurement frames see Formula 6.

There are two ways to start MAPPER measurements:

1. ***[Infinite Mapper-Mode]:*** If no number of measurements is specified in the telecommand, then the *Infinite Mapper-Mode* runs until any other Mupus-Mode telecommand is received.
2. ***[Finite Mapper-Mode]:*** If the number of measurements is specified, then the *Mapper-Mode* runs until the specified number of measurements is finished. The execution of another received Mupus-Mode telecommand is postponed until all measurements are completed.

Formula 6: Duration of a single MAPPER and ANCHOR temperature measurement cycle:

$$T_{CycleMAP} \approx \{numMAP \times (interval \times LT.Tick + T_{MAP})\} \approx \{numMAP \times (interval \times 0.1 + 0.34\} s$$

$$N_{MAPPER} = numMAP / 8 \text{ [frames per cycle]}$$

71B3 interval [numTEM]			
Parameters:			
	interval		MAPPER measurement interval in longterm ticks (LT.Ticks, 100 milliseconds by default).
	numMAP [optional]	Default: <infinite>	Number of MAPPER & ANCHOR measurements.
Notes:			
	- See the formula above to estimate the duration of a MAPPER measurement cycle and the number of produced MAPPER measurement frames.		
Telemetry:			
	1 frame	Type=0x7D	MUPUS configuration dump (see Table 44)
	N_{MAPPER} frames	Type=0x74	MAPPER measurement frames (see Table 39) per MAPPER cycle

CMapper-Mode

The *Mapper-Mode* performs temperature measurements of the thermal MAPPER and ANCHOR sensors using MAPPER calibration input. For the duration of a MAPPER cycle and the rate of produced MAPPER measurement frames see Formula 7.

The MAPPER calibration measurements are always finite with a predefined number of measurement cycles. The execution of another received Mupus-Mode telecommand is postponed until all measurements are completed.

Formula 7: Duration of a single CMAPPER temperature measurement:

$$T_{CMAP} \approx \{(preMAP + calMAP + postMAP) \times (interval \times LT.Tick + T_{MAP})\} \approx \{(nMAP) \times (interval \times 0.1 + 0.34\} s$$

$$N_{CMAP} = (preMap + calMAP + postMAP)/8 \text{ [frames]}$$

71B4 interval preMAP calMAP postMAP				
	Parameters:			
		interval		MAPPER measurement interval in longterm ticks (LT.Ticks, 100 milliseconds by default).
		preMAP		Number of MAPPER/ANCHOR measurements before calibration.
		calMAP		Number of MAPPER/ANCHOR calibration measurements.
		postMAP		Number of MAPPER/ANCHOR measurements after calibration.
	Notes:			
		- See the formula above to estimate the duration of a MAPPER measurement cycle and the number of produced MAPPER measurement frames. Note that TM calibration can as well be performed simultaneously with the execution of normal MAPPER or TEM modes using the SWITCH_MAPPER command !		
	Telemetry:			
		1 frame	Type=0x7D	MUPUS configuration dump (see Table 44)
		N_{CMAP} frames	Type=0x74	MAPPER measurement frames (see Table 39)

Arm-Mode

The *Arm-Mode* telecommand performs the deployment of the Mupus penetrator.

It is a basically heritage from the original Mupus software 4.6 with the difference that the deployment status is reported by means of MUPUS TEXT frames instead of binary data.

71C0 mode parm1 parm2 parm3 parm4					
	Parameters:				
		mode	mode=0	Standard mode - Release the PEN from the balcony - Deploys it ~1 meter from the Lander	
				parm1=	Initial Motor Frequency
				parm2=	Final Motor Frequency
				parm3=	Number of motor rotations
				parm4=	Dummy parameter
			mode=1	Reserve Deploy (with fixed motor frequency)	
				parm1=	Initial Motor Frequency

				parm2=	Number of motor rotations
			mode=2	Only Release	
			mode=3	Only Deploy in the Vacuum Thermal Chamber - with different parameters	
				parm1=	Initial Motor Frequency
				parm2=	Final Motor Frequency
				parm3=	Number of motor rotations
				parm4=	Dummy parameter
	Notes:				
		Note that all 4 parameters must be provided in any case. If a parameter is x0000 then the the corresponding default value from the configuration is used. Calling ARM with default parameters can therefore be done by the telecommand: 71C0 0000 0000 0000 0000 0000 chksum			
	Telemetry:				
		N	Type=0x70	A number of Mupus TEXT frames (Table 35) status reports.	

Gear-Mode

The Gear.mode telecommand performs a check whether the position of the PHILAE lander is allowed for deployment of the MUPUS PEN and, if so, starts the deployment by calling standard ARM mode with the default configuration parameters.

71E0				
	Parameters:			
		<none>		
	Notes:			
		If the Gear-Simulation TC was used before to simulate LG BRAM contents no actual deployment is started if all conditions are ok, instead a text message “Simulated ARM mode started” is issued.		
	Telemetry:			
		N_T	Type=0x70	A number of Mupus TEXT frames (Table 35) status reports.

Hammer-Mode

The *Hammer-Mode* telecommand performs the insertion of the Mupus penetrator.

It is basically heritage from the original Mupus software 4.6 with the difference that the deployment status is reported by means of MUPS TEXT frames and depth sensor frames with additional timing information about hammer strokes.

71C8 mode parm1 parm2 parm3 parm4			
Parameters:			
mode	mode=0	Standard FLIGHT mode Insertion & Rectracting	
	mode=1	Reserve Insertion - Fixed hammer energy and number of hammer strokes	
		parm1=	Hammer energy level 0..3, default=0
		parm2=	Number of hammer 4-strokes, default=200
		parm3=	Dummy parameter
		parm4=	Dummy parameter
	mode=2	Reserve retracting - Penetrator retracting with constant motor frequency	
		parm1=	Motor frequency 200..299, default=200
		parm2=	Number of motor rotations, default=750
		parm3=	Dummy parameter
		parm4=	Dummy parameter
	mode=3	Release (TV-chamber)	
	mode=4	Insertion (TV-chamber) - Fixed hammer energy and number of hammer strokes	
		parm1=	Hammer energy level 1..4, default=2
		parm2=	Number of hammer 4-strokes, default=1
		parm3=	Dummy parameter
		parm4=	Dummy parameter
	mode=5	Retracting (TV-chamber) - Penetrator retracking for TV-test	
		parm1=	Initial motor frequency 200..299, default=200
		parm2=	Final motor frequency 200..2400, default=2400
		parm3=	Number of motor rotations, default=1033
		parm4=	Dummy parameter
	mode=6	Hammer test mode (not implemented)	
	mode=7	Reserve heating	
Notes:			
Note that the mode plus all 4 parameters must be provided in any case in the TC. If a parameter is x0000 then the the corresponding default value from the configuration is used. Calling HAMMER with default parameters can therefore be done by the telecommand: 71C8 0000 0000 0000 0000 0000 chksum			
Telemetry:			
N_T	Type=0x70	A number of Mupus TEXT frames (Table 35) status reports.	

		N_D	Type=0x72	A number of Mupus depth sensor frames (Table 37), N_D = number_of_hammer_4strokes/13

Anchor-Mode

This telecommand starts the *Anchor-Mode* as described in 6.6.

It runs until both Anchors are fired or until the CDMS report SafeAnchoring or UnsafeAnchoring. Anchor accelerator data is produced only when at least one *AnchorFire* interrupt has been detected via the Mupus INT4-line.

The *Anchor-Mode* can be simulated or stopped by the *AnchorStop* no-mode telecommand.

☝ ***For version 7.3 or higher the AnchorStop TC MUST BE provided to initiate the data transfer !***

This procedure gives more flexibility in control of the data transfer which could be advantageous for operations planning especially around touchdown because of the very busy CDMS.

71D0 mode				
	Parameters:			
		<none>		
	Notes:			
		Switch off PENEL and DSB-2 at start of mode to avoid possible disturbances Disable CDMS interrupts in post-trigger phase (for about 0.5-1 s)		
	Telemetry:			
		N_T	Type=0x70	A number of Mupus TEXT frames (Table 35) status reports.
		N_A=276	Type=0x76	276 Anchor accelerator frames per detected *AnchorFire* interrupt. (only for Versions up to V7.2 !, for V7.3ff no telemetry of accelerometer data is produced directly, instead the AnchorStop TC MUST BE used for that purpose !)

AnchorStop

This telecommand stops the *Anchor-Mode* or simulates an Anchor fire.

71D3 mode				
	Parameters:			
		flag	flag=0	Stop anchor mode, send data of both anchor shots stored in RAM pages 2 and 3 (if available)
			flag=1	Simulate Anchor fire (int4), don't stop anchor mode
			flag=2	Stop anchor mode, don't send data
			flag=3	Stop anchor mode, send data of ANC-1 shots stored in RAM page 2 (only Vers. 7.3ff)
			flag=4	Stop anchor mode, send data of ANC-1 shots stored in RAM page 3 (only Vers. 7.3ff)

	Notes:			
		<none>		
	Telemetry:			
		N_A=276/552	Type=0x76	276 Anchor accelerator frames per detected *AnchorFire* interrupt. (only version 7.3ff)

Other telecommands

DumpBRAM

The *DumpBRAM* telecommand reads CDMS Backup-RAM records of any Rosetta-Lander instrument and dumps them with a time stamp to telemetry frames type=0x7C00 (see Table 44).

DumpBRAM is implemented as a no-mode telecommand to be used for monitoring MUPUS-modes.

7024 adr1…adrN				
	Parameters:			
		adr1…adrN		Any number valid CDMS BackupRAM addresses.
	Notes:			
		- BackupRAM records of all Lander instruments can be read. - No validation of the specified BackupRAM addresses is performed. If invalid addresses are specified, a CDMS error message occurs and the corresponding HK value Hk.Cdms.BramRdErr is incremented.		
	Telemetry:			
		1+(N-1)/3 frames	Type=0x7C	A number of BackupRAM frames (see Table 44) are produced, max. 3 BRAM records fit into one telemetry frame.

UploadBRAM

The *UploadBRAM* telecommand writes a single CDMS Backup-RAM records of the MUPUS instrument. Other BRAM addresses outside the MUPUS BRAM address range adr=0x38xx produce CDMS errors.

7025 adr data1…dataN				
	Parameters:			
		adr		A valid CDMS BackupRAM address of the MUPUS instrument.
		data1…dataN		Any number of data words to write
	Notes:			
		- Only valid MUPUS instrument addresses can be written. - No validation of the specified BackupRAM addresses is performed. If invalid addresses are specified, a CDMS error message occurs and the corresponding HK value Hk.Cdms.BramWrErr is incremented.		
	Telemetry:			

		<none>		No telemetry frames are produced

WaitDataComplete

The *WaitDataComplete* telecommand is ROLIS heritage. It checks the MUPUS telemetry buffer every specified number of seconds. If no the telemetry buffer is empty and no telemetry frames have been written to the buffer during this time, Mupus sends an OCPL=10 service request to the CDMS reporting an "Operation Complete". This report could be used to switch off Mupus by the service system.

70F4 mode				
	Parameters:			
		sec		Check interval time in seconds
	Notes:			
		- Sends a OCPL=10 service request to the CDMS		
	Telemetry:			
		<none>		No telemetry frames are produced

TcmdLog

The *TcmdLog* telecommand dumps a log of the last received 125 telecommand words as a single telemetry frame.

71A0 mode				
	Parameters:			
		<none>		
	Notes:			
		- The telecommand log is a ring buffer of plain 125 telecommand words. - Only the **first two words** of each TC are saved - When more than 63 TC's are used the actual contents are overwritten starting with the first TC (e.g. the first TC is replaced by TC No 64 !) - No telecommand length information is provided separating the telecommand messages from each other. - To interpret this information one should know the expected telecommand history of the instrument.		
	Telemetry:			
		1	Type=0x7E	A single telecommand log frame is produced

Technological telecommands

There are a few more telecommands for memory upload and machine code execution. These telecommands are used mainly for ground tests and not explained in detail here.

9 Housekeeping and Science data frames

9.1 MUPUS HK data

Common-DPU vs. MUPUS HK data

The MUPUS-DPU operates in different states with different housekeeping data:

- Common-DPU mode: after power on, 4 x 32 HK values per frame
- MUPUS-4.6b/6.1: after booting MUPUS software version, 4 x 32 HK values per frame
- MUPUS-7.0: after booting MUPUS software version 7.0, 2 x 64 HK values per frame

The Common-DPU state can be identified by HK channel #16 which has the value 0xDEB0 at HK frame word offsets #16, #48, #80, #112.

The MUPUS-4.6b/6.1 state is identified by HK channel #0 equal value 0x07xx (higher byte = 0x07) at HK frame word offsets #0, #32, #64, #96.

The MUPUS-7.0 state is identified by HK channel #0 equal value 0x87xx (higher byte = 0x87) at HK frame word offsets #0, 32,#64 and 96.

- ***However, HK data cannot be safely interpreted during the transition between Common-DPU and MUPUS states.***
- ***The first 32 words of the MUPUS version 7.0 HK block are identical to the version 4.6b/6.1 HK block (with the exception of the ID-byte at offset 0 which is x87 for V7.0 instead of x07 for V4.6).***

That compatibility makes evaluation of existing and upcoming HK data much easier.

MUPUS HK data (Common-DPU state)

Table 23: MUPUS HK data (Common-DPU state)

HK word	Name	Bits	Value	Description
HK#0-15	hkADC			MUPUS ADC channels, see Table 24
HK#16	hkIdent	16	0xDEB0	ComDPU HK identifier
HK#17/18	time	32		ComDPU time after boot in milliseconds (hi,lo), max ~ 49 days
HK#19/20	cdmsTime	32		Last received CDMS lander onboard time (lo,hi)
HK#21	statMsg	16		Counter of received CDMS STAT messages
HK#22	cmdMsg	16		Counter of received CDMS CMD messages
HK#23	datMsg	16		Counter of received CDMS DATA messages
HK#24	savMsg	16		Counter of CDMS messages saved to ComDPU ring buffer
HK#25	debMsg	16		Counter of received DEBx telecommands
HK#26	srErrCount	16		Service Request error counter

HK word	Name	Bits	Value	Description
HK#27	fileStat	16	-1..+2	File Upload status: -1=error, 0=none, +1=running, +2=ready
HK#28	filePtr	16		File upload address pointer
HK#29	fileCount	16		File upload word counter Advanced upload: Telecommand checksum error counter
HK#30/31	hkFree	2x16	0xDEAD	placeholder

Table 24: MUPUS ADC values in Common-DPU state

HK word	AMUX line	Name	Physical		Description
			value	unit	
HK#0	INR1			°C	
HK#1	INR2			°C	
HK#2	INR3			°C	
HK#3	INR4			°C	
HK#4	INR5			°C	
HK#5	INR6			°C	
HK#6	INR7			°C	
HK#7	INR8			°C	
HK#8	INR9			°C	
HK#9	INR10			°C	
HK#10	INR11			°C	
HK#11	INR12			V	
HK#12	INR13			V	
HK#13	INR14			V	
HK#14	INR15			A	
HK#15	INR16	U5P	2.03*3*X/8192	V	voltage of +5V power line of MUPUS-DPU

X = binary value = -8192 ... +8191

MUPUS HK data (version 4.6b/6.1)

The following table is a copy of the original MUPUS flight software manual [1].

Table 25: MUPUS HK data (version 4.6b/6.1)

No	ADR	Signal	Description	Range	Default	Remarks
0a	00	MUPHK0A	MUPUS identifier (byte)	0:FF	07	\ 07 (x87 for Version 7.0 !)
0b	01	MUPHK0B	DPU Status flags (byte) (see Table 27)	0:FF	3C	\ DPU Status (DpuStatusFlags)
1	02	MUPHK1	EPROM Checksum	0:FFFF	-	\ EEPROM Check Sum

No	ADR	Signal	Description	Range	Default	Remarks
2a	04	MUPHK2A	Hk.MupusMode (byte)	0:FF	-	\ Current MUPUS mode (lower byte of 71xx mode telecommand, when mode is running, =0 otherwise)
2b	05	MUPHK2B	CDMS error flags (byte) (see Table 30)	0:FF	00	\ CDMS Error Code (ErrorCodeFlags) Last received CDMS error code; Ver7.1: reset after every HK update
3	06	MUPHK3	On-board Time Low	0:FFFF	-	\ Actual time (LOBT)
4	08	MUPHK4	On-board Time Medium	0:FFFF	-	\ Actual time (LOBT)
5	0A	MUPHK5	RCMD (word1)	0:FFFF	-	\ Last received RCMD command
6	0C	MUPHK6	RCMD (word2)	0:FFFF	-	\ Last received RCMD command
7	0E	MUPHK7	RCMD (word3)	0:FFFF	-	\ Last received RCMD command
8	10	MUPHK8	RSST	0:FFFF	-	\ Last received RSST message
9	12	MUPHK9	RMOD	0:FFFF	083C	\ Last received RMOD message
10	14	MUPHK10	(last) Command Buffer Pointer	0:FFFF	0000	\ CmdPointer word (offset)
11	16	MUPHK11	(last) Backup RAM Pointer	0:FFFF	3800	\ RamPointer word
12	18	MUPHK12	– 12V (current)	0:FFFF	0194	\ Power supply (after power on) Conversion from decimal digital VALUE [DN] to I [mA]: I [mA] = (VALUE-50)*0.01878
13	1A	MUPHK13	– 12V (voltage)	0:FFFF	EAD6	\ Power supply Conversion from decimal digital VALUE [DN] to U [V]: U [V] = (VALUE-50)*2.217/1000
14	1C	MUPHK14	–5V (current)	0:FFFF	01E1	\ Power supply (after power on) Conversion from decimal digital VALUE [DN] to I [mA]: I [mA] = (VALUE-50)*0.01953
15	1E	MUPHK15	–5V (voltage)	0:FFFF	EDE1	\ Power supply Conversion from decimal digital VALUE [DN] to U [V]: U [V] = (VALUE-50)*1.109/1000
16	20	MUPHK16	+5V (current)	0:FFFF	04DC	\ Power supply (after power on) Conversion from decimal digital VALUE [DN] to I [mA]: I [mA] = (VALUE-50)*0.174
17	22	MUPHK17	+5V (voltage)	0:FFFF	11C2	\ Power supply Conversion from decimal digital VALUE [DN] to U [V]: U [V] = (VALUE-50)*1.109/1000
18	24	MUPHK18	+12V (current)	0:FFFF	01C2	\ Power supply (after power on) Conversion from decimal digital VALUE [DN] to I [mA]: I [mA] = (VALUE-50)*0.0888
19	26	MUPHK19	+12V (voltage)	0:FFFF	1532	\ Power supply Conversion from decimal digital

No	ADR	Signal	Description	Range	Default	Remarks
						VALUE [DN] to U [V]: U [V] = (VALUE-50)*2.217/1000
20	28	MUPHK20	PENTS_ASSIST (when PEN-EL switched –on)	0:FFFF	75F0	\ PEN-EL Pt-100 voltage (when PEN-EL switched ON) Conversion from decimal digital value to temperature [deg]: T [deg] =(((VALUE*4000/65536)-20)/19.2 – 100) / 0.392
21	2A	MUPHK21	HK3 (when PEN-EL switched –on)	0:FFFF	75F0	\ Temperature detector TT100 external Sensor not calibrated
22	2C	MUPHK22	PENTS	6000:A000	7650	\ PEN-EL electronics temperature (only valid when PENEL OFF) Conversion from decimal signed digital value to temperature [deg]: T [deg] = ((85.414+VALUE*0.00537-3.65-100)/0.385 Note: temperature uncertainty high (at least 5 K) due to large and temperature dependent wire resistance (compared to R0=100 Ohms), here Rw=3.65 Ohms (@T=-100°C) assumed.
23	2E	MUPHK23	+5V (voltage)	0:FFFF	1986	\ DPU +5V supply Conversion from decimal digital VALUE [DN] to U [V]: U [V] = (VALUE-50)*0.739/1000
24	30	MUPHK24	TM Status Flags (see Table 28)	0:FFFF	0	\ Termal Map. Flags (TmStatusFlags)
25	32	MUPHK25	DD Status Flags (see Table 31)	0:FFFF	0	\ DD & ID Flags (DdStatusFlags)
26	34	MUPHK26	PEN Status Flags (see Table 29)	0:FFFF	0	\ PEN-EL Flags (PenStatusFlags)
27	36	MUPHK27	ANC Status Flags (see Table 32)	0:FFFF	0	\ Anchor Flags (AncStatusFlags)
28	38	MUPHK28	Cdms Error Count	0:FFFF	0	\ CDMS error messages number
29	3A	MUPHK29	TimeOutError	0:FFFF	0	\ MUPUS internal error counter
30	3C	MUPHK30	RES1	6000:A000	951F	\ TM electronics temperature Pt-1000, Conversion from decimal signed digital value to temperature [deg]: T [deg] = ((795+VALUE* 0.0574-1000)/3.85 Voltage drop on common RTN line causes jump by about +5 K when TM is switched ON
31	3E	MUPHK31	MUPUS Status Flags (seeTable 26)	0:FFFF	0:FFFF	\ MUPUS Status (MupusStatusFlags)

The Table 7-14 show the MUPUS sensor/subsystem Status Flags structure.

Table 26: MUPUS Status Flags Word contents (HK[31], MupusStatusFlags)

Bit	Description	Value	Mupus version 7.0 remarks
0	TM calibrated (Yes/Not)	1/0	Set after successful calibration,

Bit	Description	Value	Mupus version 7.0 remarks
			Reset after calibration refused
1	ANC-M 1 sampled (Yes/Not)	1/0	Set after INT4 reception for anchor 1, never reset, same as AncStatusFlags:Bit0
2	ANC-M 2 sampled (Yes/Not)	1/0	Set after INT4 reception for anchor 2, never reset, same as AncStatusFlags:Bit1
3	PEN released (Yes/Not)	1/0	Same as DdStatusFlags:bit3
4	PEN deployed (Yes/Not)	1/0	Set after complete PENEL deployment, same as DdStatusFlags:bit2
5	PEN inserted (Yes/Not)	1/0	Set after complete PENEL insertion, same as DdStatusFlags:bit10
6	DD released (Yes/Not)	1/0	Set after Deployment device release, same as DdStatusFlags:bit0
7	DD retracted (Yes/Not)	1/0	Set after Deployment device retraction, same as DdStatusFlags:bit8
8	Read Alloc. RAM error (Yes/Not)	1/0	<not used for MUPUS version 7.0>
9	Read STCB error (Yes/Not)	1/0	<not used for MUPUS version 7.0>
10	Read Alloc. Mem. Error (Yes/Not)	1/0	<not used for MUPUS version 7.0>
11	MUPUS Setup loaded from BRAM(Yes/Not)	1/0	<not used for MUPUS version 7.0>
12	PEN Set-up loaded from STCB (Yes/Not)	1/0	<not used for MUPUS version 7.0>
13	TM Set-up loaded from STCB (Yes/Not)	1/0	<not used for MUPUS version 7.0>
14	Read BRAM error (Yes/Not)	1/0	<not used for MUPUS version 7.0, see HK#53 instead>
15	Write BRAM error (Yes/Not)	1/0	<not used for MUPUS version 7.0, see HK#54 instead>

Table 27: DPU Status Flags Byte contents (HK[0B], DpuStatusFlags)

Bit	Description	Value	MUPUS Version 7.0 remarks
0	DPU power mode (low/normal)	1/0	Should always be 0
1	Power consumption overflow (Yes/Not)	1/0	Set 1, if at least one of the h/k power supply currents/voltages is outside of range (see below): 0 < -12I < 100 mA -12.5 V < -12U < -10.5 V 0 < -5I < 300 mA -5.4 < -5U < -4.4 V 0 < 5I < 400 mA 4.5 < 5U < 5.5 V 0 < 12I < 300 mA 11.1 < 12U < 13.1 V **No further consequences if out-of-range condition detected (ignored by flight s/w)**
2	Memory page 1 test passed (Yes/Not)	1/0	Dummy parameter, always set to 1 (no Memory check performed)
3	Memory page 2 test passed (Yes/Not)	1/0	Dummy parameter, always set to 1
4	Memory page 3 test passed (Yes/Not)	1/0	Dummy parameter, always set to 1

Bit	Description	Value	MUPUS Version 7.0 remarks
5	Memory Health test passed (Yes/Not)	1/0	Dummy parameter, always set to 1
6	NA	0	
7	NA	0	

Table 28: TM Status Flags Word contents (HK[24], TmStatusFlags)

Bit	Description	Value	MUPUS Version 7.0 remarks
0	Heater (On/Off)	1/0	1 if the TM non-op. heater is working (pre-heating)
1	Low Power Mode (On/Off)	1/0	1 if TM is operating in low power mode
2	Nominal Power Mode (On/Off)	1/0	1 if TM is operating in nominal power mode
3	Calibration Mode (On/Off)	1/0	1 if TM blackbody is heated
4	Last Heat Time > 1h40min (Yes/Not)	1/0	<not used for MUPUS version 7.0>
5	TM powered (Yes/Not)	1/0	1 if TM is powered
6	NA	0	
7	TM Ops. Heater (On/Off)	1/0	1 if TM operational heater On (V7.4ff)
8	Calibration mode (sub-mode) refused (Yes/Not)	1/0	1 if the initial blackbody temperature was too low for calibration (< -110°C) and, therefore, heating of the blackbody for calibration was refused
9-14	NA	0	
15	TM refused (yes/no)	1/0	1 if preheating was not successful (still below switch-ON temperature) and, therefore, TM was not switched ON

Table 29: PEN Status Flags Word contents (HK[25], PenStatusFlags)

Bit	Description	Value	MUPUS Version 7.0 remarks
0	PEN_EL Heater (On/Off)	1/0	1 if the PENEL heater is working (either pre-heating or background heating)
1	Mode Flag (On/Off)	1/0	<not used for MUPUS version 7.0>
2	Mean Flag (On/Off)	1/0	<not used for MUPUS version 7.0>
3	NA	0	
4	Last Heat Time > 1 h (Yes/Not)	1/0	<not used for MUPUS version 7.0>
5	PEN powered (Yes/Not)	1/0	1 if PENEL is powered
6-14	NA	0	
15	PEN refused (Yes/Not)	1/0	1 if preheating was not successful (still below switch-ON temperature) and, therefore, PENEL was not switched ON

Table 30: CDMS Error Flags Word contents (HK[2B], ErrorCodeFlags)

Bit	Description	Value	MUPUS Version 7.0 remarks

Bit	Description	Value	MUPUS Version 7.0 remarks
0	Illegal Request Code (Yes/Not)	1/0	
1	Illegal Unit, Pointer, Offset, Length (Yes/Not)	1/0	
2	Request Code Undue (Yes/Not)	1/0	
3	Mass-memory Full (Yes/Not)	1/0	
4	Allocated SC Data Vol. Exhausted (Yes/Not)	1/0	
5	Destination unit Off (Yes/Not)	1/0	
6	Cdms (TCmd) request error (Yes/Not)	1/0	
7	Cdms (RCmd) request error (Yes/Not)	1/0	

Table 31: Deployment Device Status Flags Word contents (HK[26], DdStatusFlags, version 7.0 only !)

Bit	Description	Value	
0	Hammer release END flag	1/0	Set after Depth Sensor Release
1	Deployment START flag	1/0	Set when starting motor deployment (forward)
2	Deployment END flag	1/0	Set after stopping motor deployment (forward). (same as Mupus status flag, bit 4)
3	Balcony release END flag	1/0	Set after finishing 2nd balcony release (same as Mupus status flag, bit 3)
4	Retracting START flag	1/0	Set when starting motor retraction (backward)
5	NA	0	
6	Balcony release START flag	1/0	Set before starting 1st balcony release
7	Hammer release START flag	1/0	Set when starting Depth Sensor Release
8	Retracting END flag	1/0	Set after stopping motor retraction (backward). (same as Mupus status flag, bit 7)
9	Inserting START flag	1/0	Set when starting insertion (hammering)
10	Inserting END flag	1/0	Set after finishing insertion (hammering) (same as Mupus status flag, bit 5)
11	DSB2 power ON flag	1/0	Set if DSB2 powered, unset if not
12	Penel power ON flag	1/0	Set if PENEL powered, unset if not (same as Penel status flag, bit 5)
13	Penel heating ON flag	1/0	Set if DSB2 powered, unset if not (same as Penel status flag, bit 0)
14	Penel continous heating ON flag	1/0	Set/unset during Penel background heating
15	NA	0	

Table 32: Anchor Status Flags Word contents (HK[27], AncStatusFlags)

Bit	Description	Value	MUPUS Version 7.0 remarks
0	ANCM1 sampled (Yes/Not)	1/0	1 if accelerometer ANCM1 was sampled (since MUPUS ON)
1	ANCM2 sampled (Yes/Not)	1/0	1 if accelerometer ANCM2 was sampled (since MUPUS ON)
2	NA	1/0	

Bit	Description	Value	MUPUS Version 7.0 remarks
3	NA	1/0	
4	NA	1/0	
5	Anchor 1 powered (Yes/Not)	1/0	1 if ANC-1 electronics is powered
6	Anchor 2 powered (Yes/Not)	1/0	1 if ANC-2 electronics is powered
7-15	NA	0	

MUPUS HK data (version 7.0)

The first 32 words of the MUPUS version 7.0 HK block are identical to the version 4.6b/6.1 HK block. The following 32 words are additional HK data valid only for software version 7.0.

Table 33: MUPUS HK data (version 7.0)

HK word	Name	Bits	Value	Description
HK#0-31	hkVersion46	32x16		MUPUS version 4.6b/6.1 HK data, see Table 25
HK#32	MupusId70	16	0x873C	MUPUS version 7.0 identifier, for 64-word HK block
HK#33-34	RefTime	32		MUPUS reference time for LOBT in milliseconds, 32 bits (hi,lo)
HK#35	regStatG0	16		G-BUS-0 register status
HK#36	regStatG1	16		G-BUS-1 register status
HK#37	regStatG2	16		G-BUS-2 register status
HK#38	regMotCPMS	16		Motor rotation counter
HK#39	Spare	16		Unused
HK#40	SentFrames	16		Total number of sent telemetry frames
HK#41	BufferedFrames	16		Current number of buffered telemetry frames
HK#42	RejectedFrames	16		Total number of rejected telemetry frames (buffer full)
Version 7.1 only:				
HK#43	TcmdReceived	16		Total number of received telecommand messages (RCMD)
HK#44	TcmdExec	16		Executed telecommand counter
HK#45	TcmdError	16		Telecommand error counter, checksum/tcmdCode
Version 7.0 only:				
HK#43-45	Spare	3x16		Unused
HK#46-47	Spare	2x16		Unused
HK#48	Temp.Res2	16		Res2 MUPUS-TM 2.5V reference voltage Conversion from decimal digital VALUE [DN] to U [V]: U [V] = (VALUE-50)*0.3662/991

HK word	Name	Bits	Value	Description
HK#49	Temp.Anchor1	16		Anchor 1 temperature measurement, updated only during longterm measurements (Hk.MupusMode=0xB0..0xB4) Conversion from decimal digital VALUE [DN] to U [V] and T [deg]: U [V] = (VALUE-50)*0.3662/991 T [deg] = -22.9+62.5*U+0.825*U^2
HK#50	Temp.Anchor2	16		Anchor 2 temperature measurement, updated only during longterm measurements (Hk.MupusMode=0xB0..0xB4) Conversion from decimal digital VALUE [DN] to U [V]: U [V] = (VALUE-50)*0.3662/991 T [deg] = -22.9+62.5*U+0.825*U^2
HK#51-52	Spare	2x16		Unused
HK#53	Cdms.BramRdErr	16		Backup-RAM read error counter
HK#54	Cdms.BramWrErr	16		Backup-RAM write error counter
HK#55	Cdms.ChksumErr	16		Counter of CDMS telemetry frame checksum errors
Hk#56-63	MupusErrors	16x8	0..255	Byte counters for 16 error codes=0..15, see Table 12

9.2 Science data frames (telemetry)

General data frame structure

MUPUS science data are packed into science data frames of 128 words. The first word defines the data source (MUPUS) and the frame type. The second word is a frame counter, running separately for each frame type. A global frame counter is added by the CDMS. Maximum 124 data words are following. The last word of the data frame is a 16-bit checksum calculated by the following formula:

Checksum = – 1 – (word1 + word2 + … + word127) mod 65536

Internally this checksum makes it very easy to verify onboard communication between MUPUS and the CDMS. After receiving a complete frame - usually in 4 blocks of 32 words – the CDMS send a RSCS message with the an overall 16-bit sum of all received data words. With the MUPUS checksum included the CDMS checksum should always be the same:

RSCS = 0xFFFF

If a different RSCS is returned by the CDMS, then there must be a communication problem between MUPUS and the CDMS. The number of those errors is included in housekeeping data (Cdms.ChksumErr).

Table 34: General science data frame structure

Source	Word number	Bits	Field	Value	Description
MUPUS	1.Word	bit15..12	MUPUS-ID	=7	MUPUS science data identifier
(**5**tss)		bit11..8	TYPE	=0..15	MUPUS frame type

Source	Word number	Bits	Field	Value	Description
		bit7..0	SUBTYPE	=0..255	MUPUS frame subtype
	2.Word	bit15..0	TCOUNT		MUPUS frame counter since power-on-reset (separate for each frame TYPE)
	3.-127.Word		DATA		MUPUS data (content depends on frame TYPE, SUBTYPE)
	128. Word		CHKSUM		MUPUS frame checksum = -1 -word1-…-word127

Text frames (type=0x70)

Table 35: MUPUS ASCII text frames (TYPE=0x70)

	Offset word:bit	Size	Field	Value	Description
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x70	MUPUS text frame
	0:8	1 Byte	SUBTYPE	=0	
	1:0	1 Word	TCOUNT		MUPUS frame counter for TYPE=0x70
	2:0	250 Bytes	TEXT		Plain ASCII text
	127:0	1 Word	CHKSUM		MUPUS frame checksum

MUPUS heating frames (type=0x71)

Table 36: MUPUS heating frame (TYPE=0x71)

	Offset word:bit	Size	Field	Value	Description
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x71	MUPUS heating frame
	0:8	1 Byte	SUBTYPE	=0	
	1:0	1 Word	TCOUNT	=0..65535	MUPUS frame counter for TYPE=0x71
	2:0	1 Word	Heat.PenOnLimit	=	PENEL power-on temperature (PENTS sensor)
	3:0	1 Word	Heat.PenLowLimit	=?	PENEL working temperature (PENTS_ASSIST sensor), lower limit
	4:0	1 Word	Heat.PenHiLimit	=?	PENEL working temperature (PENTS_ASSIST sensor), upper limit
	5:0	1 Word	Heat.TmOnLimit	=	MAPPER power-on temperature (RES1

	Offset word:bit	Size	Field	Value	Description
					sensor)
	6:0	1 Word	Heat.MaxLoops	=?	Max. number of power-on heating loops
Heating record #1					
	7:0	1 Word	Heat.Id	=	0x01,0xF1 Penel pre-heating (PENTS); =0xF1 for the last PENTS record at Penel power-on 0x02,0xF2 Mapper pre-heating (RES1); =0xF2 for the last RES1 record at Mapper power-on 0x03,0xF3 PENEL continuous heating (PENTS_ASSIST); =0xF3 when Penel heater on 0x00 No more data in frame
	8:0	2 Words	MUPUS time		MUPUS time in milliseconds
	10:0	1 Word	Heat.Temp		PENTS, RES1 or PENTS_ASSIST temperature
	11:0	4 Words Bytes	Heating record #2		2nd heating record
	…		...		…
	123:0	4 Words	Heating record #30		Last heating record
	127:0	1 Word	CHKSUM		MUPUS frame checksum

MUPUS depth sensor (hammering) frames (type=0x72)

Table 37: MUPUS depth sensor frame (TYPE=0x72)

	Offset word:bit	Size	Field	Value	Description
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x72	MUPUS depth sensor frame
	0:8	1 Byte	SUBTYPE	=0	
	1:0	1 Word	TCOUNT	=0..65535	MUPUS frame counter for TYPE=0x72
	2:0	1 Byte	MupusMode	=0xC8 (hammering)	Current MUPUS mode: Should be =0xC8 for hammering (=MupHK2a)
	2:8	1 Byte	ErrCode	=?	CDMS error codes (=MupHK2b)
	3:0	2 Words	LOBT	=	Last received Lander onboard time
	5:0	2 Words	MUP-TIME	=	MUPUS reference time for LOBT
	7:0	1 Word	MupusStat	=?	MUPUS status flags

	Offset word:bit	Size	Field	Value	Description
	8:0	1 Byte	MupusId	=x87	MUPUS ID=x87 for version 7.X
	8:8	1 Byte	DPU.StatFlags	=?	MUPUS DPU status flags
	9:0	1 Word	DepthRef	=?	Reference depth measure at begin of insertion
	10:0				Hammer record #1
	10:0	1 Word	#4Stroke	=?	Counter of hammer cycles, 4 strokes each
	11:0	1 Byte	Energy	=0..3	Hammer energy level
	11:8	1 Byte	nSaf	=0	Hammer cycles at this energy (mod 256)
	12:0	2 Words	Time		MUPUS time of starting hammer cycle
	14:0	4 Words	TimeDiff		Time differences (msec) for 4 hammer strokes
	18:0	1 Word	DepthVal		Current depth value measurement
	…	9 Words			More Hammer records
	118	9 Words			Hammer record #13
	127:0	1 Word	CHKSUM		MUPUS frame checksum

PENEL data frames (type=0x73)

Table 38: MUPUS PENEL measurement frames (TYPE=0x73)

	Offset word:bit	Size	Field	Value	Description
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x73	PENEL measurement frame
	0:8	1 Byte	SUBTYPE	=0x00..02	Penel measurement mode: 0=Longterm; 1=TEM, 2=THC
	1:0	1 Word	TCOUNT	=0..65535	MUPUS frame counter for TYPE=0x73
PENEL header					
	2:0	1 Word	ThcCount	=0..4	Number of THC records in this frame (redundant information from 4x Rec.HeatFlags used for convenience)
	2:0	2 words	ThcStartTime		MUPUS time (millisec) when this heating (THC) cycle has been started
	4:0	2 Words	Spare		<unused>
PENEL data record #1					
	7:0	1 Word	Rec.Count		Penel record counter, reset for each new mode telecommand
	8:0	2 Words	Rec.Time		MUPUS time in milliseconds

	Offset word:bit	Size	Field	Value	Description
	10:0	1 Word	Rec.PowerFlags		Power status flags
	11:0	1 Word	Rec.HeatFlags	=0 for TEM <>0 for THC	16 Flags which Penel temperature sensors are heated (Bit0 corresponds to sensor#0)
	12:0	1 Word	Rec.Spare		<unused>
	13:0	16 Words	Rec.PenTemp		Penel temperature sensor values
	29:0	8 Words	Rec.PenHK		Penel HK sensor values
	37:0	30 Words	Penel record #2		2nd Penel data record
	…		...		…
	97:0	30 Words	Penel record #4		4th Penel data record
	127:0	1 Word	CHKSUM		MUPUS frame checksum

THERMAL MAPPER data frames (type=0x74)

Table 39: MUPUS MAPPER measurement frames (TYPE=0x74)

	Offset word:bit	Size	Field	Value	Description
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x74	MAPPER measurement frame
	0:8	1 Byte	SUBTYPE	=0x00..04	Mapper measurement mode: 0=Longterm; 1=TEM, 2=THC, 3=MAPPER, 4=CMAPPER
	1:0	1 Word	TCOUNT	=0..65535	MUPUS frame counter for TYPE=0x74
MAPPER header					
	2:0	5 Words	Spare		<unused>
MAPPER data record #1					
	7:0	1 Word	Rec.Count		Mapper record counter, reset for each new mode telecommand
	8:0	2 Words	Rec.Time		MUPUS time in milliseconds
	10:0	1 Word	Rec.PowerFlags		Power status flags
	11:0	9 Words	Rec.Mapper		Mapper sensor values
	20:0	2 Words	Rec.Anchor		Anchor sensor values
	22:0	15 Words	Mapper record #2		Mapper data record
	…		...		…
	112:0	15 Words	Mapper record #8		Mapper data record

	Offset word:bit	Size	Field	Value	Description
	127:0	1 Word	CHKSUM		MUPUS frame checksum

THC-Power data frames (type=0x75) (Version ≥7.3 only)

Table 40: MUPUS THC-power measurement frames (TYPE=0x75)

	Offset word:bit	Size	Field	Value	Description
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x75	THC-power measurement frame (Version ≥7.3 only)
	0:8	1 Byte	SUBTYPE	=0x00, 0x02	measurement mode: 0=Longterm; 2=THC
	1:0	1 Word	TCOUNT	=0..65535	MUPUS frame counter for TYPE=0x75
THC-Power header					
	2:0	1 Word	IdxHeat		Heater start index, taken from TC (if provided) or Config
	3:0	1 Word	Numheat		Number of heaters simultaneously heated, taken from TC (if provided) or Config
	4:0	1 Word	THCInterval		Time interval (ticks) with THC heating active between 2 temperature scans, taken from TC (if provided) or Config
	5:0	1 Word	LT.THCHeatPause		Number of additional ticks (after each heated tick) without heating, if > 0 the total heating power is reduced, taken from Config
	6:0	1 Word	LT.Tick		Tick length in [ms], from Config (note that THC ticks for Vers 7.3ff are effectively longer because of additional 12V current measurements !)
THC-Power data record #1					
	7:0	1 Word	Rec.Count		THC-power record counter, reset for each new mode telecommand
	8:0	2 Words	Rec.Time		MUPUS time in milliseconds (Time of the last current/voltage measurement)
	10:0	1 Word	Rec.Voltage+12V		Last measured voltage +12V
	11:0	16 Words	Rec.Current+12V		THC sensor heating currents for all 16 heater sensors (=0 if sensor not heated)
	27:0	20 Words	THC-power record #2		Next THC-power data record

	Offset word:bit	**Size**	**Field**	**Value**	**Description**
	…		...		…
	107:0	20 Words	THC-power record #8		Last THC-power data record
	127:0	1 Word	CHKSUM		MUPUS frame checksum

ANCHOR accelerometer measurement frames (type=0x76)

Table 41: ANCHOR accelerometer measurement frames (TYPE=0x76)

	Offset word:bit	**Size**	**Field**	**Value**	**Description**
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x76	MAPPER measurement frame
	0:8	1 Byte	SUBTYPE	=0x01..02 =0x11..12	Anchor number 1=Main, 2=Redundant Test Anchor mode (Tcmd=0x707D) frames for Main/Redundant anchor
	1:0	1 Word	TCOUNT	=0..65535	MUPUS frame counter for TYPE=0x76
ANCHOR header					
	2:0	1 Word	Cdms.SST		CDMS service system status at the begin of anchor measurements
	3:0	1 Word	PreTriggerLimit	=1..30000	Number of pre-trigger measurements to store, value from configuration (typical value ~3000 = 10% of data)
	4:0	2 Words	Int4.Time		Mupus time in milliseconds, when INT4 (anchor fire) has been received
	6:0	1 Word	Post.Time	=400..1000	Milliseconds measurement time in post-trigger phase, total number of measurements = 32768 - PreTriggerLimit
	7:0	1 Word	Index	<signed>	Starting index of first measurement in frame: -PreTriggerLimit…-1 Pre-trigger phase 0…32767-PreTriggerLimit Post-trigger phase

	Offset word:bit	Size	Field	Value	Description
ANCHOR measurement data					
	8:0	119 Word	Values		Anchor accelerometer measurement values
	127:0	1 Word	CHKSUM		MUPUS frame checksum

Although originally intended to be used only for test purposes, problems encountered during the GRM SDL campaign in 2010 led the MUPUS team to decide to use the test (or "time-stamped") variant of the anchor mode also for the real SDL at CG-67. This variant of the ANCHOR mode is used when the additional TC x707D x8F83 was sent before ANCHOR mode is commanded. In this case the anchor measurement subsection of the ANCHOR frames looks different because each accelerometer value is accompanied by a 16-Bit timer counter which can be used to exactly reconstruct the time of sampling with a resolution of 0.2 µs.

☝ ***Test or "time-stamped"ANCHOR mode shall be used for the real SDL in 2014 !***

Table 42 ANCHOR accelerometer measurent frames for test ("time-stamped") mode (TYPE=0x7611 or 0x7612)

					SUBTYPE =0x11..12 only !
		1 Word	Timestamp record #1		16 Bit timercounter, 1 count = 0.2 µs
		1 Word	Value record #1		Anchor accelerometer measurement values
		2 Words	Record #2		
	…		ANCHOR data record		
	124:0	2 Words	8:0		
	126:0	1 word	9:0		
	127:0	1 Word	10:0		MUPUS frame checksum

ADC data frames (type=0x7A)

Table 43: MUPUS ADC measurement frame (TYPE=0x7A)

	Offset word:bit	Size	Field	Value	Description
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x7A	ADC measurement frame
	0:8	1 Byte	SUBTYPE	=0x00..02	Raw ADC mode: 0="Soft"; 1/2="Hard" single/multi-channel; see RawAdc telecommand

	Offset word:bit	**Size**	**Field**	**Value**	**Description**
				=0x10..0x14	Average ADC measurements PENEL correction mode=0..2; see AverageADC telecommand
	1:0	1 Word	TCOUNT	=0..65535	MUPUS frame counter for TYPE=0x7A
ADC header					
	2:0	1 Word	Average	=0..32767	Number of values to average
	3:0	1 Word	Delay1	=0..65535	Delay in milliseconds/microseconds
	4:0	1 Word	Delay2	=0..65535	Delay in milliseconds/microseconds
	5:0	1 Word			Placeholder
	6:0	1 Word	End-Marker	=0,0xFFFF	0=continue; 0xFFFF=End of Raw ADC mode
ADC data record #1					
	7:0	1 Byte	Channel	=0x00..0x37	0x00..0x0F Common-DPU channel, see Table 5 0x10..0x1F Mupus MUX channel, cascaded at DPU channel 13, see Table 6 0x20..0x37 PENEL channel 1..24, see Table 7
	7:8	3 Bytes	MUPUS time		MUPUS time in milliseconds without most significant byte
	9:0	1 Word	MUPUS ticks	=0..4999	MUPUS-DPU clock ticks (1 tick = 0.2 µsec)
	10:0	1 Word	Raw ADC value		Raw ADC measurement value
	11:0	8 Bytes	ADC record #2		Channel, time, ticks, ADC value
	…		...		…
	123:00	8 Bytes	ADC record #30		Channel, time, ticks, ADC value
	127:0	1 Word	CHKSUM		MUPUS frame checksum

BackupRAM dump (type=0x7C)

Table 44: CDMS BackupRAM dump (TYPE=0x7C)

	Offset word:bit	Size	Field	Value	Description
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x7C	MUPUS configuration dump
	0:8	1 Byte	SUBTYPE	=00	

	Offset word:bit	Size	Field	Value	Description
	1:0	1 Word	TCOUNT	=0..65535	MUPUS frame counter for TYPE=0x7C
BRAM record #1					
	2:0	2 Words	BramLOBT	=	Last received Lander onboard time
	4:0	2 Words	BramTime	=	MUPUS time for BackupRAM dump
	6:0	1 Word	BramAddr	=	Lander device and BRAM buffer address
	7:0	32 Words	BramBuf	=	CDMS BackupRAM buffer
	39:0	37 Words	BRAM record #2		Optional 2nd BackupRAM record
	76:0	37 Words	BRAM record #3		Optional 3rd BackupRAM record
	113:0	14 words	unused		
	127:0	1 Word	CHKSUM		MUPUS frame checksum

Configuration dump (type=0x7D)

Table 45: MUPUS configuration dump (TYPE=0x7D)

	Offset word:bit	Size	Field	Value	Description
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x7D	MUPUS configuration dump
	0:8	1 Byte	SUBTYPE	=0F	
	1:0	1 Word	TCOUNT	=0..65535	MUPUS frame counter for TYPE=0x7D
Software compilation date, time, version					
	2:0	1 Word	Comp.Year	>=2006	Year, e.g. 0x07D6=2006
	3:0	1 Byte	Comp.Month	=1..12	Month of year
	3:8	1 Byte	Comp.Day	=1..31	Day of month
	4:0	1 Byte	Comp.Hour	=0..23	Hour
	4:8	1 Byte	Comp.Min	=0..59	Minute
	5:0	1 Byte	Comp.Sec	=0..59	Second
	5:8	1 Byte	Comp.Frac	=0..99	Fraction of second, 1/1000 second
	6:0	1 Word	SW.Version		Software version, e.g. 0x0701 = version 7.01
	7:0	1 Word	spare	=0	<unused>
Mupus software configuration					
	8:0	119 Words	CONFIG		Configuration settings, see 5.1
	127:0	1 Word	CHKSUM		MUPUS frame checksum

Telecommand log frames (type=0x7E)

Table 46: MUPUS telecommand log (TYPE=0x7E)

	Offset word:bit	Size	Field	Value	Description
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x7E	MUPUS telecommand log
	0:8	1 Byte	SUBTYPE	=0x00..0xFF	= current index in telecommand log ring buffer
	1:0	1 Word	TCOUNT	=0..65535	MUPUS frame counter for TYPE=0x7E
	2:0	125 Words	TELECMD		Telecommand data
	127:0	1 Word	CHKSUM		MUPUS frame checksum

Memory dump frames (type=0x7F)

Table 47: MUPUS memory page dump (TYPE=0x7F)

	Offset word:bit	Size	Field	Value	Description
MUPUS					
	0:0	1 Byte	ID-TYPE	=0x70	MUPUS text frame
	0:8	1 Byte	SUBTYPE	=0..8	RAM-page
	1:0	1 Word	TCOUNT	=0..65535	MUPUS frame counter for TYPE=0x7F
	2:0	1 Word	ADR	=0x0..0xFFFF	RAM page address
	3:0	124 Words	DATA		Dumped data words
	127:0	1 Word	CHKSUM		MUPUS frame checksum

10 MUPUS Calibration

10.1 PEN

Table 48 PENEL MUX channels and calibration coefficients

PENEL Channel number	MUPUS name(s)	R_0 [Ohms]	α [K^{-1}]	L_{sens} [mm]	Description
1	R1	90.642	0.002759	9.1	PEN sensor 1, d=4.57 mm* (top)
2	R2	100.62	0.002619	9.9	PEN sensor 2, d=14.01 mm
3	R3	87.783	0.002812	11.0	PEN sensor 3, d=24.39 mm
4	R4	68.002	0.003074	12.2	PEN sensor 4, d=36.04 mm
5	R5	72.597	0.002879	13.3	PEN sensor 5, d=48.70 mm
6	R6	76.979	0.002826	14.8	PEN sensor 6, d=62.73 mm
7	R7	95.393	0.002609	16.2	PEN sensor 7, d=78.33 mm
8	R8	81.960	0.002698	17.5	PEN sensor 8, d=95.14 mm
9	R9	75.365	0.002865	19.1	PEN sensor 9, d=113.30 mm
10	R10	76.900	0.002804	21.0	PEN sensor 10, d=133.27 mm
11	R11	78.648	0.002800	23.5	PEN sensor 11, d=155.40 mm
12	R12	75.197	0.002860	26.0	PEN sensor 12, d=180.24 mm
13	R13	82.212	0.002783	28.1	PEN sensor 13, d=207.47 mm
14	R14	83.971	0.002786	30.5	PEN sensor 14, d=236.45 mm
15	R15	85.698	0.002737	34.3	PEN sensor 15, d=268.50 mm
16	R16	93.085	0.002655	39.2	PEN sensor 16, d=305.14 mm (bottom)
17	HK1				Short circuit/0 Ohm reference #1 for R1-R8
18	HK2	100	0.00385		PT100 holder inset (outside)**
19	HK3				TT100 holder ring (outside), not calibrated
20	HK4, PENTS_ASSIST	100	0.00385		PT100 detector, temperature control sensor inside PENEL
21	HK5				TT100 holder inside, not calibrated
22	HK6, R100				100 Ohm reference resistor
23	HK7				Short circuit/0 Ohm reference #2 for R9-R16
24	HK8, R20				20 Ohm reference resistor

*d=distance to centre of sensor measured from top of sensor foil

**sensor inside holder for Flight Spare

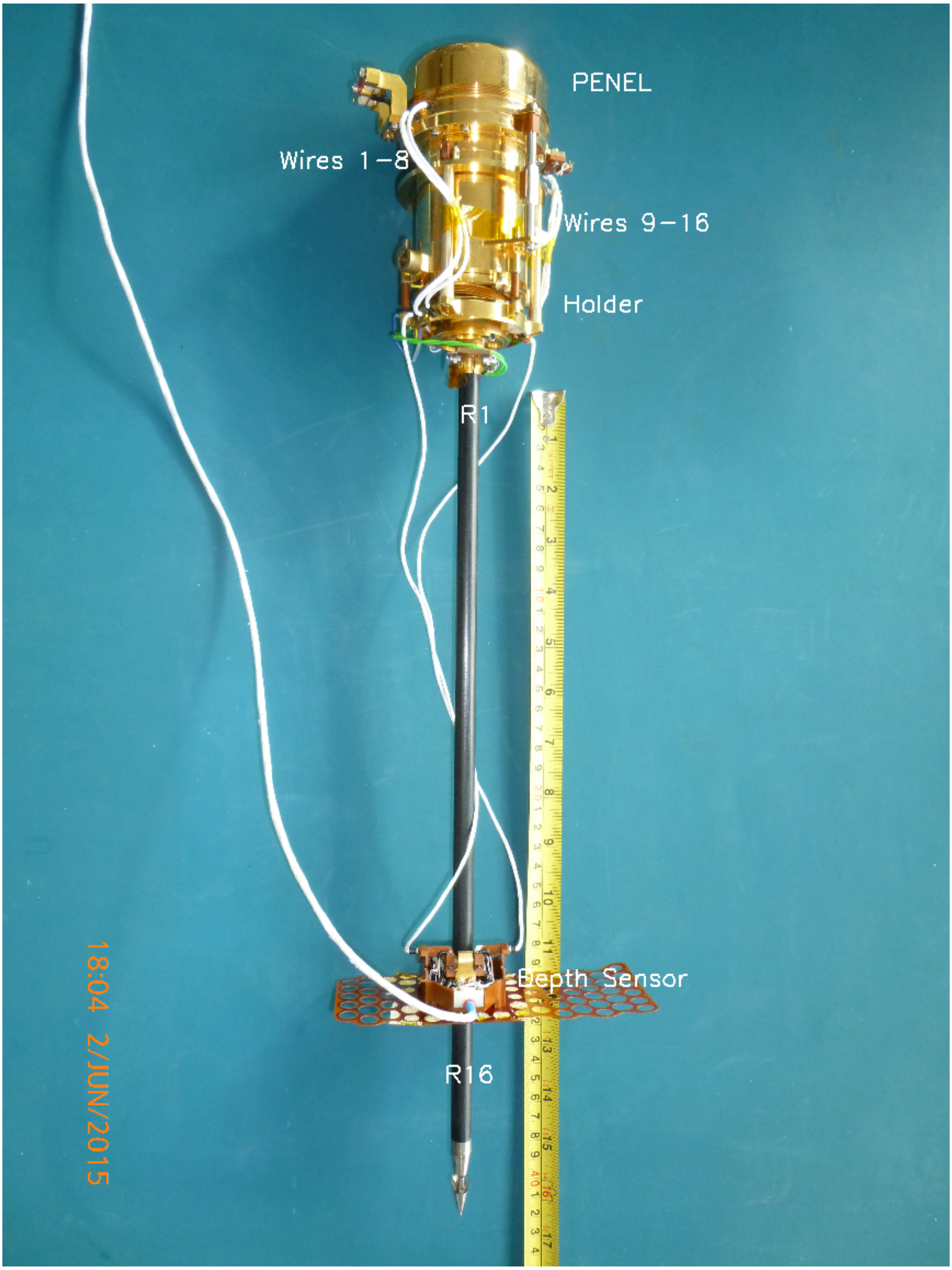


Figure 10-1 MUPUS PEN with hammering mechanism and PENEL on top.

MUPUS PEN description

The MUPUS PEN is shown in Figure 10-1. On top of the black penetrator rod of ~ 370 mm length and 10 mm diameter a cylindrical housing contains the hammer mechanism and a front end electronics (PENEL) which controls the readout of the thermal measurements. Inside the penetrator tube 16 temperature sensors R1,…,R16 are mounted where R1 is located on top and R16 on the bottom. The MUPUS PEN sensors are realized as Ti-Resistors sputtered on a Kapton foil and belong to the class of RTD-type (Resistance Thermometer Device, like Pt-100) temperature sensors. These sensors show a nearly linear dependence of resistance on temperature with a typical quadratic term of only ~2 10^{-7} K^{-2} (as determined with the PEN FS, about 1/3 of the non-linearity of a Pt-100) . The sensor foil was then rolled up and glued to the inner wall of the rod. The dependence between measured resistance R and sensor temperature T was individually calibrated for each sensor. The resistance is measured in 2-wire configuration be applying a “constant” current to the sensors and measure the corresponding voltage drop using a 16-Bit ADC. The sensor signals are routed from the top of the sensor foil via internal wires to the PENEL holder where they are soldered to two external cables with 8 wires each (see Figure 10-1) which connect the PEN sensors to PENEL.

PEN calibration

During each temperature scan performed in the MUPUS modes TEM, THC, and LONGTERM 24 PENEL MUX channels are sampled, 16 PEN sensors and 8 internal housekeeping values. The h/k channels comprise of 2 fixed reference resistors located inside PENEL (channel 22 and 24), 2 shortcuts located inside the holder to determine part of the wire resistances (17 and 24), and 4 RTD-type sensors (Table 48). Two of the RTDs are Ti-sensors (TT100) manufactured with a technology similar to the foil sensor and located inside the PENEL holder which were not calibrated, the others are Pt-100 which measure the PENEL temperature (h/k-4) and the holder temperature (h/k-2).

The recorded digital numbers DN can be converted into raw voltages by the relation U=4*DN/(2^{16}-1) . The resistance measurement is affected by two disturbing effects. The first is that the current source (inside PENEL) shows significant temperature and self-heating effects, the other is that the input buffer amplifier circuit generates an additional offset voltage which is dependent on the measured voltage itself. To remove these disturbances during every PEN temperature scan additionally two fixed resistors R_{20} and R_{100} with values of 20.18+/-0.05 Ohms and 99.87+/-0.05 Ohms which are located inside PENEL are also measured. Note that these values for the reference resistors include a contribution of 0.17 Ohms from the track resistances in the PCB computed for the nominal operation PENEL temperature of -48°C. Since the offset voltage dependence on the input value is nearly linear it can well be approximated by:

$$U_{off} = \frac{U_{20} - U_{100} * r_q}{1 - r_q} \quad (1)$$

where U_{20} is the voltage drop over R_{20}, U_{100} is the voltage drop over R_{100} and $r_q = R_{20}/R_{100}$. The corrected resistance of each PEN sensor $R_{i,l}$ (i=1,24, still including the line resistance !) can then be written as:

$$R_{i,l} = R_{100} \frac{U_i - U_{off}}{U_{100} - U_{off}} \quad (2)$$

where U_i is the voltage drop across the i-th PEN channel.

For the removal of the line resistances two different parts have to considered. The first part comprises the tracks on the PENEL PCB’s plus the wires connecting PENEL with the PEN holder (just on top of the PEN tube). Inside the PENEL holder there are two shortcuts realized (one for each sensor wire connecting PENEL with the holder, wire 1 consisting of sensors PEN 1-8, wire 2 of PEN 9-16.) which are also measured during each temperature scan. Therefore, by subtracting this value from $R_{i,l}$ the influence of this part of the sensor wires on the measured value can be removed.

$$R_{i,c} = R_{i,l} - R_{17} = 1{,}8\,; \qquad R_{i,c} = R_{i,l} - R_{24} = 9{,}16 \quad (3)$$

The part of the copper line resistance due to wires connecting the holder and the foil and the tracks on the sensor foil itself is dependent on the temperature and can only be removed in an iterative manner as described in detail below.

The MUPUS Ti-RTD calibration can be described by the following relation between the resistance of sensor i R_i and its temperature T_i (i=1,..16) :

$$R_i(T) = R_{0,i}(1 + \alpha_i (T_i - T_0)) \quad \text{or} \quad T_i = T_0 + \frac{R_i(T) - R_0}{\alpha R_0} \tag{4}$$

where R_0 is the sensor resistance at the temperature T_0 (here T_0=-100°C) and α is the temperature coefficient of resistance (see Table 48). The calibration coefficients were finally updated after the inflight calibration carried out during Post-Hibernation-Commissioning and are given in Table 48.

The remaining part of the line resistances for sensor i can be written as:

$$R_{w,i} = \frac{\rho(T_H)L_H}{A_H} + \rho(T_1)(L_0/A_w + L_0/A_{Com}) + \frac{\sum_1^i \rho(T_i)L_i}{A_w} + \frac{\sum_1^i \rho(T_i)L_i}{A_{com}} \tag{5}$$

where $\rho(T)$ = 1.78 10^{-8} + 0.0039*(T-T_{20}) + 6 10^{-7}*(T-T_{20})^2 is the temperature dependent specific resistivity of the copper tracks (T in [K], T_{20}=293.15 K), L_H=0.12 m is the length of the AWG36 wires connecting the top of the sensor foil with the holder, A_H=1.267 10^{-8} m^2 is the cross section of these wires, L_i is the length of sensor i (from Table 48), L_0 is the length of 17 mm from top of the foil until the first sensor begins, A_w is the cross section of the tracks connecting each sensor 1,..,16 on the Kapton foil (300x17 µm), and A_{com} (800x17 µm) is the cross section of the common return line. It is assumed here that the uppermost part of the foil is at the same temperature T_1 as the first sensor.

By inserting the already computed $R_{i,c}$ values from Eq. 3 into Eq. 4 a first approximation of the sensor temperatures can be computed. Now inserting these temperatures into Eq. 5 an estimate for the wire resistance is calculated and subtracted from $R_{i,c}$ to get a better estimate of the true sensor resistance. Then Eq.4 is used again to get an update of the temperature values. This procedure is then repeated a few times until the maximum change in sensor temperature of all 16 sensors is less than 10 mK. Usually 3-4 iteration steps are sufficient for convergence.

10.2 TM

MUPUS TM description

MUPUS TM is a 4 channel infrared radiometer using thermopiles as detector elements. The main characteristics of the 4 TM wavelength channels are given in Table 49. Furthermore, TM is equipped with an inflight calibration unit which is realized by a cylindrical blackbody which can be heated up to 390 K and covers about 1% of the sensor FOV.

The MUPUS TM is mounted with two struts made from PEEK to the horizontal "V-strut" of the Philae lander, and is, therefore, thermally well isolated from the Lander structure. The sensorhead is coated by IRIDITE-11-2 with an emissivity of 0.2 and a solar absorptance of 0.4,

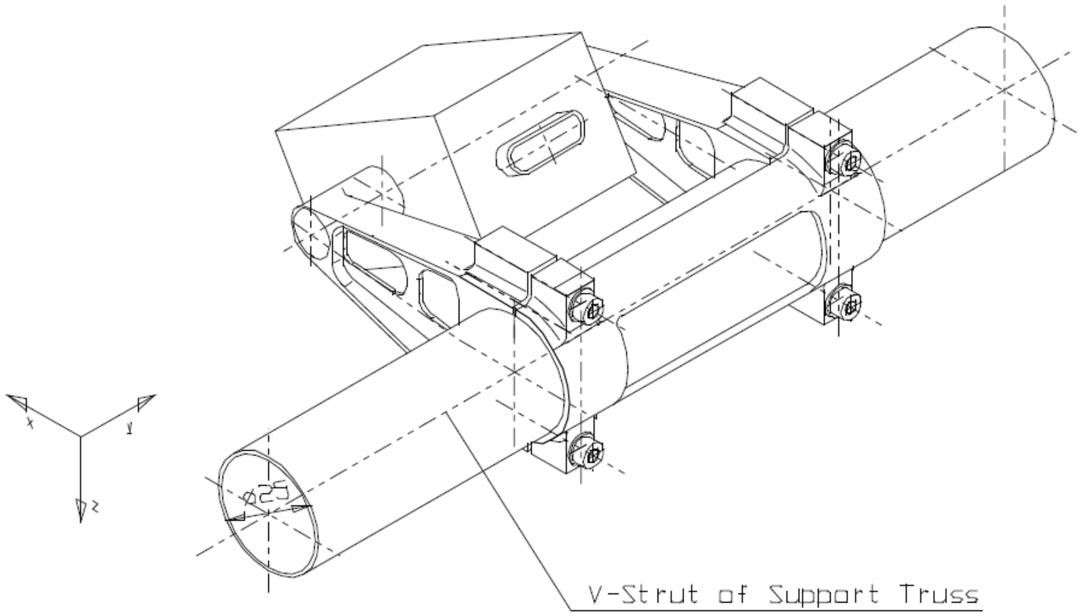


Figure 10-2 Mechanical I/F of MUPUS TM to Lander structure. The TM sensor head is mounted thermally isolated to the V-strut of the Lander by two struts made from PEEK.

The MUPUS TM location in the Lander RF as well as the line-of-sight are given in the following table. The sensor head is looking downward with an elevation of -45° w.r.t. the X-Y plane of the Lander. The sensor head is mounted below the Tx/Rx antenna plate which acts as a Sun shield for TM. But it should be noted that direct illumination by the Sun cannot be excluded under all circumstances.

Camera	X LDR (mm)	Y LDR (mm)	Z LDR (mm)	Azimuth with respect to Axis X LDR angle (trigonometric direction)	Elevation with respect to plan (oXY) LDR
RH Mono CAM 1	28.4	-96.4	575.6	162.4°	-45°

Table 17: MUPUS-TM camera position and line of sight expressed in the LDR frame

The design of the MUPUS TM thermopile detectors is shown in Figure 10-3. The IR detectors are mounted inside a TO-5 transistor housing and an infrared filter glued to the front cover provides the desired passband in the wavelength range 5-25 µm. A black anodized aperture stop located between detector and filter is suppressing stray light. The detector itself is composed of a Si frame onto which a thin Si_3N_4 membrane is spanned that also contains 100 micromachined Bi/Sb thermocouples. In the center of the membrane, a spot of 1 mm diameter is blackened which works as absorber for the infrared radiation. Due to this design the absorber area of the detector is conductively well isolated from its environment and in radiative exchange with the target filling the field-of-view (FOV) and also with the detector housing (mainly the aperture stop). Depending on the net flux between the absorber area and the target (here assuming that the housing is in thermal equilibrium with the detector) the temperature of the absorber varies with the temperature of the target. The change in temperature is sensed by the thermocouples via the Seebeck effect. The hot junctions are located beneath the absorber area and the cold (or reference) junctions on the Si detector frame, which is acting as a heat sink. The measured signal is a thermovoltage that is proportional to the temperature difference between the absorber and the reference junction.

A miniature Pt-1000 sensor located beside the thermopile detector measures the temperature of the reference junction itself.

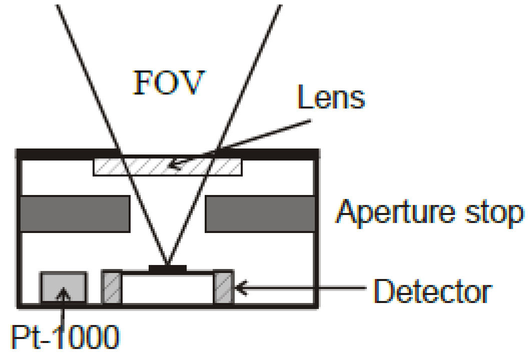


Figure 10-3 Sketch of MUPUS TM detectors

The voltage generated by a thermopile can be expressed as

$$U_{th} = SA_d\pi\left[v\int_0^\infty \varepsilon(\lambda)\varepsilon_a(\lambda)\tau(\lambda)\big(B(\lambda,T)-B(\lambda,T_{ref})\big)d\lambda + v_H\int_0^\infty \varepsilon_a(\lambda)\big(B(\lambda,T_H)-B(\lambda,T_{ref})\big)d\lambda\right] + U_{off}(T_{FEE}) \quad (6)$$

where U_{th} is the measured thermopile voltage (the signal), λ is the wavelength of the radiation, S is the sensitivity of the detector in [V/W], A_d is the absorber area, $\varepsilon(\lambda)$ is the emittance of the object inside the field-of-view, $\varepsilon_a(\lambda)$ is the emittance of the absorber area, v and v_H=1-v are the view factors (between 0 and 1) to the target and the housing, $\tau(\lambda)$ is the wavelength dependent filter transmittance, $B(l,T)$ is the Planck function, and T, T_{ref}, T_H are the temperatures of the object, the thermopile reference junction, and the detector housing (here assumed to be homogeneous), respectively. U_{off} is the electrical offset voltage of the front-end-electronics (FEE). Note that the second term in Eq. (6) can be viewed as a thermal offset which is generated by thermal inhomogeneity inside the detector housing. In thermal equilibrium T_H should always be very close to T_{ref}, and, thus, the thermal offset would vanish. Assuming that U_{off} is negligible and that the detector is in quasi thermal equilibrium (T_H=T_{ref}), it is obvious from Equation (1) that the output voltage is zero if the object temperature and detector temperature are equal and that the equation is symmetric about T and T_{ref}. This implies that instead of the usual approach of varying the target temperature while keeping the detector temperature constant, one may as well determine the sensitivity from a variation of the detector temperature while holding the object temperature constant. This was exactly the case when MUPUS TM observed deep space during descend while the temperature of the sensor head itself was continuously rising, thereby allowing the re-calibration of the MUPUS TM sensor during descent (re-calibration was necessary because a considerable decrease of the sensitivity of the detectors compared to the ground calibration was detected already soon after launch during Rosetta commissioning).

Table 49 TM IR channel characteristics

IR channel	**Passband [µm]**	**View factor**	**Gain factor**
A	8-14	0.367	-401
B	6-25	0.284	-401
C	12.8-15.2	0.460	-2007
D	5.5-7.0	0.284	-6043

Assuming that the detector is in quasi thermal equilibrium and that the emitting object is a blackbody the above equation can be simplified and results in the defining equation for the "brightness temperature" T_b of a thermopile channel:

$$U_{th} = S(T_{ref}) A_d \pi \nu \int_0^\infty \varepsilon_a(\lambda)\tau(\lambda)\left(B(\lambda,T_b) - B(\lambda,T_{ref})\right)d\lambda + U_{off}(T_{FEE})$$

(7)

The "brightness temperature" T_b is the equivalent temperature of a blackbody (ε_{av} = 1) generating the same signal as the real target surface.

The sensitivity S is known to be an almost linear function of detector temperature:

$$S(T) = S(T_0)\,(1 + TCS\,(T_{ref} - T_0)) \tag{8}$$

where T_0 is chosen as 173.15 K and TCS is the temperature coefficient of the sensitivity.

The offset can also be described in a linear approximation:

$$U_{off} = U(T_0) + TCV * (T - T_0) \tag{9}$$

Thus, the calibration of a TM channel is in general provided by the 4 calibration coefficients $S(T_0)$, TCS, $U(T_0)$, TCV and the required calibration factors which relate the measured reference channel voltage to the reference resistance (3 coefficients, see below) or temperature.

TM recalibration

Inflight calibration runs performed during commissioning and cruise revealed that the TM sensors had undergone a considerable decrease of sensitivity of about 10-35% and probably also a shift in offset when compared with the ground calibration. These changes had occurred already before launch and were stable during cruise and hibernation. Unfortunately, the environmental conditions encountered during cruise were such that a direct comparison required for a quantitative statement about the changes could not be made. Therefore, an attempt was made to re-calibrate the TM sensors during descent using Equation (7) and making the additional assumption that the offset voltage can be neglected. This attempt was successful with the broadband IR channel B, whereas it failed for the other channels, mainly because of the large thermal disturbances caused by temporary direct sunlight falling onto the sensor head due to rotation of the Lander during descent. The result of the re-calibration is given in Table 52. Due to the assumption of negligible offset voltage the coefficients U_0 and TCV are zero in Table 52.

TM calibration steps

To perform the calibration of TM the following steps are carried out:

1. Correction of electromagnetic disturbances caused by activities of other subsystems or instruments (mainly RF link) on the Lander. These disturbances manifest themselves as additional temporal offsets on top of the TM signal and can, therefore, quite easily be identified and largely removed from the TM signal. The offsets generated by different disturbing sources are given in Table 50 and the start and end times of the disturbances identified during SDL and FSS are given in Table 51.
2. Conversion of the digital values [DN] (signed 14-Bit integers covering the range -8192 to 8191) to "raw" voltages $U_{raw} = DN * 6/(2^{14} - 1)$
3. Conversion of the raw voltages to calibrated voltages by applying the calibration of the ADC

$$U_{cal} = c_0 + c_1 U_{raw} + c_2 U_{raw}^2$$

with the FM ADC conversion coefficients c_0= -18.58 mV, c_1=1.00886, c_2=5.03 10^{-4} V^{-1}

4. Computing the thermopile voltages by dividing the calibrated voltages by the gain factors of each TM channel $U_{th,i} = U_{cal,}i/g_i$ (i=1,…,4, see Table 50)
5. Computing the resistances of the reference temperature sensors $R_{ref,i}$ by application of the reference channel calibration coefficients from Table 52

$$R_{ref} = c_{R0} + c_{R1} U_{cal} + c_{R2} U_{cal}^2$$

6. Compute the reference temperatures $T_{Ref,i}$ from $Rr_{ef,i}$ by applying the Callendar-van-Dusen (CVD) relation with the industrial standard conversion coefficients for Pt-100/1000

$$R(t) = R_0(1 + at + bt^2 + c(t - 100)t^3)$$

where t is the temperature in °C, R_0 is the resistance at 0°C (e.g. 100 Ohm or 1000 Ohms for a Pt-100 or Pt-1000, respectively), a=0.003983, b=-5.775 10^{-7}, c=0 for t > 0 and c=-4.183 10^{-12} for t < 0.

7. Inversion of Equation (7) using the 4 thermopile calibration coefficients, S(T_0), TCS, U(T_0), TCV in terms of IR channel brightness temperature (ε_{av} = 1). The necessary filter transmittances and the absorber emissivity are provided in the ASCII text file "Tau_FM.txt".

Table 50 Offsets in thermopile signals created by switch-on of different disturbing subsystems/experiments

IR channel	Passband [µm]	Delta_RF [DN]	DU [µV]*	Source
A	8-14	-12	11.0	RF link
B	6-25	-5	4.6	RF link
C	12.8-15.2	-92	16.8	RF link
D	5.5-7.0	-341	20.7	RF link
A	8-14	-9	8.2	COSAC
B	6-25	-6	5.5	COSAC
C	12.8-15.2	-28	5.1	COSAC
D	5.5-7.0	-65	3.9	COSAC
A	8-14	4	-3.7	UNKNOWN
B	6-25	4	-3.7	UNKNOWN
C	12.8-15.2	16	-2.9	UNKNOWN
D	5.5-7.0	35	-2.1	UNKNOWN

*difference in thermopile voltage after correction for different gain factors

Table 51 EMI disturbances seen in TM signals during SDL and FSS

Start time [UTC]	End time [UTC]	Source
12-NOV-2014 10:37:24	12-NOV-2014 17:31:00	RF link
12-NOV-2014 17:31:30	12-NOV-2014 17:38:10	RF link
12-NOV-2014 17:38:30	12-NOV-2014 17:49:15	RF link
12-NOV-2014 17:49:30	12-NOV-2014 17:51:00	RF link
12-NOV-2014 17:54:00	12-NOV-2014 17:55:15	RF link
12-NOV-2014 17:59:00	12-NOV-2014 18:00:00	RF link
12-NOV-2014 18:02:00	12-NOV-2014 18:04:15	RF link
12-NOV-2014 18:09:40	12-NOV-2014 18:11:15	RF link
12-NOV-2014 18:12:40	12-NOV-2014 18:14:15	RF link
13-NOV-2014 05:33:04	13-NOV-2014 06:47:30	RF link
13-NOV-2014 06:47:00	13-NOV-2014 06:58:00	COSAC
13-NOV-2014 06:49:00	13-NOV-2014 06:54:30	RF link
13-NOV-2014 06:56:30	13-NOV-2014 07:42:00	RF link
13-NOV-2014 07:42:30	13-NOV-2014 09:04:30	RF link
13-NOV-2014 08:48:00	13-NOV-2014 09:00:15	COSAC
13-NOV-2014 09:05:15	13-NOV-2014 09:19:45	RF link
13-NOV-2014 09:21:45	13-NOV-2014 09:22:15	RF link
13-NOV-2014 09:24:45	13-NOV-2014 09:25:45	RF link
13-NOV-2014 09:26:45	13-NOV-2014 09:28:00	RF link
13-NOV-2014 09:29:20	13-NOV-2014 09:30:20	RF link
13-NOV-2014 09:38:00	13-NOV-2014 09:38:30	RF link
13-NOV-2014 09:39:00	13-NOV-2014 09:39:30	RF link
13-NOV-2014 10:50:15	13-NOV-2014 11:01:45	COSAC
13-NOV-2014 12:51:50	13-NOV-2014 13:04:00	COSAC
13-NOV-2014 19:27:37	13-NOV-2014 23:04:15	RF link
13-NOV-2014 23:04:45	13-NOV-2014 23:05:15	RF link
13-NOV-2014 23:07:30	13-NOV-2014 23:08:30	RF link
13-NOV-2014 23:08:45	13-NOV-2014 23:09:20	RF link
13-NOV-2014 23:18:30	13-NOV-2014 23:31:00	UNKNOWN
14-NOV-2014 03:50:20	14-NOV-2014 04:24:30	APX
14-NOV-2014 05:00:20	14-NOV-2014 05:31:45	APX
14-NOV-2014 09:00:34	14-NOV-2014 09:06:15	RF link
14-NOV-2014 09:07:00	14-NOV-2014 10:03:15	RF link
14-NOV-2014 10:04:00	14-NOV-2014 11:49:35	RF link
14-NOV-2014 21:47:18	15-NOV-2014 00:10:13	RF link

Table 52 TM IR channel calibration coefficients

	Channel A**	**Channel B**	**Channel C****	**Channel D****	**Blackbody**
$S(T_0)$* [V/W]	813.5	593.6	901.8	18933	
TCS [%/K]	-0.693	-0.326	-0.544	-2.67	
$U(T_0)$ [V]	0	0	0	0	
TCU [V/K]	0	0	0	0	
c_{R0}	801.707	801.815	802.720	800.440	114.261
c_{R1}	180.987	180.898	181.565	180.274	35.050
c_{R2}	1.06060	1.10246	1.08723	1.11075	1.0567

*T_0=173.15 K

**calibration failed

10.3 ANC-T

The calibration of the anchor temperature sensors is achieved by:

1. Conversion of the digital values [DN] (signed 14-Bit integers covering the range -8192 to 8191) to "raw" voltages $U_{raw} = DN * 6/2^{14}$
2. Conversion of the raw voltages to temperature in [K] by applying the calibration relation:

$$T[K] = c_{0,A} + c_{1,A}U_{raw} + c_{2,A}U_{raw}^2 - 273.15$$

with the coefficients $c_{0,A}$=-22.92, $c_{1,A}$=62.574, $c_{2,A}$=0.8283.

10.4 Depth Sensor (DS)

The MUPUS DS measurements are based on a potentiometer principle. The upper contact of the potentiometer is fixed at the upper end of the PEN tube whereas the lower contact is the DS assembly itself. The resistor is realized by a conductive paint which covers the whole outer part of the PEN tube. Assuming that the paint is homogeneous the resistance is proportional to the distance d between the two contacts.

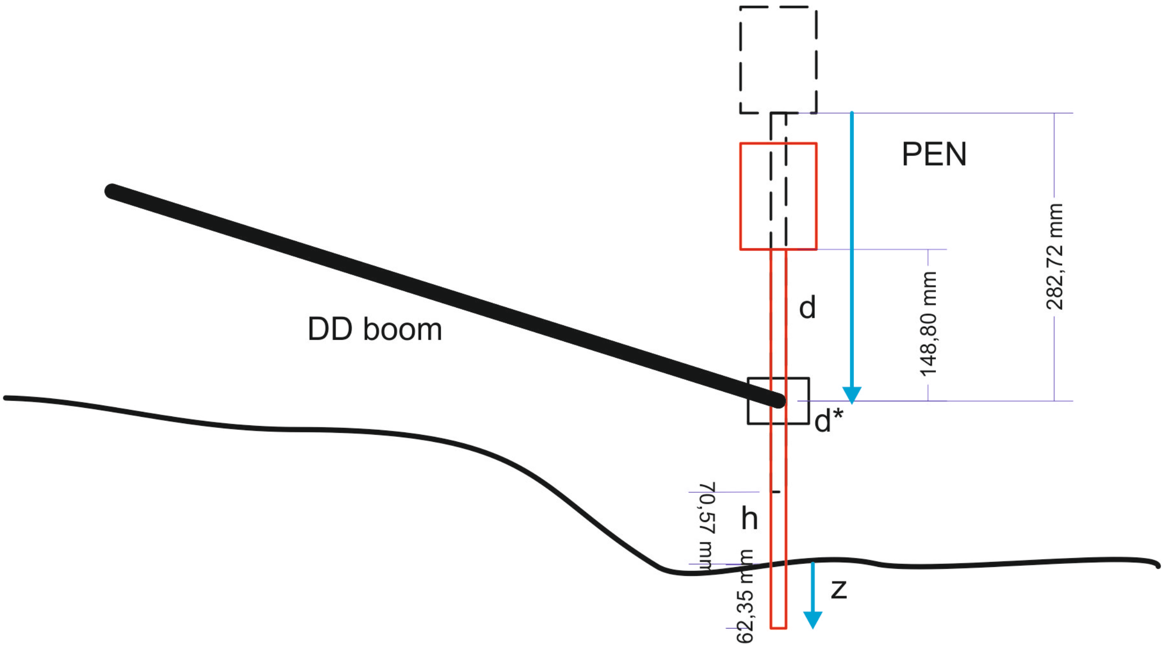


Figure 4 Depth Sensor measurement principle

Defining d=0 at the upper contact and positive downwards along the tube, the initial location of the DS during deployment is d*=282.5 mm. After hammering starts the PEN tube is sliding down through the DS assembly (note that DS is still connected by the DD boom to Philae and hold in place by the boom), thereby decreasing the distance d between the upper contact and DS. If the PEN tip was put onto the ground before insertion starts, then the depth z (z=0 at the surface, positive downwards) of tip is just z=d*-d, but in general the height h above the surface in the deployed position after Lander lowering (e.g. before start of insertion) needs to subtracted :

$$z = d^* - d \quad - h$$

This means that the real depth of the tip can only be known if the initial height of the DS position before insertion can be determined. Therefore, another CIVA image taken after lowering the Lander would be very valuable, and especially important for the interpretation of SESAME/CASSE measurements. Note that z might already be positive if the tip has already penetrated the soft surface during lowering of the Lander.

Since the resistance of the DS paint is slightly temperature dependent (measurements performed with the FS showed a nonlinear increase of resistance with decreasing temperature, for T > 200 K there is little change compared to room temperature, but at LN2 temperature of -196°C the resistance of the paint is about 15% higher than at RT) and may also change a bit over time due to ageing processes, to compute the actual distance d between the DS assembly and the upper contact, a reference measurement of the DS is made before insertion starts. This reference measurement is thne used to normalize the actual readings and z is computed from:

$$z = d * \left(1 - \frac{V - OFF}{V * - OFF}\right) - h$$

Where V* is the digital numbers [DN] at the initial DS reference position and V is the actual reading at a distance d. The offset OFF is a constant corresponding to the top position of the DS which is mainly due to a remaining resistance ROFF (caused by the finite thickness of DS) plus the offset voltage of the ADC. The numerical value of the offset is taken from the MUPUS configuration parameter INS.DepthOff (Word 44).

Note: For MUPUS operating on the comet the value of h could not be determined, due to lacking 3D information from images. Therefore, only z values computed with the assumption h=0 (i.e the movement of the PEN tip relative to its initial position) were determined and archived (in units of [mm]) in the "MUP_HAMS3B_141114002115.TAB" file on the ESA PSA server.

At some point during PEN insertion the DS should have reached the Position for DS release (Figure 5). At this point phase H3 starts, and, only after the DD boom is fully retracted, the hammering will continue. In the current FSS procedure (V1.6) the hard coded default of $V_{Retract}$=256/890 V*+Ins.DepthOff is modified by a patch TC to $V_{Retract}$ =128/890 V*+OFF which approximately corresponds to about d=43 mm. The magic number 890 used in the above formula stems from the fact that the DS reference position is located at a fraction of 0.89 of the full possible downward movement of the DS. Note further that this is NOT the lower end of the tube but corresponds to a distance of 317 mm below the upper contact which is still a few cm above the tip.

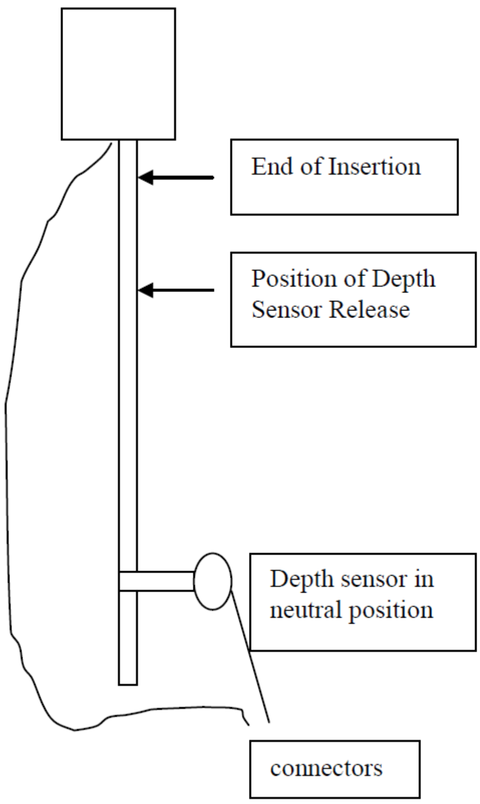


Figure 5 Critical DS positions

After the boom is retracted the interpretation of DS measurements as insertion depth of the tip becomes less reliable. This is because either the whole DS assembly could move downwards together with the PEN tube without changing the d-value (e.g. d=constant but real z further increases) or the DS assembly may even slide down the tube during electromagnet operation between two 4-strokes (if friction does not prevent this) leading to a temporary increase of the measured d value. In any case hammering will be continued until the final value for maximum insertion given by $V_{max} = 37*V_{Ref}/890$ + Ins.DepthOff is reached or until the maximum configured number of strokes have been made, which ever condition is fulfilled first.

By interpreting MUPUS DS measurements it should also be kept in mind that this is not high precision measurement. The repeatability of a measurement at the same depth has been determined by experiments with the FS to about 70-75 DN (1-sigma) which corresponds to an uncertainty of +/-3.5 mm. Furthermore, systematic deviations from the assumed linear relationship between resistance reading and d may also have the same order of magnitude, since inhomogeneities in the thickness of the conductive paint cannot be excluded.