Original draft November 2002
Issue 0.3 April 2003
Commanding changed in line with
DRCU/DPU Interface Document - SPIRE-SAP-PRJ-001324 issue 1.0 14/02/2003
MCU/DPU Comand List ICD - LAM/ELE/SPI/011011 Issue 3.0 15/01/2003
SCU Design Description - SEDI-SCU-MM-2002-1 0.7 17/02/2003
DCU Design Description - SAp-SPIRE-FP-0063-02 Issue 0.3 18/02/2003

Issue 1.0 11 June 2003
MCU Commanding/reaction section completed
DCU Commanding for Photometer completed

SCU

General description:
The SCU provides power; signal conditioning and data acquisition for the heaters and thermistors on the SPIRE FPU. This includes the heaters necessary for the 300 mK cooler operation; the heaters for the calibration sources and all temperature sensors on the cold FPU except those associated with the detectors. The unit is powered on when the 28 V power is applied to the DRCU power supply unit. At power on the FPGA enters a reset state, boots in some fashion, counts its fingers and sets the ScuStatus word to some TBD value. It is then ready to accept commands from the DPU.

In this version of the document I have used guessitimated conversion curves – CEA have now provided more accurate simulated curves which I will use in future. I have still left out the detector stuff as this requires more work.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
Request to CDMS to switch on 28V line to DRCU	ScuStatus	Undetermined	0x0000	If any of first three bits set then an error has occurred
	ScuCHTp05	Undetermined	0xB554	Used 0.366 mV/ADU over +-12V with 2^{15}-1 = 0V
	ScuCHTp09	Undetermined	0xE00D	Ditto
	ScuCHTn09	Undetermined	0x1FF0	Ditto
	ScuCHTp25	Undetermined	0x9AAD	Ditto
	CsuTempRd	Undetermined	0x4001	Complete guess Need conversion curve between ADC value and temperature
	TsuTempRd	Undetermined	0x4002	Guess again – bit different to make sure in right place
	PsuTemp1Rd	Undetermined	0x4004	Ditto
	PsuTemp2Rd	Undetermined	0x4008	Ditto

Telemetry
The SCU "science" data nominally consists of 17 temperatures plus the currents and voltages for the three calibrators. There are two frame types defined – nominal and test. The frame type, sampling rate and the number of frames to send are set up by command before a request to start sending data. One can either request a given number of frames or continuous data transfer. In the latter case a command must be sent to stop the data transfer at the appropriate time.

Command sent	**Parameter that should respond**	**Status/value before**	**Status/value after**	**Comment**
SetFrameConf	FrameConf	0x0000 (TBC)	0x004F	Bit 15 is type 0 is normal 1 is test Bits 0-7 set rate as 80/(value+1) Hz Sets 1 Hz sampling normal frame
SetSeqLength	SeqLength	0x0000	0x001F	Sets 31 frames to be sent
SetFrameCtrl	FrameCtrl	0x0000	0x0001	Bit 0 starts and stops frame transfer – reset at end of 31 frames (TBC)
	ScuStatus	0x0000	0x0004 during transfer 0x0000 at end of transfer	Bit 4 is a mirror of FrameCtrl bit 0

Thermistors
General Description:
There are 16 "normal" thermistors on SPIRE. These are switched on by applying a small, stabilised current and the temperature is calculated from measurement of the voltage across the devices. Each of the sixteen temperature channels is individually commanded on and off and the corresponding parameter will change from 0 to a value dependent on the temperature as the current is applied.

Command sent	**Parameter that should respond**	**Status/value before**	**Status/value after**	**Comment**
TempOnOff 0xFFFF	TempOnOff	0x0000	0xFFFF	16 bits of on/off to address 16 temp channels. All temps on.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
	CPHPtemp CPHStemp CEHStemp CSHTtemp SOBtemp SL0temp PL0temp SUBtemp BAFtemp BSMStemp SCL2temp SCL4temp SCSTtemp FTSStemp FTSMtemp BSMMtemp	All 0x0000	0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x0DAC (ltg) 0x0DAC (ltg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg)	In fact the response curves are non-linear – to make things easy I have two thermistor temepratures following the SCU design description – Low Temperature Group (ltg) with 0.5 mK/Bit linear over 1-2 K range High Temperature Group (htg)with 2 mK/bit over the 3-6 K temperature range I have ignored offsets and thermistors will move from one group to another depending on the temperature
TempOnOff 0x0000	TempOnOff	0xFFFF	0x0000	Turns temp sensor N on or off – send this command for each or does parameter address lots of them?
	CPHPtemp CPHStemp CEHStemp CSHTtemp SOBtemp SL0temp PL0temp SUBtemp BAFtemp BSMStemp SCL2temp SCL4temp SCSTtemp FTSStemp FTSMtemp BSMMtemp	0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x0DAC (ltg) 0x0DAC (ltg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg) 0x08CA (htg)	All 0x0000	Need conversion curve for ADC value to temperature for Cernox 1030's *Need table of which one is which.*

PCAL

General description:

There is only one nominal PCAL operational mode. When the PCAL heater current is commanded there is nominally no reaction from any other sub-system parameter except that the detectors will give a signal. The waveform of the PCAL output is controlled by sending successive commands to set the current.

The PCAL heater supply is "four wire" which allows direct measurement of the voltage across the devices. This parameter is reported in the housekeeping as well as the commanded set current.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetPhCalBias 0x07FD	PhCalBias	0x0000	0x07FD	Assume 4.5 mA full scale for 4095 steps 1.1 µA/step This sets current to half way ~2.25 mA
	PhCalVolt	0x0000	0x08BC + 0x1721*(1-e$^{t/0.1}$) Where t is time in seconds	I have bodged this from reading the SCU and PCAL design description using 76 µV/bit to describe a change in voltage given by V(t)= 0.17+0.45(1-exp(t/0.1).

SCAL

General Description

There is only one nominal SCAL operational mode. When the SCAL-2 or SCAL-4 current is commanded the SCAL-2 or 4 thermistor responds with a TBD time constant as the calibrator warms up. The temperature is achieved as quickly as possible and then stabilised using software control in the DPU. The power falling on the detectors will increase with the SCAL temperature but this is only DC power so is not seen as signal – only increased noise. Before the thermistors can respond they must be switched on.

The SCAL-2 and SCAL-4 heater supplies are "four wire" which allows direct measurement of the voltage across the devices. This parameter is reported in the housekeeping as well as the set current.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetSCal4Bias 0x068F	Scal4Bias	0x0000	0x068F	Set up for 5.5 mA full scale over 4095 steps 1.34 µA/step This sets 2.25 mA – equivalent to ~2.5 mW dissipation in source.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
	SCal4Volt	0x0000	0x08BC + 0x23FA*$(1-e^{t/5})$ t is time in seconds	Assume same as PCAL for the time being with a 5 second time constant (wrong but otherwise we wait for ever!). In dec the voltage is V(t) = 0.17+0.7(1-exp(t/5))
	SCL4temp	0x08CA	0x08CA + 0x9376*$(1-e^{-t/5})$	Response curve is actually non-linear – here I have assumed 2 mK/Bit linear from 4.5 K to 80 K.
SetSCal2Bias 0x068F	Scal2Bias	0x0000	0x068F	Same as SCAL-4
	Scal2Volt	0x0000	0x08BC + 0x23FA*$(1-e^{t/5})$ t is time in seconds	Ditto
	SCL2Temp	0x08CA	0x08CA + 0x9376*$(1-e^{-t/5})$	Ditto

Cooler

General Description:

The cooler has two operational modes – recycle and cooling. During the recycle mode the current to the evaporator heat switch is set high, the evaporator thermistors, both “sub-K” and “normal”, will read 1.7 K. At the same time, the current to the pump heat switch is set low. The pump heater current is set under software control to keep the temperature of the pump, registered by the pump thermistor, at TBD K for some length of time (about 30 minutes). At the end of this period the current to the pump heat switch is set high and (a short time later) the current to the evaporator is set low. Following this the pump temperature will fall to ~1.8 K and the evaporator temperature will fall to ~300 mK – this is the cooling operating mode condition. In table below the command sequence is set out for recycling the cooler and leaving it in the cold condition.

All the heater supplies are “four wire” which allows direct measurement of the voltage across the devices. Only the voltage is reported in the housekeeping and NOT the set current because the same parameter name/address is used for the Set and Get commands for the heater control – see comment below.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetSubKOnOff 0x0001	SubKOnOff	0x0000	0x0001	Switches on the sub-K thermistor

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
	SubKTempP	0x0000	0x05DC	When the cooler is first turned on the evaporator (where the sub-K sensor is located) will be at the L1 temperature (or thereabouts) The SubK is highly non-linear – at 1.75 and 4.5 K I assume 3 mK/bit – see below for strange response curve
SetEVHSHeatCur 0x0800	EVHSHeatCur	0x0000	0x0800	This closes the evaporator heat switch connecting it directly to L0. Complete guesswork for the value here!
	EVHSHeatVolt	0x0000	0x61F7	SCU design desc. gives voltage across the switch is given by $V=9.31e^{-4}$*DAC $+2.57e^{-4}$
	SubKTempP	0x05DC	0x0247	Evaporator falls to L0 temperature. Assume instantly for these purposes. Again 3 mK/bit
SetSPHeaterCur 0x0708	SPHeaterCur	0x0000	0x0708	At the beginning of the recycle the heater is set to 200 mW - about 22 mA (TBC). I assume 1.22e-5 A/DAC bit
	SPHeaterVolt	0x0000	0x1C30	Can take ADC= $3280V_{sense}$ from CEA document – this equivalent to 2.2 V as given in SCU Design Desc. In fact it should follow the resistance but this will do.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
	CPHPtemp	0x08CA	0x08CA + 0x4556(1-e$^{t/300}$)	For now can assume an RC response with time constant of 5 minutes to get to 40 K. Assume thermistor value is given by 2 mK/bit linear from 4.5 to 40 K.
SetSPHeaterCur 0x0287	SPHeaterCur	0x0708	0x0287	Once the pump gets hot (>40 K) the current is backed off to keep it at a (more or less) fixed temperature. Here I assume backed to 25 mW or 7.9 mA – relationship as before.
	SPHeaterVolt	0x1C30	0x07B0	This equivalent to 0.6V as given in SCU Design Desc.
	CPHPtemp	0x4E20	0x4E20	Assume for now that the system behaves perfectly – i.e. it gets to 40-K in 5 min and just sits there when the current is backed off.
SetSPHeaterCur 0x0000	SPHeaterCur	0x0287	0x0000	After some time the heater is switched off
	SPHeaterVolt	0x07B0	0x0000	Voltage also falls to zero
	CPHPtemp	0x4E20	0x4E20	Once the heat is removed the pump will actually cool – but very slowly – ignore this for now.
SetSPHSHeatCur	SPHSHeatCur	0x0000	0x0800	This closes the pump heat switch connecting it directly to L0. Complete guesswork for the value here!
	SPHSHeatVolt	0x0000	0x61F7	SCU design desc. gives voltage across the switch is given by V=9.31e^{-4}*DAC +2.57e^{-4}

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
	CPHPtemp	0x4E20	0x4E20 – 0x4AB5(1-$e^{-t/60}$) t is time in seconds	When the heat switch closes the pump temperature falls to 1.75 K with some time constant – here assumed as 1 minute – 2mK/Bit over 40 to 1.75 K range
SetEVHSHeatB 0x0000	EVHSHeatCur	0x0800	0x0000	This opens the evaporator heat switch allowing the evaporator temperature to fall as the ^{3}He is pumped.
	EVHSHeatVolt	0x61F7	0x0000	As current is switched off voltage falls to 0 as well
	SubKTempP	0x06D6	1.75 – 1.45(1 – $e^{-t/60}$) This is dec see below for hex conversion	Evaporator falls to 300 mK with some time constant – assume 1 minute – not correct but we can't wait all day!. The resolution versus temperature is weird – see below – and should be replicated as the temperature will appear to rise before it falls! This is the "cooling" operating condition – these values are maintained during all observations.

SubKtemp conversion curve

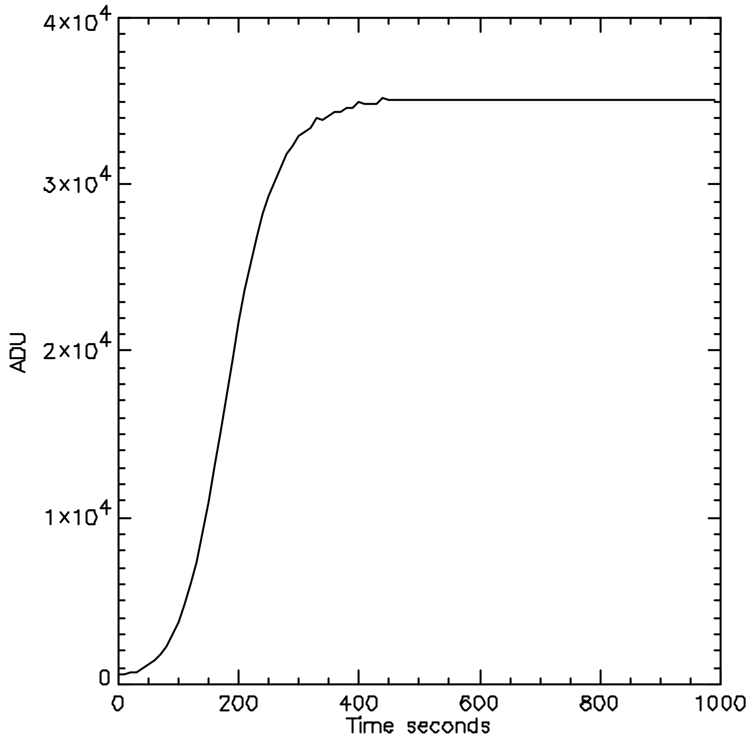
Coefficients for resolution versus temperature (in IDL syntax) are approximately:

cf=[0.032015060,-0.0039288996,-1.1377826,3.3303894,-1.2193793]

So the temperature resolution will follow this curve as a function of temperature (in K)

mK/bit(T)= cf(0)+cf(1)*htemp+cf(2)*htemp^2+cf(3)*htemp^3+cf(4)*htemp^4

This combined with the exponential decay in temperature results in the response in terms of ADU as a function of time looking like this.



Simulated response of the SubKtemp parameter as the evaporator cools to 300 mK. ADU is dec here – convert to Hex for simulator output.

MCU

General Description:
The SMEC and BSM are controlled by a motion control unit based on a DSP – this appears as hardware to the outside world although it has a reasonably sophisticated "firmware" system for motion control. The first operation for the MCU is to switch it on and get it initialised. The unit is powered by a command to the SCU to switch supply from the power supply unit to the MCU. The DSP is commanded to boot from PROM and the boot status word is set. The unit is now ready to accept commands from the DPU.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
DrelOnOff 0x0004 *This is a command to the SCU*	P5V	Undetermined	0xA804	Assume MCU voltages digitised to ±16 V fullscale this is 0.488 mV/bit with 0V = 0x7FFF

Command sent	**Parameter that should respond**	**Status/value before**	**Status/value after**	**Comment**
	P15V	Undetermined	0xF810	Ditto
	M15V	Undetermined	0x07ED	Ditto
	P13V	Undetermined	0xE80E	Ditto
	M13V	Undetermined	0x17EF	Ditto
	MACTemp	Undetermined	0x1000	Same as for SCU board temps
	SMECTemp	Undetermined	0x1000	Ditto
	BSMTemp	Undetermined	0x1000	Ditto
SetDownLoadConf 0xC000 *Wait 10 seconds to simulate boot of DSP*	BootStatusRegister	Undetermined	0x0004	Set to this value during simulated upload
After 10 seconds	BootStatusRegister	0x0004	0x0001	Simulates successful boot from PROM
SetBootRAM 0x0001	BootRAM	0x0000	0x0001	Launches program from RAM

<u>Launch Latches</u>
The SMEC has a physical movement restriction provided by a solenoid type latch. This must be activated and moved to an “unlatched” position before the mechanism can be used. A dedicated command is provided for each latch to engage/disengage it. The status of the latch is set following the command to be the same as the command – there is no independent confirmation of the latch status except through EGSE.

Comments:
The BSM latch will not be fitted so the command is now redundant.
Need to make the inadvertent use of the “latch engage” and “latch disengage” commands difficult or impossible. Methods of doing this could be a double command scenario or by only allowing the commands when the SMEC status shows the SMEC is in the correct position. This is being discussed as part of the OBS specification.

Command sent	**Parameter that should respond**	**Status/value before**	**Status/value after**	**Comment**

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetSLaunchLatch 0x0001	SlaunchLatch	0x0002 *Disengaged*	0x0001 *Engaged*	The parameter that is read back is a reflection of the command sent – NOT an indication that anything has actually happened!
SetSLaunchLatch 0x0002	SetSlaunchLatch	0x0001 *Engaged*	0x0002 *Disengaged*	*Ditto*

Telemetry

The data sampling and collection is initiated independently from the mechanism operation. The sampling rate is determined as a fixed number of cycles of the basic DSP software cycle time of 360 µsec. There are two packet types for nominal operation “10” which is the SMEC data and “12” which is the BSM data. During nominal scan mode operations we expect to sample the SMEC packet at 240 Hz (11 DSP cycles) and the BSM packet at up to 64 Hz (44 DSP cycles) *(assume just these two speeds for now – different ops. Modes may need other sampling rates)*. The values of the parameters in the packets should reflect what is going on with the subsystem – so during scanning for instance the encoder and LVDT position values should increase in the same manner as the housekeeping values.

The appropriate frame type and the transfer of the frames should all be set up and running before the sub-system is operated. If continuous data transfer has been requested it must be stopped by command at the appropriate time – i.e. the end of operations. In nominal operations the DPU command sequences will take care of all this.

Here are the command responses expected for setting first the SMEC packet followed by setting the BSM packet followed by setting no packet transfer.

This is the commanding scheme as defined in the MCU ICD v3.0 and DRCU/DPU ICD v1.0 and therefore implemented in the QM0 MCU. We are seeking a change to this command scheme to the one defined in the MCU ICD v3.1 which separates the definition of the sampling frequency from the action starting the frame transfer.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetTP10SampFreq 0x000B	TP10SampFreq	0x0000 *Packet 10 not being sent*	0x000B	This sets 240 Hz sampling of the SMEC packet. The packets start being sent when this command executes

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
	TelemetryStatus	0x0000 *No packet being sent*	0x0001 *Packet 10 is being sent*	This is a new parameter since last issue – I assume the bit pattern is: 0 – TP10 on/off 1 - TP12 on/off 2 – TP14 on/off 3 – TP15 on/off
SetTP10SampFreq 0x0000	TP10SampFreq	0x000B *Packet 10 is being sent*	0x0000	This turns off packet 10 transmission
	TelemetryStatus	0x0001 *Packet 10 is being sent*	0x0000 *No packet being sent*	See above
SetTP12SampFreq 0x002C	TP10SampFreq	0x0000	0x002C	This sets ~64 Hz sampling and the packets start being sent when this command executes
	TelemetryStatus	0x0000 *No packet being sent*	0x0002 *Packet 12 is being sent*	See above
SetTP12SampFreq 0x0000	TP10SampFreq	0x002C	0x0000	This sets ~64 Hz sampling and the packets start being sent when this command executes
	TelemetryStatus	0x0002 *Packet 12 is being sent*	0x0000 *No packet being sent*	See above

<u>SMEC</u>

The SMEC has two nominal operating modes: step and look and rapid scan. In both cases, the mechanism must first be initialised by switching it on and finding the home position.

<u>Initialisation</u>
The first stage of initialisation is to apply power to the various parts of the SMEC system – namely:

> The optical encoder pre-amplifier *(I can't see the command to do this – I assume it comes on with the power up?)*
> The optical encoder LED
> The LVDT
> The motor drive current amplifier

To do this safely the system must also be commanded definitively into open loop control mode. Once the power has been applied the SMEC can be sent to home position which I assume to be at –3.2 mm from the ZPD. I also assume that the encoder increment step zero is at the mechanical end stop which is –5 mm from the ZPD. These values are almost certainly wrong but serve to illustrate the operation. Finding the home position is an automatic procedure within the MCU unit.

Switching on the optical encoder LED will cause a signal to be seen on each of the three photodiodes within the optical encoder. If the encoder is switched on successfully the SMEC status Encoder Signals bit is set high.

Switching on the LVDT will lead to a signal on the LVDT AC and DC signals. There is a more-or-less linear relationship between the LVDT signal and position within a ±4 mm range around ZPD Outside of this range the relationship is highly non-linear (see appendix with a note on this from Didier). For the present purposes I assume that the signal saturates at the 4 mm value at either end of the ±4 mm range.

With everything switched, on the SMEC is commanded to initialise itself; the result is that the SMEC ends up at the Home position and the MCU knows where it is in terms of encoder position. After the SMEC has been sent to the home position the optical encoder and LVDT position should have definitive values and the SMEC status Home bit is set high.

Once the SMEC has been initialised it can be commanded to closed loop mode on the optical encoder. Doing this causes the SMEC status Open/Closed Loop bit to be set high as long as there is no error status flag set *(this needs more description/explanation from Didier – for present purposes can we assume that the SmecStatus bit 1 has to be 0 – i.e. no error flag is set? Should other things also be checked – like the encoder is set correctly etc).*

Operation

Once the initialisation is completed successfully the SMEC operation is straightforward. In scan mode the start and stop positions; the velocity and the number of scans to execute are commanded separately followed by a command to commence scanning in the desired mode (saw tooth or triangular). As the mechanism scans, the motor current, back e.m.f and the encoder position will follow the mechanism motion. The SMEC Status Scientific Operation bit is set high for the duration of the operation. When the SMEC position is within the range of the LVDT, the LVDT Position value will also follow the motion.

In step mode the stop position is commanded followed by an instruction to move. This causes the SMEC to move as fast a possible to the commanded position. The encoder position will change to the commanded position and the motor current and back e.m.f will also respond. When the SMEC position is within the range of the LVDT, the LVDT position will also change. *For the present purpose we can assume that the LVDT and encoder will give the same position.*

Version 0.3 Notes:
I have made up all the conversion curves based on what I have from Didier from the benchtop testing. It may not be quite right but it will do for the present purposes.

Questions/comments:
There ought to be a DPU housekeeping parameter to indicate that packets are being received from the MCU.
How does Home position get found following scan mode or step and look operation? Can the mechanism be commanded “Go Home” with the status bit being set as necessary.
Also (see below) the step and look operation needs some more thought – we need to be able to command the number of repeat scans (if any) and to indicate – by the status bit? – that the SMEC is still in scientific operation mode during the whole of a step and look scan.

The SMEC status definition used here is the old one – Didier has proposed a new definition which I'll use when it has been agreed.
Need to sort out how the Error Code is used to indicate that all is well – this is still not described in the ICD – note that the new SMECStatus proposal uses Status bits to indicate error conditions – the error code could be suppressed?

Command sent	**Parameter that should respond**	**Status/value before**	**Status/value after**	**Comment**
SetSEncoderPwr 0x0003	SencoderPwr	0x0000	0x0003	Sets encoder LED to ~half power
	SmecStatus	0x0000	0x0080	Bit 7 is set to show that the encoder LED has been switched on.
	SEncoderSignal1	0x0000	0x7FFF	Completely made up see simulated conversion curve below!
	SEncoderSignal2	0x0000	0xC9E8	Completely made up see simulated conversion curve below!
	SEncoderSignal3	0x0000	0x3616	Completely made up see simulated conversion curve below!
SetSLVDTPwr 0x0001	LVDTPwr	0x0000	0x0001	Switches power on to LVDT
	LVDTAC	0x0000	0x1DB7	Conversion curve is as defined below taken from DF's note. Conversion is assumed to be valid only between ±5 mm from ZPD. This is smoothed signal
	LVDTDC	0x0000	0x1DBA	This is noisy version of LVDT signal – simulated as "slightly different" from LVDTAC
	LVDTposition	0x0000	0x0000	LVDT position will read full negative range until the SMEC is initialised (?)
SetStrajMode 0x0004 (*Cross check with Didier)*	StrajMode	0x0000	0x0004	Sets the mode to initialisation. Not sure if the value 4 is set or bit 4 is set?

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
	SmecStatus	0x0080	0x0188	Mechanism ends up at Home (bit 3?) Encoder has been initialised (bit 8?)
	SEncoderSignal1	0x7FFF	0xD527	Completely made up see simulated conversion curve below!
	SEncoderSignal2	0xC9E8	0x5A4F	Completely made up see simulated conversion curve below!
	SEncoderSignal3	0x3616	0x5085	Completely made up see simulated conversion curve below!
	LVDTAC	0x1DB7	0x315F	Home position is 1800 microns from mech. End stop (-3.2 mm from ZPD)
	LVDTDC	0x1DBA	0x3165	Bit different just for the sake of it.
	LVDT Position	0x0000	0x0709	Ah! How does this get counted? I am assuming in microns from the mechanical reference to keep it in line with the optical; encoder. This is 1801 dec.
	SMotorCurrent	0x0000	0x03E8	Completely fictitious assuming value 1000 at home rising to 50000 at full scale
	SBEMF	0x0000	0x03E8	Total nonsense but assume same as current
	EncoderIncrPosition	0x0000	0x0708	This is position 1800 – as for LVDTPosition
	EncoderFinePosition	0x0000	0x0000	This is the position at 10 nm/bit – assume its zero at Home
SetSLoopMode 0x0001	SloopMode	0x0000	0x0001	Loop is closed on the encoder using signals from 0th and 1st order.
	SmecStatus	0x0188	0x0588	Status bit 10 is set?

The SMEC is now initialised at HOME position 0 the next table deals with commands for the scan and step and look modes of operation.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetSScanFwdSpeed	SScanFwdSpeed	0x0000	0x1388	Set the speed for the forward going scans. Value set as 0.1 µm/s per bit – set here to 500 µm/s
SetSScanRevSpeed	SScanFwdSpeed	0x0000	0x1388	Ditto for the reverse going scans. Set at the same speed.
SetSScanNumber 0x0002	SScanNumber	0x0000	0x0002	Sets number of scans to do 2 here (one back one rev?) Default value is for infinite?
SetSTrajStartPosition 0x076C	STrajStartPosition	0x0000	0x076C	Start scan at –3.1 mm from ZPD – step 1900
SetSTrajEndPosition 0x1FA4	STrajEndPosition	0x0000	0x1FA4	Scan to +3.1 mm from ZPD – step 8100
SetStrajMode 0x0002 (Forward and reverse scanning)	StrajMode	0x0000	0x0002	Not sure if value 2 is set or bit 2 is set
	SmecStatus	0x1088	0x9080 during operation 0x1088 at end of operation	Mechanism starts and ends up at Home (*correct?*) Also bit 16 gets set during operation to indicate something is going on
	SEncoderSignal1	0xD527	Follows position with D=1900 at start and D=8100 at end incrementing by 500 per sec	Completely made up see simulated conversion curve below!
	SEncoderSignal2	0x5A4F	Follows position with D=1900 at start and D=8100 at end incrementing by 500 per sec	Completely made up see simulated conversion curve below!
	SEncoderSignal3	0x5085	Follows position with D=1900 at start and D=8100 at end incrementing by 500 per sec	Completely made up see simulated conversion curve below!

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
	LVDTAC	0x315F	Goes to 0x33D4 at start of scan then Increases at 3145 ADUs/sec until 0xCC2A then decreases at 3145 ADUs/sec until 0x33D4. At end of operation goes to 0x315F	At Home at start and end of operation.
	LVDTDC	0x3165	As above with some noise added!	At Home at start and end of operation.
	LVDT Position	0x0709	Goes to 0x076D at start of scan then Increases at 500 ADUs/sec until 0x1FA5 then decreases at 500 ADUs/sec until 0x076D. At end of operation goes to 0x0709	Starts and finishes at home.
	SMotorCurrent	0x03E8	Goes to 0x041F at start of scan then Increases at 700 ADUs/sec until 0x2607 then decreases at 700 ADUs/sec until 0x041F. At end of operation goes to 0x03E8	*This is fantasy again!* Need real conversion curve between motor current and physical position
	SBEMF	0x03E8	Goes to 0x041F at start of scan then Increases at 700 ADUs/sec until 0x2607 then decreases at 700 ADUs/sec until 0x041F. At end of operation goes to 0x03E8	*This is fantasy again!* Need real conversion curve between motor back e.m.f and physical position

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
	EncoderIncrPosition	0x0708	Goes to 0x076C at start of scan then Increases at 500 ADUs/sec until 0x1FA4 then decreases at 500 ADUs/sec until 0x076C. At end of operation goes to 0x0709	Start and finishes at home
	EncoderFinePosition	0x0000	Increments at 50000 ADU/sec for forward scan then decrements at 50000 ADU/sec for reverse scan Reset to 0 at end of scan	This is the position at 10 nm/bit – assume its zero at Home – increments to 2^{16}-1 then wraps round. *Not sure what happens at home position – I assume it gets reset – this is convenient for testing here anyway.*
	ActualVelocity	0x0000	0x1388 during scan in both directions	*In reality it will have some noise should add random noise with amplitude of 10* *How do we indicate scan direction – should this value have a sign bit?*
	MeanSpeed	0x0000	0x1388 during scan in both directions	*In reality it may have some noise – this is o.k. for now* *How do we indicate scan direction?*
SetSTrajMode 0x0001 (Moves mechanism to value set by StrajEndPosition)	STrajMode	0x0000	0x0001	Not sure if value 1 is set or bit 1 is set
	SmecStatus	0x1088	0x9080 during operation 0x1088 at end of operation?	Indicates mechanism is doing something and then that its idle *This needs to change we need a bit to be set during whole of step and look scan?*

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
	SEncoderSignal1	0xD527	0x580D (D=8100)	Completely made up see simulated conversion curve below!
	SEncoderSignal2	0x5A4F	0x52A8	Completely made up see simulated conversion curve below!
	SEncoderSignal3	0x5085	0xD548	Completely made up see simulated conversion curve below!
	LVDTAC	0x315F	0xCC2A	Starts at home goes to end of scan position
	LVDTDC	0x3165	0xCC2A	Starts at home goes to end of scan position
	LVDT Position	0x0709	0x1FA5	Starts at home goes to end of scan position
	SMotorCurrent	0x03E8	0x2607	Comment as before
	SBEMF	0x03E8	0x2607	Comment as before
	EncoderIncrPosition	0x0708	0x1FA4	Starts at home goes to end of scan position
	EncoderFinePosition	0x0000	Set to random vale between 0 and 2^{16}-1	This is the position at 10 nm/bit – assume its zero at Home – and we get some jitter for the position setting.

Encoder signals simulation

I assume that the three encoder signals respond in the following fashion as a function of distance, D, from the mechanical reference in microns (psuedo code):

ADU = 2^{15}-1 + (sin(πD/2+ { case encoder eq 1,2,3 (0, 2π/3, 4π/3) })/4.577x10^{-5})

LVDT Signal Simulation

Taken from Didier Ferrand's note on the bench measurements of the Schaevitz MHR100. The ADU for the signal is linear (ish) over the range from –4 mm to +4 mm from the centre of the LVDT range

ADU = (2^{15}-1) + 6.29 (D-5000)

D is the distance from mechanical end stop –i.e. encoder increment step 0. ZPD is supposed to be at 5 mm (or 5000 steps) from the mechanical stop.

BSM

The Beam Steering Mirror (BSM) has two separately controlled axes – CHOP and JIGGLE. It is controlled by directly commanding the position of either axis from the DPU. Once a position is commanded the MCU controls the motion to get to the given position. The chop mode of the instrument is carried out by sending a series of commands to the MCU from the DPU – i.e. there will be no autonomous chop operation – unlike the SMEC scanning.

Command sent	**Parameter that should respond**	**Status/value before**	**Status/value after**	**Comment**
SetCSensorPwr 0x0001	CsensorPower	0x0000	0x0001	Turn on the chop sensor
	ChopStatus (bit N)	0	1	*Bit values not yet defined?*
	CMagnetoresistiveSignal	0x0000	0x8000	The actual value of the sensor output Assumes starts at middle of range
SetJSensorPwr 0x0001	JsensorPower	0x0000	0x0001	Turn on the jiggle sensor
	JigStatus (bit N)	0	1	*Bit values not yet defined?*
	JMagnetoresistiveSignal	0x0000	0x8000	The actual value of the sensor output Assumes starts at middle of range
SetBSMMove 0x0000	BSMMove	Undefined?	0x0000	This means the axes move independently when the TargetPosition is set. *What is the value of this when the MCU boots? I think this should be the default and therefore this command is redundant?*
	ChopStatus(Bit N)	0	1	Bit set to reflect operating mode?
	JigStatus(Bit N)	0	1	Bit set to reflect operating mode?
SetChopTargetPosition	ChopTarget Position	0x0000	0x8000	Set the desired position to the middle of the range as well

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetJigTargetPosition	JigTarget Position	0x0000	0x8000	Set the desired position to the middle of the range as well
SetChopLoopMode 0x0001	ChopStatus (Bit N)	0	1	This closes the servo loop between the motor drive and the position sensor – if all is working properly nothing should happen! *There should be a Status bit that tells us this has happened successfully - Bit values not yet defined?*
	CmeanPositionError	Undefined	0x0000	As the loop closes the error signal should be 0? *What is the value before the loop is closed?*
SetJigLoopMode 0x0001	JigStatus (Bit N)	0	1	This closes the servo loop between the motor drive and the position sensor – if all is working properly nothing should happen! *There should be a Status bit that tells us this has happened successfully - Bit values not yet defined?*
	JmeanPositionError	Undefined	0x0000	As the loop closes the error signal should be 0? *What is the value before the loop is closed?*

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetBSMMove 0x0001	BSMMove	0x0000	0x0001	This means the axes synchronously move to the position set by TargetPosition only when a further BSMMove(1) command is sent. On issuing the command at this time nothing happens except the mode is changed? *I think this is necessary otherwise I can't see logically how we get out of the other mode without the BSM moving!?*
	ChopStatus(Bit N)	0	1	Bit set to reflect operating mode?
	JigStatus(Bit N)	0	1	Bit set to reflect operating mode?

The BSM is now initialised at its "rest" – zero current – position and can now be used operationally. The following commands move the BSM to an example position – repeat as required for chop/jiggle modes.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetChopTargetPosition	ChopTarget Position	0x8000	0xAAAB	Set the desired position to positive half range in chop (see below)
SetJigTargetPosition	JigTarget Position	0x8000	0x8AAB	Set the desired position to positive half range in jiggle (see below)
SetBSMMove 0x0001	CMagnetoresistiveSignal	0x8000	0xA55B	See conversion curve below
	CmotorCurrent	0x8000	0x4CCD	Starts at zero goes to –20 mA
	CBEMF	0x8000	0x4CCD	Follows current exactly (*wrong but what can you do!)*
	CmeanPositionError	0x0000	0x0000	Should be zero

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
	JMagnetoresistiveSignal	0x8000	0x8957	See conversion curve below
	JMotorCurrent	0x8000	0x7333	Starts at zero goes to –5 mA
	JBEMF	0x8000	0x7333	Follows current exactly (*wrong but what can you do!)*
	JmeanPositionError	0x0000	0x0000	Should be 0

Conversion curve for commanded position
Assume that 2^{15} bits represents 1.5x full deviation in chop axis (guesswork!) – so this means 0.00549316 arcsec/bit for commands and – given P is position in ±arcsec about the rest position we have:

$$\text{TargetPosition} = (2^{15}) + P/0.00549316 \text{ adu}$$

Assume this is true for both jiggle and chop. In the example given in the tables the chop and jiggle are commanded to half range: +60 arcsec and +15 arsec respectively, or 43691 and 35499 dec.

Conversion curve for position sensor output
Kludged from Didier's document – assume ±7V is full scale movement on chop (±2 arcmin) digitise to ±12V full scale (0.366e-3 V/bit) this gives 19126 ADU for 120 arcsec deflection or 0.00627418 arcsec/bit

$$\text{MagnetoresistiveSignal} = (2^{15}-1) + P/0.00627418 \text{ adu}$$

P is position in ±arcsec from boresight. For simplicity use this on both axes.

Conversion curve for motor current and BEMF
Assume current is zero at rest position and goes ±50 mA full scale or 0.00152588 mA/bit. Assume BEMF is the same (wrong but needs proper calibration). Just for fun assume that positive deflection requires negative current. So +120 arcsec needs –40 mA giving 3 arcsec/mA in each axis. So the current (and BEMF) conversion used for the simulator is:

$$\text{Current} = 2^{15} - P/\,0.00457764 \text{ adu}$$

DCU

The DCU provides power; signal conditioning and data acquisition for the detectors on the SPIRE FPU. The unit is powered on when the 28 V power is applied to the DRCU power supply unit. A separate command is sent to the SCU to switch on EITHER the photometer or the spectrometer LIA cards – in the commands given below it is assumed we go into one or other mode following power up not from one to the other. Note also that the parameters listed here are accessed by the DCU GetHKChannel command with the appropriate ID and not as separate "Get*param*" commands.

I assume that at power on the FPGA enters a reset state, boots in some fashion, counts its fingers and starts sending housekeeping when asked. It is then ready to accept commands from the DPU. I assume this is the same as for the SCU as it is a standard interface?

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
Request to CDMS to switch on 28V line to DRCU	DAQ_IF_TEMP	Undetermined	0x4001	Complete guess Need conversion curve between ADC value and temperature
	BIAS_TEMP	Undetermined	0x4002	Guess again – bit different to make sure in right place
	LIA_B#_TEMP (12 boards in all # = 1 to 12)	Undetermined	0x4003	Guess again – bit different to make sure in right place
	PWR_STATUS (12 bits – one for each LIA card)	Undetermined	0x0000	Bits only set when LIA cards are turned on
	BDAQ_P5	Undetermined	0xB554	Used 0.366211 mV/ADU over +-12V with 2^{15}-1 = 0V
	BDAQ_P9	Undetermined	0xE00D	ditto
	BDAQ_N9	Undetermined	0x1FF0	ditto

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SCU Command DrelOnoff 0x0002	PWR_STATUS	0x0000	0x01FF	Photometer LIA cards switched on – I assume bits 0-8 are P and 9-11 are S?
	LIAP_P5	0x0000	0xB554	Used 0.366211 mV/ADU over +-12V with 2^{15}-1 = 0V
	LIAP_P9	0x0000	0xE00D	ditto
	LIAP_N9	0x0000	0x1FF0	ditto
SCU Command DrelOnoff 0x0004	PWR_STATUS	0x0000	0x0E00	Spectrometer LIA cards switched on – I assume bits 0-8 are P and 9-11 are S?
	LIAP_P5	0x0000	0xB554	Used 0.366211 mV/ADU over +-12V with 2^{15}-1 = 0V
	LIAP_P9	0x0000	0xE00D	ditto
	LIAP_N9	0x0000	0x1FF0	ditto

Initialisation

The DRCU/DPU ICD has a pretty comprehensive commanding scenario set out. As far as I can tell apart from the housekeeping reflecting the commanded values or status – no other parameters change for ANY of the DCU commands! The only thing that happens is that the detector offsets will change (to a value commensurate with the combination of the bias amplitude and the background power) and the detector noise will increase.

These commands set up the photometer JFETs; bias levels and bias frequencies and run the automatic offset routine. A similar set of commands is used for the spectrometer I won't bother to repeat them.

The command order used here is slightly different from those proposed in the ICD and starts with switching up the JFETs and then the detector conditioning voltages.

The thermal control system will be dealt with separately at some future date.

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetPhSWJfetVSS# # =1 to 6 0x007F	PhSWJfetVSS#	0x0000	0x007F	All SW detector JFET source voltages switched to half level. There is no indication that anything has happened except reflection of commanded value

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetPhMWJfetVSS# # =1 to 4 0x007F	PhMWJfetVSS#	0x0000	0x007F	All MW detector JFET source voltages switched to half level. There is no indication that anything has happened except reflection of commanded value
SetPhLWJfetVSS# # =1 to 2 0x007F	PhMWJfetVSS#	0x0000	0x007F	All LW detector JFET source voltages switched to half level. There is no indication that anything has happened except reflection of commanded value
SetPhSWJfetPwr 0x003F	PhSWJfetPwr	0x0000	0x003F	All SW JFET drain voltages are switched high. There is no indication that anything has happened except reflection of commanded value
SetPhMWJfetPwr 0x003F	PhMWJfetPwr	0x0000	0x003F	All MW and LW JFET drain voltages are switched high. There is no indication that anything has happened except reflection of commanded value
SetPhotoHeaterBias 0x007F	PhotoHeaterBias	0x0000	0x007F	The heater in the JFET modules is set to half power for a few minutes (say 1 for these purposes)….
SetPhotoHeaterBias 0x0000	PhotoHeaterBias	0x007F	0x0000	And then heater switched off

The photometer JFETs should now be switched on and at there operating voltage. At this stage there is no definitive indication that this has actually happened – i.e. no independent confirmation is available of the temperature of the JFETs or that the voltages are actually appearing at the JFET membranes. We have to wait until the detectors and LIAs are operating. Again I have changed the order of the commanding compared to the DRCU/DPU ICD. Here the bias frequency is set first – i.e. the LIAs begin to operate - and the noise checked before the amplitude; phase and offset are set. The sampling of the detector data is synchronous with the sinusoidal bias voltage frequency in order to avoid aliasing. The bias frequency is set as a division of the DCU 10 MHz master clock and will be between ~50 and 300 Hz. The sampling rate is set as a division of the bias frequency.

Command sent	**Parameter that should respond**	**Status/value before**	**Status/value after**	**Comment**
SetPhotoBiasFreq 0x0062	PhotoBiasFreq	0x0000	0x0062	Sets the bias frequency to ~200 Hz (10MHz/(512x98)
SetPhotoSampFreq 0x000C	PhotoSampFreq	0x0000	0x000C	Sets the sampling frequency to ~15.3 Hz (200/(1+12))
SetPhotoBiasMode 0x00FF	PhotoBiasMode	0x0000	0x00FF	Starts the bias generation in sine mode and sets the LIA references running *I am assuming this is what happens - CCara to confirm.*

I believe at this point the amplifiers should be working and it should be possible to sample the data – noise – coming from the JFETS. The signal level should reflect the JFET noise of ~15 nV/Hz$^{-1/2}$. If we assume 5 Hz bandwidth, gain of ~450x12 and 16 bits digitisation at the ADC with 5 V full scale (2.3951 µV/bit), there will be random noise on the signal with an amplitude of ~2.4 bits.

This is actually the only test we have available that all is well. See the last section for how the science data collection gets set up. Meanwhile let's turn on the detectors! First set the amplitude then set the phase. Once this is done – on the assumption that the phase is set correctly – we can run the automatic offset operation.

Command sent	**Parameter that should respond**	**Status/value before**	**Status/value after**	**Comment**
SetPhotoBiasAmplSW 0x004D	PhotoBiasAmplSW	0x0000	0x004D	Sets bias to ~60 mV. The parameter read back is the DAC value
SetPhotoBiasAmplMW 0x004D	PhotoBiasAmplMW	0x0000	0x004D	Sets bias to ~60 mV. The parameter read back is the DAC value
SetPhotoBiasAmplLW 0x004D	PhotoBiasAmplLW	0x0000	0x004D	Sets bias to ~60 mV. The parameter read back is the DAC value
SetPhotoDemodSW	PhotoDemodSW	0x0000	0x007F	Phase set to 180 degrees. The parameter read back is the DAC value

Command sent	Parameter that should respond	Status/value before	Status/value after	Comment
SetPhotoDemodMW	PhotoDemodMW	0x0000	0x007F	Phase set to 180 degrees. The parameter read back is the DAC value
SetPhotoDemodLW	PhotoDemodLW	0x0000	0x007F	Phase set to 180 degrees. The parameter read back is the DAC value
SetDataMode 0x0010	DataMode	0x0000 (?)	0x0010	This sets the operational mode to the automatic offset setting for the photometer. After a few seconds the offset values will respond if the electronics chain and detectors are working normally. *Note that the default – i.e. a value of 0x0000 - is "photometer normal"*
	LIAP#Offset # = 1 to 9 The 32 channel IDs per LIA card are encoded in bits 4 to 8 and the offset value in 0 to 3	0x0000	0x0nn3 where nn goes from 0 to 1F for the channel number	Offset is a four bit number related to the bias and the resistance of the bolometer – let us assume it is fix(60/200x16) = 3 for the purposes of the simulation
SetDataMode 0x0000	DataMode	0x0010	0x0000	Set the operational mode to normal bolometer again – noise at ~2-3 bits should appear on the science data.

Telemetry

These commands set the number of frames to be sent to the DPU – here set to be continuous – and the start and stop of the frame transmission. The frame type is defined by DataMode this was set previously to full photometer.

Command sent	**Parameter that should respond**	**Status/value before**	**Status/value after**	**Comment**
SetFrameCount 0x0000	FrameCount	0x0000	0x0000	It doesn't appear that this command is necessary to get continuous frame transfer?
SetStartFrame 0x0001	StartFrame	0x0000	0x0001	Starts frame transfer – something in the DPU should respond to frames starting to be sent? Send a few and then….
SetStartFrame 0x0000	StartFrame	0x0001	0x0000	Stops frame transfer – something in the DPU should respond to frames stopping being sent?