Herschel-SPIRE
MCU / DCU
Command List ICD

SPIRE-LAM-DOC-003128

Prepared by :	Signature
Didier Ferrand Date : 12 jan 2006	Didier Ferrand.
Pour application	**Signature**
Dominique Pouliquen Date : 12 Jan 2006	

Change Record

Date	Issue	Rev	Modifications
20/09/2000	1	0	First release
11/04/2002	2	0	Second release
	2	1	Changed command `0x56` ScanReverseSpeed instead of SMaxSpeed.This command is used to define the speed in reverse way.
15/01/2003	3	0	All pages
30/09/2004	4	0	Scheduling of the MCU:400 µs instead of 360 µs No forbidden command implemented SETBOOTRAM Mnemonic is `0x24` The number of scan decrements each time the ramp sign changes (odd/even) Added telemetry number of frame Added telemetry frame start/stop New order for MAC HK: 5V,14V,-14V,15V,-15V, mac temp,smec temp,bsm temperature starting at mnemonic `9E0` Removed the LVDT Look Up Table set parameter **SetLVDTLUT** New bit definition in SMEC status word New BSM control parameter after FS tests at ATC Added 5th parameter in SMEC science packet
14/10/2005	4	1	New task scheduler sampling: 420 µs instead of 400 µs Debug of Jiggle mnemonic errors Removed not used commands Update actual command list implemented on QM2 to be implemented as well on future Flight Model Added complete scenarios to calibrate instrument from beginning
06/01/2006	5	0	Put in accordance command list with declared commands in “as built” QM2 and PFM code Includes the commands for debugging.

Distribution list

Institute	Name	Issue/Revision									
		1/0	2/0	2/1	3/0	4/0	4/1	5/0			
CNES	Blanc Yves										
CNES	Mercier K.										
RAL	Clark E										
RAL	Griffin M.J										
RAL	King K.J	X	X	X	X	X	X	X			
RAL	Sawyer E							X			
RAL	Swinyard B.M	X	X	X	X	X	X	X			
CEA	Auguères J.L	X	X	X	X	X	X	X			
CEA	Cara C	X	X	X	X	X	X	X			
CEA	Tourette T										
CEA	Fontignie J										
LAM	BaluteauJ.P							X			
LAM	Blanc J.C										
LAM	Blanchard P										
LAM	Boit J.L										
LAM	Calabria P										
LAM	Castinel L										
LAM	Colin C										
LAM	Dohlen K										
LAM	Fabron C.										
LAM	Ferrand D	X	X	X	X	X	X	X			
LAM	Garcia J.										
LAM	Grassi E.										
LAM	Laurent Ph.										
LAM	Le Fevre O										
LAM	Levacher P										
LAM	Martin L										
LAM	Moreaux G.										
LAM	Origné A.										
LAM	Pouliquen D.	X	X	X	X	X	X	X			
LAM	Repetti B.							X			
LAM	Rousset G.							X			
LAM	Vicq H.										
OHP	Vors P.										

Table of contents

1 INTRODUCTION

1.1 PURPOSE AND SCOPE

This document describes the commands which can be sent by the DPU to MCU in order to control and monitor the 3 motorised axis of SPIRE (BSM Chopper and Jiggle axis and the SMEC axis). These commands are the implemented ones in the MCU QM1, ½ QM2, QM2, PFM and FS.

1.2 APPLICABLE AND REFERENCE DOCUMENTS

1.2.1 APPLICABLE DOCUMENTS

AD#	Title	Reference issue and date
1	DRCU/DPU Interface Control Document	Issue 1.1 25/06/04
2	A Generic DPU interface for SPIRE DRCU Subsystems	SPIRE/XXXXXX Issue 0.4 11/06/02
3	Beam Steering Mirror Control Software Requirements	SPIRE-ATC-XXX-000xxx Issue 2.1 12 November 2002

1.2.2 REFERENCE DOCUMENTS

None

1.2.3 ACRONYMS

AD	Applicable Document
BSM	Beam Steering Mirror
CQM	Cryogenic Qualification Model
EGSE	Electrical Ground Support Equipment
FM	Flight Model
FPU	Focal Plane Unit
FTS	Fourier Transform Spectrometer
FTSE	FTS warm Electronics
FTSP	FTS Preamplifier for the position encoder signals
H/K	House Keeping
H/W	Hardware
I/F	Interface
MAC	Multi Axes Controller
MCU	Mechanism Control Unit
PTA	Parameter Table Address
N/A	Not Applicable
RD	Reference Document
S/C	Spacecraft
SM	Spare Model
SMEC	Spectrograph MEChanism
S/W	Software
TBC	To Be Confirmed
TBD	To Be Define
TBW	To Be Written
TC	TeleCommand
TM	TeleMetry
WE	Warm Electronics

2 MCU CONTROL DESCRIPTION

The control of the 3 axis is performed by a 21020 DSP on the basis of trajectory generators and digital PID controllers acting on the 3 mechanisms by mean of 3 DACs.
The sensors (optical encoder and LVDT for SMEC, Magneto resistive sensors for BSM) are read out by mean of 2 multiplexed ADCs.
The control parameters are put in memory for configuration purpose by mean of a command line (bi-directional 32 bit/330 kHz synchronous serial line) connected with the DPU.
The software is based on a master scheduler on the principle of time sharing without the use of a specific multitask kernel.
The tasks to be performed shall be called on a software interrupt generated by the inner DSP timer.
The software interrupt defines the global sampling time (i.e. the computation cycle) of the DSP tasks @ a rate of 420 µs.
At each cycle the following tasks are performed:

- the SMEC control loop task (including trajectory generation and PID)
- the chopper control loop task (including slew rate limiter of position reference and PID),
- the jiggle control loop task,
- the communication with the command line and other various internal DSP tasks,
- telemetry packet concatenation and transmission to high rate serial link

3 COMMAND TYPES

There are typically three types of commands:

- **Set Command**: the parameter is put in the DSP memory for immediate or delayed execution. This DSP memory area is called Parameter Table, the related address is called **PTA** (Parameter table Address)
- **Get Command**: the dedicated parameter in memory can be read by the DPU
- **Immediate Command:** Reset Command or Start Time stamp Counter (sync). This type of command is directly interpreted by the communication FPGA logic.

3.1 COMMAND GENERAL FORMAT

The command format specification is compatible with DRCU/DPU Interface Control Document [AD1].

A command is a 32 bits word with the following allocation:

32 bits Command Word					
	Sync Signal	Sub System header	Command mnemonic		Parameter value
			Set/Get Bit	PTA	
Bit Alloc	31-30	Bits 29-28	Bits 27	Bits 26-16	Bits 15-0
Meaning	10:Reply 11: No reply	01 is the MCU address	Set:0 write parameter Get: 1 read parameter	see mnemonic list	16 bits parameter

3.2 EXAMPLES OF COMMANDS

The following example shows how to send bit pattern on the command line for the SMEC optical encoder power up at level 7 (max power).

#1	SetEncoderPwr	Level 7
Sync	SS@	12 Bits command mnemonic	16 Bits parameter
1 0	0 1	0 0 0 0 0 1 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1
9		0	4	0	0	0	0	7

The following example shows how to send bit pattern on the command line to read the chopper magneto -resistive ADC value. The get bit is set to 1.

#1	Get CMagnetoresistiveSignal	
Sync	SS@	12 Bits command mnemonic	16 Bits parameter
1 0	0 1	1 0 0 1 0 0 0 0 0 0 1 1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
9		9	0	3	0	0	0	0

3.3 COMMAND REPLY

A part a broadcast command that synchronises the date counter, all command shall generate a reply to DPU according to AD1 and AD2.

The response format is identical to the command format, where the SubsystemID field (bits 29 and 28) is replaced by a response status as follows:

- 00: command accepted
- 01: command unknown
- 10: command forbidden
- 11: Timeout

Example of reply of:

#1	Get CMagnetoresistiveSignal	
Sync	SS@	12 Bits command mnemonic	16 Bits parameter

1	0	0	1	1	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9				9				0				3				0				0				0				0			

Is:

1	0	0	0	1	0	0	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0
8				9				0				3				F				E				1				0			

(here we suppose that the Magnetoresistive signal value is 0xFE10)

3.4 PARAMETER TABLE ADDRESS (PTA) GENERAL MAPPING

The command field allows 11 bits for Parameter table address, but this table is in fact limited to 9 bits in DSP software which defines a 640 word buffer area.
The allocation has been defined according to the expected number of parameters for 7 general command categories, including further possible extension. When the DSP receives a command on the communication register of the command line, it extracts the 16 bits parameters, verifies the existence of the command and puts the parameter in the parameter table pointed out by the command mnemonic.

Start@	End@	Length	Command category
000h	**1Fh**	**32**	Communication FPGA
020h	**03Fh**	**32**	MCU boot and configuration
040h	**0BFh**	**128**	SMEC Control and HK
0C0h	**13Fh**	**128**	Chopper control and HK
140h	**1BFh**	**128**	Jiggle control and HK
1C0h	**1DFh**	**32**	Telemetry and Trace configuration set commands
1E0h	**1FFh**	**32**	MCU general HK and Miscellaneous Set Commands

If the command does not exists (a specific bit is set in the parameter table at initialisation to determine authorised commands), the reply consist in the pattern ‘command unknown’.
The “forbidden command” capability is not implemented in MCU software since proper operation is guaranteed by proper scenario and parameter use. Indeed MCU cannot guarantee a proper use by internal very complex checking. Safety aspects on mechanics and electronics are ensured by inner software and electronics design (limitations, scales, filtering, slew rate limiters, etc…).

4 COMMAND LIST

The commands relates to following functions:

- Set/get of control parameters for the axis motion (start, stop, position and speed,...)
- Set/get of parameters for minimum calibration needed for adjustment of actuators and sensor
- Get of maximum information about signals and processing (ADC and computed values).

The complete list of parameters associated with the Set or Get command is listed after.

4.1 COMMUNICATION FPGA COMMANDS

These commands are only available at FPGA level. See AD1 and AD2 documents. The main commands are the subsystem cold reset (90010005h) and reset of the date counter (F0030000h) which is used to time stamp the sampled data provided in the telemetry.

Command category:		FPGA Commands
PTA	Command Name	Action and Parameter specification
000h 01Fh	FPGA Commands	Cf DPU/Control Document [AD1]

4.2 MCU COMMANDS WHEN PROGRAM IN PROM AFTER COLD RESET

These commands are only available when the DSP is booted on PROM after cold reset i.e. after power on of the Power Supply Unit (PSU).
The boot program functions are the automatic test of the ram (program and data) and the copy of the program to the ram program on specific commanding scenario.
During development, there is the possibility to load an external program in the RAM. This feature is not to be inhibited for flight, so the final code can be uploaded with proper EGSE.
When the program is running in PROM and before second reset on RAM, there is no code for mechanism control neither signal acquisition since the PROM needs 4 wait states which slow down the computation. The RAM data is not used as well, only verified.
We completely rely on RAM integrity for proper operation of the control.

Mnemo	Command Mnemonic Name	Action and Parameter specification
820h	**GETBOOTSTATUSREGISTER**	Reply parameter: Boot Status Register(ref boot.h file) bit 0: 1 when RAM integrity done 0 otherwise bit 1: 1 when test program ram failed bit 2: 1 when test data ram failed bit 3: 1 when copy from PROM to RAM program is in progress bit 4: *BOOT_MODE :* 0 when mode is copy from PROM (always the case for flight) When proper ram integrity the returned parameter is 1
021h **821h**	**SETDOWNLOADCONF** **GETDOWNLOADCONF**	Configure the download of program Read the download configuration. In flight the only possibility is the copy from PROM to RAM. To download the program from PROM the parameter is 0xC000 (command to be sent: 9021C0000)
022h	**SETDOWNLOADWORD** **(WITH EGSE ONLY)**	Download a 16 bit word to program memory
822h	**GETDOWNLOADWORD** **(WITH EGSE ONLY)**	Read the last 16 bit word to be downloaded program memory
023h	**SETDOWNLOADCOUNTER** **(WITH EGSE ONLY)**	Set the downloaded program counter
	CANCELLED	~~Upload (dump) configuration of program memory~~
	CANCELLED	~~Reads the program memory upload configuration~~
	CANCELLED	~~Upload a 16 bit word from program memory~~
	CANCELLED	~~Read the uploaded program counter~~
024h	**SETBOOTRAM**	With parameter =1: Launch the second reset and boot on RAM

Table 1: MCU/DSP Command Mnemonics when in PROM

4.3 SMEC COMMANDS (AVAILABLE AFTER SECOND RESET ON RAM)

Mnemo	Command Mnemonic Name	Action and Parameter specification
040h **840h**	**>SetSEncoderPwr** **>GetSEncoderPwr**	Power the optical encoder led at various 8 illumination levels encoded on 3 bits. 000=not led illumination 111(7)= max 5V illumination. Other value: 8 allows to init the offset values of encoder on actual signal and then return to previous commanded value.
041h **841h**	**>SetSLVDTPwr** **>GetSLVDTPwr**	Power on/off the LVDT oscillator (Boolean): 0: the LVDT oscillator is inhibited. 1: the LVDT oscillator is activated. Other value: LVDT oscillator remains at same state.
042h **842h**	**Cancelled** ~~**SetSMotorMode**~~ ~~**GetSMotorMode**~~	Control of the relay which enable the motor pwr amp Previously, the motor was short cut automatically by a monostable relay when no power. Shall produce unknown command
043h **843h**	**>SetSLaunchLatch** **>GetSLaunchLatch**	Engage/disengage SMEC Launch Latch 1 : engage 2 : disengage Other value: no action. Inhibition of this command may be done in DPU. The command parameter is automatically set to 0 after being into account (impulse command).
044h **844h**	**>SetSLoopMode** **>GetSLoopMode**	Open or close loop mode selection: 0: open loop, DAC imposed to steady value which is frozen. 1: closed loop on encoder signals 0&1 (sine and sine 120° encoder ADC signals) ~~2: closed loop on encoder signals 1& 1~~ **(cancelled)** ~~3: closed loop on encoder signals 1&0~~ **(cancelled)** 4: closed loop on LVDT position calibrated which LVDT scale and LVDT offset from direct LVDT ADC value (raw without linearity compensation). ~~5:closed loop on LVDT corrected for scan using a LUT compensation for speed~~ **(cancelled)** 6: open loop with feedforward control + backemf effect;
045h **845h**	**>SetSTrajEndPosition** **>GetStrajEndPosition**	Specifies the position to reach in both scan or step mode Range 0:65536 Unit: 1µm (0:65536 unsigned number) from mechanical limit which is around -8mm from ZPD. 0 is set after program initialisation and motion initialisation. The mechanical limit is the absolute reference for the motion (typically at –8mm from ZPD and 0 LVDT) Typical values: -3.2mm from ZPD i.e. **4800**µm from mechanical limit:**0x12C0** ZPD position at about +**8000**µm starting from mechanical limits: **0x1F40** +3.2mm from ZPD i.e. **11200**µm from mechanical limit:**0x2BC0** +32mm from ZPD i.e. **40000**µm from mechanical limit :**0x9C40**
046h **846h**	**>SetSTrajStartPosition** **>GetSTRajStartPosition**	Specifies the initial position for an automatic scan. The scan shall evolve N times from Start to End Positions. Same unit as TrajEnd Position
047h **847h**	**>SetSScanSpeedForward** **>GetSScanSpeedForward**	Specifies the scan speed in positive direct way Range 0:65535 Unit: 0.1µm/s e.g. 0x90471388 command set speed at around 500µm/s. The speed should be used in closed loop Limited to 2mm/s (20000) to avoid miscounting of encoder signals sampled in software. The actual speed should be fine calibrated due to scan speed and MCU clock relationship
048h **848h**	**>SetSScanNumber** **>GetSScanNumber**	Specifies the number of scans value Range 0-65535: Default value: 0. The max number of scans is 65535 The number of scan is decremented each time the ramp sign changes (i.e. each forward or reverse one way scan)
049h **849h**	**>SetSTrajMode** **>GetSTrajMode**	Specifies the type of trajectory: 0: trajectory generation is stopped and remains steady. Allows to start/stop trajectory computation 1: move the SMEC to end trajectory in a filtered step mode 2: scan mode for N times at programmed direct and reverse speed 3: not used. 4: initialisation: reset encoder counter on start trajectory value if user wants to force encoder counter value. (note: when closing the loop the encoder is automatically reset to actual trajectory value to avoid transient glitches) ~~5 initialisation move to capture LVDT within linear range~~ **(cancelled)**
04Ah **84Ah**	**>SetSKp** **>GetSKp**	Proportional gain of the digital PID controller To be tuned after Kd only because of servo loop stability constraints
04Bh **84Bh**	**>SetSKd** **>GetSKd**	Derivative Gain of the digital PID controller

MCU / DPU Command List ICD

Mnemo	Command Mnemonic Name	Action and Parameter specification
04Ch 84Ch	>SetSDerivFilter >GetSDerivFilter	Sets the filtering time constant to calculate the filtered derivative term. See note on PID tuning to use this parameter along with Kd parameter.
04Dh 84Dh	>SetSKi >GetSKi	Integral gain of the digital PID controller To be tuned only after Kd and then Kp (stability constraints)
04Eh 84Eh	>SetSIntegrationLimit >GetSIntegrationLimit	Loads/Reads the integration saturation for the integral compensation of the servo Limit the Integral effect by saturation level
04Fh 84Fh	>SetSIntegrationThreshold >GetSIntegrationThreshold	Loads/Reads the integration threshold (dead zone before integral computation) for the integral compensation of servo. Typically no threshold is necessary, so no need to change default value.
050h 850h	>Not used	
051h 851h	>SetSRateLimit >GetSRateLimit	Trajectory speed slew rate limiter to reduce encoder count loss risk during step changes. Default: 300 to get around 0.5 s transient during scans. 20 value: high limitation 65535: no limitation
052h 852h	>SetSDerivFilter2 >GetSDerivFilter2	Command declared but with no effect and dedicated for possible future new derivative filtering algorithm.
053h 853h	>SetSFeedforwardDiffgain >GetSFeedForwardDiffGain	Command declared but with no effect and dedicated for possible future derivative on feed forward open loop control
054h 854h	>SetSFeedforwardgain >GetSFeedForwardGain	Specifies the ratio between EndTrajectory and motor current for open loop operation Unit: (delta current ADU)/(delta position microns) * 10000. Default value: 30518.
055h 855h	>SetSFeedForwardOffset >GetSFeedForwardOffset	Specifies the offset current to be applied when mechanism at 0 position. Unit: Dac unit Default:0x8000 (0mA). Use in case of gravity
056h 856h	>SetSScanRevSpeed >GetSScanRevSpeed	Specifies trajectory speed for fly back (reverse speed) Unit: 0.1µm/s. Default:5000 Should be Limited to 2mm/s in closed loop operation
057h 857h	>Set EncoderSignal1Amp >Get EncoderSignal1Amp	Read or set the computed encoder sine amplitude for normalisation of the arcsine computation (fine interpolated position). After a few periods of the sine signal during a scan, the amplitude is automatically computed by software.
058h 858h	>SetEncoderSignal1Offset >GetEncoderSignal1Offset	Since the software automatically computes an offset value (as well as amplitude) on the encoder sine signals after few sine periods, as a get command it allows to read the computer offset. As a set command, allows to adjust manually the expected first mean signal 1 value for zero crossing and position computation during the initialisation process. The offset adjustment is then automatic based on 3 sine periods.
059h 859h	>SetEncoderSignal2Amp >GetEncoderSignal2Amp	Idem for signal 2
05Ah 85Ah	>SetEncoderSignal2Offset >GetEncoderSignal2Offset	Idem for signal 2
05Bh 85Bh	>SetEncoderSignal3Amp >GetEncoderSignal3Amp	Idem for signal 3
05Ch 85Ch	>SetEncoderSignal3Offset >GetEncoderSignal3Offset	Idem for signal 3
05Dh 85Dh	Declared but no effect/not used ~~>SetLVDTLUT~~ ~~>GetLVDTLUT~~	~~Load non linear correction for LVDT 16 Values~~ ~~The parameter consists in 2 fields:~~ ~~4 Most significant bits: LUT address~~ ~~12Least significant bits: O.1 µm correction value~~
05Eh 85Eh	>SetLVDTOffset >GetLVDTOffset	The LVDT Offset is the value in µm starting from mechanical limit of the LVDT 0V position. Unit: The offset is the difference in µm. Default: decimal 8000 (8mm).
05Fh 85Fh	>SetLVDTScale >GetLVDTScale	Set the scale factor between ADC value of LVDT DC signal and actual position in µm. Computed LVDT position(micron) = (LVDT ADC-0x8000) *LVDTSCALE *0.00002 +LVDT_OFFSET. Unit:1µm

The following commands are get commands, i.e. read-only commands for the SMEC computed parameters.

Mnemo	Command Mnemonic Name	Action and Parameter specification
860h	**>GetSmecStatus**	Reads the SMEC Status Register. The status register aims to indicate to master DPU external actions (commands) to be performed in relation with some events related to the SMEC axis. This Status Register can only be read and no bit can be cleared since the word is refreshed by the chipset. The status register does not provide precise information on parameter values since individual get commands are available (example: the state of digital I/O shall be read out by individual get LVDT Power, get encoder power, get latch command, etc…) **Bit 0: FATAL ERROR This bit is a summary for DPU about the fact that SMEC mechanics is in fatal error state due to closed loop malfunction and put at safe configuration. DPU must re-launch the entire commanding scenario.** **Bit 1 - LVDT positive (1) or negative (0). Allows the detection of LVDT 0V home position on transient of this bit** **Bit 2 - SMEC Initialised (1) not initialised (0) or has become lost (0). The encoder count is lost when the servo error exceeds tolerated value or when in open loop.** **Bit 3 - Scan up/down indicator: increasing steps (0) decreasing steps (1)** **Bits 4-15 - scan number same as the 16 bit parameter reduced on 12bits**
861h	**>GetSEncoder IncrementPosition**	Read the incremental count on 0Xing of the encoder sine signal to give encoder crude position. Unit: 1µm i.e. every zero crossing of a 2µm sine period encoder signal.
862h	**>GetSEncoderSignal1**	Read the optical encoder sine signal 1 (0°) Signal used for counting and fine arcsine interpolation. Unit: ADU on 16bits
863h	**>GetSEncoderSignal2**	Read the optical encoder sine signal 2 (120°) Signal used for sense discrimination Unit: ADU on 16bits
864h	**>GetSEncoderSignal3**	Read the optical encoder sine signal 3 (240°) Unit: ADU on 16bits. This signal is not used for the position counting, it is an extra signal.
865h	**>GetSLVDT position**	Reads the computed LVDT position = (LVDT ADC-0x8000) *LVDTSCALE *0.00002 +LVDT_OFFSET. Unit:1µm
866h	**>GetSLVDTAC**	Reads the AC decoupled LVDT signal (as an AC channel on a scope to be able to zoom the signal with removal of DC value)). Unit: ADU on 16bits
867h	**>GetSLVDTDC**	Read the DC LVDT signal. Unit: ADU on 16bits
868h	**>GetTrajectoryPosition**	Returns the current desired trajectory position value generated by the trajectory program. Unit: 1µm
869h	**>GetSmecDACvalue**	Returns the value of the DAC value which drives the motor amplifier (0mA=0x8000).
86Ah	**>GetPosition difference encoder/lvdt**	Returns the encoder count value when the LVDT crosses 0. Unit:1µm count
86Bh	**>GetEncoderFinePosition**	Returns the fine arcsine interpolated position of the encoder sine signal. Unit:1nm
86Ch	**Not used**	
86Dh	**Not used**	
86Eh	**>GetMeanSpeed**	Actual scan mean measured speed. This number is signed. Allows verifying the velocity scan error. Unit 0.1 µm/sec digitally filtered 20hz 1st order
86Fh	**>GetMeanPositionError**	Mean position error of the servo (delta between trajectory and sensor position). Unit:10nm
870h	**>GetSMotorCurrent**	Returns the measured motor current value. Unit : 1 ADC unit; 0x8000=0mA; Full scale 0xFFFF=100mA
871h	**>GetSMotorVoltage**	Returns the motor voltage value. Unit 1ADC unit. The motor voltage is Rmot*I +Backemf). (Back emf is proportional to motor speed about 17V/ms-1. 0XFFFF = 0.5Volts, 0X8000= 0Volt.

Mnemo	Command Mnemonic Name	Action and Parameter specification
		Example: motor resistance: 5Ohm, current=100mA, voltage=0.5 Volts ADC value= 0XFFFF

Additional set/get parameters for fine tuning of back emf

Mnemo	Command Mnemonic Name	Action and Parameter specification
090h 890h	>Set Motor Backemf constant >Get "	Gain applied for the feedback of the backemf damping control. To be tuned from 0 with small increases until open loop position well damped.
091h 891h	>SetMotor resistance >Get "	Motor + harness resistance used to subtract RI value from motor voltage. Unit: motor voltage/motor current ratio * 10000. Example for a delta voltage: 11000 related with a delta current of 10000 ADU, gives a MotorResistance parameter = 1.1*10000=11000.
892h	>GetMotorBEMF	Read the motor BEMF voltage i.e. the motor voltage - RI
093h 893h	>SetRate scale factor >Get "	Rate limitation to be applied on motor backemf to avoid fast transient.
094h 894h	cancelled as well as 095h and so on ~~SetPosition scale factor~~ ~~Get "~~	~~Scale factor between reference position and estimated mechanical force~~

4.4 CHOP COMMANDS

Mnemo	Command Mnemonic Name	Action and Parameter specification
0C0h 8C0h	>SetCSensorPwr >GetCSensorPwr	Power on 1mA current in magneto resistive sensor of the chopper axis. 0: Off 1: On
0C1h 8C1h	Cancelled ~~SetBSMLaunchLatch~~ ~~GetBSMLaunchLatch~~	~~Engage/disengage Launch Latch and coil short cut (0 or 1)~~ ~~Note: the get command reflects the command and not the status (available in the chopper status word)~~
0C2h 8C2h	>SetChopLoopMode >GetChopLoopMode	Open or close loop mode selection Parameter definition: 0 = Loop opened : DAC steady 1 = Loop closed on magneto resistive sensor (prime control scheme) ~~2: not used: previously for a second set of parameters~~ 3 = no sensor (open loop operation using backemf)
0C3h 8C3h	>SetChopTargetPosition >GetChopTargetPosition	Set the position step to be reached (also called X position). Unit: ADC Unit (1 ADU)
0C4h 8C4h	>SetChopPosition2 >GetChopPosition2	For Chopper tuning: allow defining a starting position when using BSM Move test (see command C6). Unit: ADC Unit
0C5h 8C5h	Not Used	
0C6h 8C6h	>SetBSMMove >GetBSMMove	0: Chopper and Jiggle are independent and move on free run mode (i.e. the position reference is taken into account immediately) 1: Both axis ordered to move synchronously on this order 2: Test pattern on both axis (step pattern of 1sec)
0C7h 8C7h	>SetCFFOFFSET >GetCFFOFFSET	Allow to add a DAC offset on chopper motor current to move the motor until the sensor ADC value is centred for closed loop control. Same unit as DAC unit (0x8000 is 0mA offset). DAC= (TARGET-CFF_OFFSET)*FFGAIN/3051.8
0C8h 8C8h	>SetCKp >GetCKp	Proportional gain of the digital PID controller
0C9h 8C9h	>SetCKd >GetCKd	Derivative Gain of the digital PID controller (also called rate feedback gain)
0CAh 8CAh	>SetCKi >GetCKi	Integral gain of the digital PID controller
0CBh 8CBh	>SetCIntegrationThreshold >GetCIntegrationThreshold	Integration threshold (dead zone on servo error before starting integration). 0xFFFF = no gap, no threshold
0CCh 8CCh	>SetCIntegratorLimit >GetCIntegratorLimit	Integration limitation for the integral compensation of the servo Allow to saturate the integral effect to avoid too much current.
0CDh 8CDh	>SetCFeedForwardGain >GetCFeedForwardGain	Feed forward gain of open loop control. Unit: DAC value= FFGain/3051*(TargetPosition-CFF_OFFSET) . With. Default value =3051 (0xBEB), the DAC value shall be equal to TargetPosition.(with small truncature)
0CEh 8CEh	>SetCFeedForwardDiffGain >GetCFeedForwardDiffGain	Derivative gain on the Feed forward of open loop. Default value: 0
0CFh 8CFh	>SetDifffiltertc1 >GetDifffiltertc1	Differential filter for PID derivative control. Unit: 10^{e-}4*coeff. default:6667. Common with bemf control may be changed if no sensor mode. Tune the derivative filtering cut off frequency
0D0h 8D0h	>SetDifffiltertc2 >GetDifffiltertc2	Differential filter for PID control. Unit: 10^{e}-1*coeff. default:8333 Common with bemf control may be changed if no sensor mode
0D1h 8D1h	>SetCRateLimit >GetCRateLimit	**Slew rate limiter** to reduce power dissipation during step changes. Unit:10^{e}5 degrees/0.4msec.Default:20
0D2h 8D2h	>SetCMotorBEMFGain >SetCMotorBEMFGain	Motor back emf constant, i.e. the feedback gain applied on the computed backemf damping. Default: 0
0D3h 8D3h	>SetCMotor resistance > GetCMotor resistance	Motor + harness resistance used to subtract RI value from motor voltage. BEMF= MotorVoltage- MotorCurrent*MotorResistance*0.0001. Default: 0

Mnemo	Command Mnemonic Name	Action and Parameter specification
0D4h 8D4h	>SetCMotor inductance > GetCMotor inductance	Motor + harness inductance compensation Default: 0 (Normally not used)
0D5h 8D5h	>SetCRateScalefactor >Get CRateScalefactor	Command declared for possible future use but not implemented
0D6h 8D6h	>SetChopPositionScalefactor >GetChopPositionScalefactor	Scale factor between reference position and estimated mechanical force. Had the same function as Feed forward gain so the command is still declared but replaced in the code by a unique FeedForwardGain. Default:3051
0D7h 8D7h	>SetBEMF rate filter 1 >Get "	Digital filter coeff 1 for rate filtering of the BACK EMF effect Default: conversion of 847.458(*0.000001)
0D8h 8D8h	>SetBEMF rate filter 2 > GetBEMF rate filter 2	Digital filter coeff 2 for rate filtering of the BACK EMF effect Default: conversion of 0.69491 (*0.0001)
0D9h 8D9h	>SetC2JCrosscoupling >GetC2JCrosscoupling	Cross coupling constant between chopper and jiggle: JIGDAC=(CHOPTRAJ-0x8000)*(C2JCROSSCOUPLING-0x8000)*10e-8 (*32765) Default value: 0x8000: 0mA static compensation
0DAh	>SetC2JDCrosscoupling >GetC2JDCrosscoupling	Derivative of cross coupling Command declared but not implemented because not used.
DBh	Cancelled	
DCh	Cancelled	
DDh	Cancelled	
0DEh	Cancelled	
0DFh	Cancelled	

The following commands are **read only** for HK and telemetry data.

Mnemo	Command Mnemonic Name	Action and Parameter specification
900h	>GetBSMStatus	Reads the Chopper and Jiggle axis Activity Status Register. This Status Register can only be read and no bit can be cleared since the word is refreshed by the chipset. 2 bit : b0: Chop fatal error b1: Jiggle fatal error. Since no fatal error definition, command declared but not implemented.
901h	cancelled ~~GetCPosition~~	
902h	>GetCMeanPositionError	Returns the servo error between target and MR sensor
903h	>Get CMagnetoresistiveSignal	Returns the sensor measurement from ADC
904h	>Get CDACValue	Returns the DAC value sent to chopper motor
905h	>GetCMotorCurrent	Returns the motor current value from ADC
906h	>GetCVoltage	Returns the motor voltage signal value from ADC

4.5 JIGGLE COMMANDS

Mnemo	Command Mnemonic Name	Action and Parameter specification
140h 940h	>SetJSensorPwr >GetJSensorPwr	Power on magneto resistive sensor of the jiggle axis 0: Off 1: On
141h 941h	Free	
142h 942h	>SetJigLoopMode > GetJigLoopMode	Open or close loop mode selection Parameter definition: 0 = Loop opened : DAC steady 1 = Loop closed on magneto resistive sensor 2: not used 3 = no sensor (open loop operation using backemf)
143h 9C3h	>SetJigTargetPosition >GetJigTargetPosition	Set the position step to be reached. Unit: in closed loop: MR sensor ADC Unit to reach (1 ADU) in open loop : DAC desired value (1ADU)
144h 9C4h	>SetJigTargetPosition2 >GetJigTargetPosition2	For Jiggle tuning: allow defining a starting position when using BSM Move test (see command C6). Unit: ADC Unit
0C5h 8C5h	Not Used	
146h 9C6h	Free	
147h 947h	>SetJFFOFFSET >GetJFFOFFSET	Should allow to add an offset on DAC **Command declared but not implemented in algorithm**
148h 948h	>SetJKp >GetJKp	Proportional gain of the digital PID controller
149h 949h	>SetJKd >GetJKd	Derivative Gain of the digital PID controller (also called rate feedback gain)
14Ah 94Ah	>SetJKi >GetJKi	Integral gain of the digital PID controller
14Bh 94Bh	>SetJIntegrationThreshold >GetJIntegrationThreshold	Integration threshold (dead zone on servo error before starting integration). 0xFFFF = no gap, no threshold
14Ch 94Ch	>SetJIntegratorLimit >GetJIntegratorLimit	Integration limitation for the integral compensation of the servo
14Dh 94Dh	>SetJFeedForwardGain >GetJFeedForwardGain	Feed forward gain of open loop control. Unit: DAC value= FFGain/3051*(TargetPosition) . With Default value =3051 (0xBEB), the DAC value shall be equal to TargetPosition. (with small truncature).
14Eh 94Eh	>SetJFeedForwardDiffGain >GetJFeedForwardDiffGain	Derivative gain on the Feed forward of open loop. Default value: 0
14Fh 94Fh	>SetJDifffiltertc1 >GetJDifffiltertc1	Differential filter for PID derivative control. Unit: 10^{e-}4*coeff. Common with bemf control may be changed if no sensor mode
150h 950h	>SetJDifffiltertc2 >GetJDifffiltertc2	Differential filter for PID control. Unit: 10^{e}-1*coeff. Common with bemf control may be changed if no sensor mode
151h 951h	>SetJRateLimit >GetJRateLimit	Slew rate limiter to reduce power dissipation during step changes. Unit:10^{e}5 degrees/0.4msec.Default:1000°/sec
152h 952h	>SetJMotorBEMFGain >SetJMotorBEMFGain	Motor back emf constant, i.e. the feedback gain applied on the computed backemf damping. Default: 0
153h 953h	>SetJMotor resistance >GetJMotor resistance	Motor + harness resistance used to subtract RI value from motor voltage. BEMF= MotorVoltage-MotorCurrent*MotorResistance*0.0001. Default: 0
154h 954h	>SetJMotor inductance > GetJMotor inductance	Motor + harness inductance (normally not used)
155h	>SetJRateScalefactor	Command declared for possible future use but not implemented

Mnemo	Command Mnemonic Name	Action and Parameter specification
955h	>Get JRateScalefactor	
156h 956h	>SetJigPositionScalefactor >GetJigPositionScalefactor	Scale factor between reference position and estimated mechanical force. Had the same function as Feed forward gain so the command is still declared but replaced in the code by an unique FeedForwardGain. Default:3051
157h 957h	>SetJBEMF rate filter 1 > GetJBEMF rate filter 1	Digital filter coeff 1 for rate filtering of the BACK EMF effect Default: conversion of 847.458(*0.000001)
158h 958h	>SetJBEMF rate filter 2 >GetJBEMF rate filter 2	Digital filter coeff 2 for rate filtering of the BACK EMF effect Default: conversion of 0.69491 (*0.0001)
159h 959h	>SetJ2CCrosscoupling >GetJ2CCrosscoupling	Cross coupling constant between jiggle and chopper Default value: 0mA i.e. `0x8000`
15Ah 95Ah	>SetJ2CDCrosscoupling >GetJ2CDCrosscoupling	Derivative of cross coupling **Command declared but not implemented because not used.**

The following commands are **read only** for HK and telemetry data.

Mnemo	Command Mnemonic Name	Action and Parameter specification
980h	Free	
981h	Cancelled ~~GetJPosition~~	Cancelled
982h	>GetJMeanPositionError	Returns the servo error between target and MR sensor
983h	>Get JMagnetoresistiveSignal	Returns the sensor measurement from ADC
984h	>Get JDACValue	Returns the DAC value sent to chopper motor
985h	>GetJMotorCurrent	Returns the motor current value from ADC
986h	>GetJVoltage	Returns the ~~back emf~~ motor voltage signal value from ADC

4.6 TELEMETRY CONFIGURATION COMMANDS

4.6.1 TELEMETRY PACKETS

The MCU may generate 4 different telemetry packets delivered by the fast 1.25 Mbit/s serial link:

Packet Name	Packet Number	Sampling unit	Purpose and typical use
SMEC	0x10	N*420µs Default value: N=11	**In scientific operation** using the FTS : **FTS scan mode** : 240 Frames/sec (unit=11) **FTS Step & look mode** : 64 Frames/sec
Cancelled	0x11		
BSM	0x12	N*420µs Default value: N=42	**In scientific operation** : **Photometer Chopmode**: 64 Frames/second **Photometer non-chop** : 4 Frames/second **FTS scan mode** : 16 Frames/second **FTS step & look mode**:64 Frames/second
Cancelled	0x13		
Engineering data (alias Trace)	0x14	N*420µs Default value: N=0	For engineering additional SMEC and BSM values used for diagnostic
Test	0x15	N*420µs Default value: N=0	For link tests with DPU with a fixed pattern of the same length as Engineering data packet.

4.6.2 BUFFERING

The telemetry rate is limited to 1.25 Mbit/s so in case of fast sampling rate of data acquisition, the internal MCU buffer may be saturated.This buffer is fixed to 8K of 16 bits data. In case of buffer saturation the telemetry is stopped and the sampling intervals of all packets shall be imposed to 0 in MCU software.

Example of use:

Command to be sent	Action	Command (Hex)							
SetTelemetryPacket10sampling,10*420µs rate	Set up the SMEC science packet to 250 Hz frame rate	9	1	C	0	0	0	0	A
SetTelemetryPacket12sampling,20ms	Set up the BSM science packet to 20ms	9	1	C	2	0	0	3	2
SetTelemetryFrameNumber,inf	Unlimited number of frames	9	1	C	3	F	F	F	F
SetFrameStart,1	Start the frame emission from MCU to DPU	9	1	C	1	0	0	0	1

4.6.3 SMEC PACKET CONTENTS

In yellow : Added automatically by FPGA Generic communication interface

In green: default value only: the choice of parameter is programmable within parameter table list.

Length (12)
Frame# (0x10)
Acquisition Date MSW
Acquisition Date LSW
Optical encoder coarse position (µm)
Optical encoder fine position (nm)
LVDT position
Commanded current to motor (DAC value)
Motor Backemf value
Transmission date MSW
Transmission date LSW
Checksum (Longitudinal Parity Check)

4.6.4 BSM TELEMETRY PACKET

Length (13)
Frame# (0x12)
Acquisition Date MSW
Acquisition Date LSW
Chopper Magneto resistive Sensor Signal (ADC)
Chopper Motor Current (commanded)
Chopper Motor Voltage from ADC
Jiggle Magneto resistive Sensor Signal (ADC)
Jiggle Motor Current (commanded)
Jiggle Motor Voltage from ADC
Transmission date MSW
Transmission date LSW
Checksum (Longitudinal Parity Check)

4.6.5 ENGINEERING DATA PACKET

Length (21)
Frame# (0x14)
Acquisition Date MSW
Acquisition Date LSW
Encoder count
SMEC encoder signal 1 from ADC
SMEC encoder signal 2 from ADC
SMEC encoder signal 3 from ADC
SMEC LVDT DC signal from ADC
SMEC LVDT AC signal from ADC
SMEC Motor Current from ADC
SMEC Motor Voltage from ADC
Chopper Magneto resistive sensor signal from ADC
Chopper Motor current signal from ADC
Chopper Motor Voltage from ADC
Jiggle Magneto resistive Sensor Signal from ADC
Jiggle Motor Current from ADC
Jiggle Motor voltage from ADC
Transmission date MSW
Transmission date LSW
Checksum (Longitudinal Parity Check)

4.6.6 TEST PACKET CONTENTS

Word definition	HEX value (Decimal Value)	Binary Value
Packet length	**0x0015 (21)**	0000000000010101
Packet Id	**0x0015 (21)**	0000000000010101
Test data	**Acquisition date (MSW)**	
	Acquisition date (LSW)	
	0x5555	0101010101010101
	0xAAAA	1010101010101010
	0x5554	0101010101010100
	0xAAA8	1010101010101000
	0x5550	0101010101010000
	0xAAA0	1010101010100000
	0x5541	0101010101000001
	0xAA82	1010101010000010
	0x5505	0101010100000101
	0xAA0A	1010101000001010
	0x5414	0101010000010100
	0xA828	1010100000101000
	0x5050	0101000001010000
	0xA0A0	1010000010100000
Transmission date MSB	**Computed by FPGA**	
Transmission date LSB	**Computed by FPGA**	
Parity check	**Computed by FPGA**	

4.6.7 TELEMETRY COMMAND LIST

Mnemo	Command Mnemonic Name	Action and Parameter specification
1C0h **9C0h**	**SetTelemetryPacket10Sampling** **Get “**	Set the sampling rate of the telemetry packet#10 (SMEC Science data). Do not affect start/stop telemetry parameter 0 means no packet sent/stop the telemetry packet. Unit= number of scheduler cycle (420 µs)
1C1h **9C1h**	**SetFrameStart**	0001: the programmed telemetry frames are sent to DPU. 0000: the programmed telemetry frames are not sent to DPU
1C2h **9C2h**	**SetTelemetryPacket12Sampling** **Get “**	Set the sampling rate of the telemetry packet#12 (BSM science). (BSM Science data). Unit= number of scheduler cycle (420µs) 0 means no packet sent/stop the telemetry packet.
1C3h **9C3h**	**SetFrameNumber**	Set the total number of telemetry frames to send to DPU. FFFF means infinite number of frames. The number of frame is decreased by one after each end of whatever packet. When reaching zero, the telemetry stops with reset of FrameStart parameter.
1C4h **9C4h**	**SetTelemetryPacket14Sampling** **Get “**	Set the sampling rate of the telemetry packet#14. (Engineering data). Unit= number of scheduler cycle (420 µs) 0 means no packet sent/stop the telemetry packet.
1C5h **9C5h**	**SetTelemetryPacket15Sampling** **Get “**	Set the sampling rate of the telemetry packet#15. (Test data). Unit= number of scheduler cycle (420 µs) 0 means no packet sent/stop the telemetry packet.
1C6h **9C6h**	**SetPacket10Param#5** **Get “**	Programmable value. Unit: address in the PTA of the desired parameter. Default value: motor BEMF
1C7h **9C7h**	**SetPacket10Param#1** **Get “**	Programmable value. Unit: address in the PTA of the desired parameter. Default value: coarse encoder position (1µm counts) Example of use: 91C70040: put the SMEC encoder level parameter in Packet 10, position 1. Encoder level mnemomic is 0040.
1C8h **9C8h**	**SetPacket10Param#2** **Get “**	Programmable Default value: fine encoder position (result of arcsine)
1C9h **9C9h**	**SetPacket10Param#3** **Get “**	Programmable Default value: LVDT Position
1CAh **9CAh**	**SetPacket10Param#4** **Get “**	Programmable Default value : motor commanded current to DAC
1CBh **9CBh**	**SetPacket12Param#1** **Get “**	Programmable Default value: Chop. position (MR signal)
1CCh **9CCh**	**SetPacket12Param#2** **Get “**	Programmable Default value : Chopper motor current (value to DAC)
1CDh **9CDh**	**SetPacket12Param#3** **Get “**	Programmable Default value : Chopper motor voltage (from ADC)
1CEh **9CEh**	**SetPacket12Param#4** **Get “**	Programmable Default value : Jiggle position (MRS signal from ADC)
1CFh **9CFh**	**SetPacket12Param#5** **Get “**	Programmable Default value : Jiggle motor commanded current
1D0h **9D0h**	**SetPacket12Param#6** **Get “**	Programmable Default value : Jiggle motor voltage (from ADC)
1D1h **9D1h**	**SetPacket14Param#1** **Get “**	Programmable Default value : SMEC encoder count
1D2h **9D2h**	**SetPacket14Param#2** **Get “**	Programmable Default value : SMEC encoder signal 1 (from ADC)
1D3h **9D3h**	**SetPacket14Param#3** **Get “**	Programmable Default value : SMEC encoder signal 2(from ADC)
1D4h **9D4h**	**SetPacket14Param#4** **Get “**	Programmable Default value : SMEC encoder signal 3(from ADC)
1D5h **9D5h**	**SetPacket14Param#5** **Get “**	Programmable Default value : SMEC LVDT DC signal(from ADC)

Mnemo	Command Mnemonic Name	Action and Parameter specification
1D6h 9D6h	SetPacket14Param#6 Get "	Programmable Default value : SMEC LVDT AC signal(from ADC)
1D7h 9D7h	SetPacket14Param#7 Get "	Programmable Default value : SMEC motor current(from ADC)
1D8h 9D8h	SetPacket14Param#8 Get "	Programmable Default value : SMEC motor voltage (from ADC)
1D9h 9D9h	SetPacket14Param#9 Get "	Programmable Default value: Chop. MR Signal(from ADC)
1DAh 9DAh	SetPacket14Param#10 Get "	Programmable Default value: Chop. motor current signal(from ADC)
1DBh 9DBh	SetPacket14Param#11 Get "	Programmable Default value: Chop. Motor voltage (from ADC)
1DCh 9DCh	SetPacket14Param#12 Get "	Programmable Default value: Jiggle MR signal(from ADC)
1DDh 9DDh	SetPacket14Param#13 Get "	Programmable Default value: Jiggle motor current signal(from ADC)
1DEh 9DEh	SetPacket14Param#14 Get "	Programmable Default value: Jiggle motor voltage (from ADC)
1DFh 9DFh	GetTelemetryStatus Get "	Status of the telemetry:b0=1: Packet10 on, b2=1:Packet12 on b4=1: Packet 14 on. b5=1: Packet 15 on.

4.7 HK AND MISCELLANEOUS MCU COMMANDS

Note: the order of the HK parameters has been put in this order because it is linked to the hardware (MUX component) routed in MAC board.

Mnemo	Command Mnemonic Name	Action and Parameter specification
9E0h	GET5V	Read the +5V power supply voltage
9E1h	GETP14V	Read the + 14V power supply voltage
9E2h	GETM14V	Read the –14V power supply voltage
9E3h	GETP15V	Read the + 15V power supply voltage
9E4h	GETM15V	Read the –15V power supply voltage
9E5h	GETMACTEMP	Read the MAC board temperature probe
9E6h	GETSMECTEMP	Read the SMEC board temperature probe
9E7h	GETBSMTEMP	Read the BSM board temperature probe
1E8h 9E8h	~~SETDPUPOLLINGTIME~~ CANCELLED FOR SIMPLIFICATION BECAUSE THE WATCHDOG MAY BE DONE AT DPU LEVEL (SWITCH OFF)	~~Set a maximum time between 2 commands before a IO Error code is generated, meaning a communication problem~~
9E9h	GETERRORCODE	NOT FUNCTIONAL (See SMEC Status instead).
9EAh	GETSCHEDULERCOUNTLSW	Read MCU scheduler cycle counter (Least Sig). May be used for 420 µs cycle calibration on ground (comparison with dates)
9EBh	GETSCHEDULERCOUNTMSW	Read MCU scheduler cycle counter (Most Sig).
1ECh	SETWARMRESET	COMMAND NOT IMPLEMENTED.

4.8 ADDITIONAL COMMANDS USED FOR DEBUG ONLY

Mnemo	Command Mnemonic Name	Action and Parameter specification
1ED	**DIGOUT_STATUS**	By a get, read the contents of the DIGITAL OUTPUT Port: b0= latch command b1=latch sense b2=lvdt on/off b3 to b5= encoder level b6= chopper sensor supply on/off b7= jiggle sensor supply on/off
1EE	**TCOUNT1**	By a get returns the DSP timer counter at the end of task 1
1EF	**TCOUNT2**	By a get returns the DSP timer counter at the end of task 2
1F0	**TCOUNT3**	id
1F1	**TCOUNT4**	id
1F2	**TCOUNT5**	id
1F3	**TCOUNT6**	id
1F4	**TCOUNT7**	id
1F5	**TCOUNT8**	id
1F6	**TCOUNT9**	id
1F7	**TCOUNT10**	id
1F8	**TCOUNT11**	id
1F9	**TCOUNT12**	id
1FA	**TCOUNT13**	id
1FB	**TCOUNT14**	id
1FC	**TCOUNT15**	id
1FD	**TCOUNT16**	id
1FE	**TCOUNT17**	id
1FF	**TCOUNT18**	id
200	**TCOUNT19**	id
201	**TCOUNT20**	id (inhibited)
202	**SINE_COMPUTED_OFFSET**	By a get see the intermediate result of encoder offset computation
203	**SINE_COMPUTED_AMP**	By a get see the intermediate result of encoder amplitude computation
204	**SINE120_COMPUTED_OFFSET**	id for sine 120
205	**SINE120_COMPUTED_AMP**	id for sine 120
206	**SCRATE_LIMITER**	Slew rate limitation of the current offset applied on the SMEC
207	**HKMUX**	By a get see the state of the HK ADC MUX
20C	**SINE_DEADZONE**	Define a dead zone on ADC encoder signal to avoid miscounting during zero crossing (cancellation of noise)
20D	**SINE120_DEADZONE**	id for sine 120 signal
20E	**CHOPSIGN**	Allow to redefine the polarity of the MR sensor in case of wrong cabling
20F	**JIGSIGN**	id.

5 MAIN FORMULAE

5.1 SMEC

The current send to motor is:

DAC= ffoffset + feedforward - backemffeedback + PID (in closed loop mode only)

DAC unit: 65535= 100mA; 32768=0mA; 0= -100mA

- **ffoffset= FFOFFSET** (Note: the ffoffset change is limited in slew rate by the parameter SMEC_BEMF_RATE)
- **feedforward= TargetPosition*FFGAIN *32768*10^{e}-9**
- **backemffeedback= (((Motor_Voltage_ADC-0x8000) - (Motor_Current_ADC-0x8000)*Motor_Resistance *0.0001)* SMEC_BEMF_CONSTANT*0.0000001)* 32767 +32767**
- **PID = positionerror(t) * Kp + ((positionerror(t)-positionerror(t-1))+ Kdeffect(t-1)* SDERIVFILTER*0.0001)*Kd*0.00001 + Ki *0.000001*sum(position(t-i))**
  - **positionerror**=TargetPosition-SensorPosition (micron)
  - **sum(position(t-i))** is the sum of all samples of positionerrors limited by **SINTEGRATION_LIMIT**
  - The sum is produced only if the sum exceeds **SINTEGRATION_THRESHOLD**.

NOTE: Note: See smecpid.asm source listing for precise algorithm.

Encoder signal computation:

Normalised sine= (encoder sine signal - sine offset)/sine amplitude to get a -1 +1 normalised signal.
The count is done on each 0 crossing of the sine signal 1. To start counting, the user must program an offset in order to get a centered signal around 0 to allow zero crossing detection. Then the offset and amplitude is automatically re-calculated during the counting to adjust new values depending on signal 1 and 2 offset and amplitude evolutions.
The count or decount is decided depending on signal2 sign. When the loop is closed, the encoder counter is automatically reset to trajecctory position value (so the servo error is zero) to avoid transcient currents with the risk of shocks on mechanism and loss of encoder counts.

Then a 500 points arcsin LUT produces a fine position (nm unit) on the normalised sine signal1

LVDT:

computed LVDT position(micron) = (LVDT ADC-0x8000) *LVDTSCALE *0.00002 +LVDT_OFFSET. Unit:1µm
(LVDT scale to be programmed is= delta (µm encoder counts)/ delta(LVDT ADU) * 50000)) in its linear range.

5.2 BSM

For each chopper/jiggle axis:

DAC= feedforward - backemf_feedback + PID (in closed loop mode only) + crosscoupling

- **crosscoupling on chopper= (J2CCROSSCOUPLING-0x8000)*0.00000001*(jigtraj-0x8000)**
- **feedforward (chopper)= (choptraj- CFF_OFFSET)* CFFGAIN*32768*10e-8**

(For jiggle the feedforward is: (jigtraj- 0x8000)* JFFGAIN*32768*10e-8)
backemf: filtered with digital filter (CBEMF_RATE_FILT1 and CBEMF_RATE_FILT2 coefficients) is:

- **(((Motor_Voltage_ADC-0x8000) - (Motor_Current_ADC-0x8000)*Motor_Resistance *0.0001)* CHOP_BEMF_GAIN*0.0000001)* 32767 +32767 (Motor inductance not to be used and set to 0)**

- **PID = positionerror(t) * Kp + ((positionerror(t)-positionerror(t-1))* CDIFF_FILTER_TC2*0.1 + Kdeffect(t-1)* CDIFF_FILTER_TC1*0.0001)*Kd*0.0000000001 + Ki *0.000001*sum(position(t-i))**

  - **positionerror**=TargetPosition-SensorPosition in sensor ADC unit
  - **sum(position(t-i))** is the sum of all samples of positionerrors limited by **INTEGRATION_LIMIT**
  - The sum is produced only if the sum exceeds **INTEGRATION_THRESHOLD**.

Note: See choppid.asm and jigpid.asm source code files for precise algorithm.

--- end of document ---