NIRSPEC

UCLA Astrophysics Program **U.C. Berkeley** **W.M.Keck Observatory**

George Brims

Revised January 6, 1999

NIRSPEC Software Programming Note 31.00

Transputer keywords (cids)

1 Introduction

The transputer messaging system uses a packet format consisting of a keyword (usually called the `cid`, an abbreviation for command identifier) and a parameter value (usually called `param`). The `cid` is a byte and the parameter a 32-bit integer. This document explains the function of each `cid`. Wherever a name is given in `Courier typeface`, this indicates it is a variable name within the occam code.

2 Contents

cid.go.spec

Numeric value: 1

Source: Host

Destination: Spectrometer clock generator and acquisition transputers.

Function:
Signals the spectrometer camera section to begin an integration, using most recently sent values for integration time etc.

Parameter values:
Irrelevant.

Returned values:
This cid isn't echoed but eventually results in cid.frame.ready.spec coming back, followed by frame data.

cid.abort.spec

Numeric value: 2

Source: Host.

Destination: Spectrometer clock generator and acquisition transputers.

Function:
Tells the spectrometer camera to abort the current integration. Will take effect as soon as the current co-add is complete. If you send it too late or the frame is a single integration, it won't have any effect, and the integration will complete as normal.

Parameter values:
Irrelevant.

Returned values:
Echoed with param = 1.

cid.frame.ready.spec

Numeric value: 3

Source: Initially, spectrometer acquisition transputer #0 (nearest clock generator). Root transputer code then generates it 16 times, preceding the 16 sets of data from the 16 acquisition transputers.

Destination: Host.

Function:
Signals that the current frame is complete and data will follow.

Parameter values:
0 initially, then 1 to 16 preceding the 16 data blocks.

Returned values:
N/A.

cid.go.scam

Numeric value: 4

Source: Host

Destination: SCAM clock generator and acquisition transputers.

Function:
Signals the slit-viewing camera section to begin an integration, using most recently sent values for integration time etc.

Parameter values:
Irrelevant.

Returned values:
This cid isn't echoed but eventually results in cid.frame.ready.scam coming back, followed by frame data.

cid.abort.scam

Numeric value: 5

Destination: SCAM clock generator and acquisition and acquisition transputers.

Function:
Tells the slit-viewing camera to abort the current integration. Will take effect as soon as the current co-add is complete. If you send it too late or it's a single integration, won't have any effect.

Parameter values:
Irrelevant.

Returned values:
Echoed with param = 0.

cid.frame.ready.scam

Numeric value: 6

Source: Slit-viewing camera acquisition transputer #0.

Destination: Host

Function:
Signals that the current frame is complete and data will follow.

Parameter values:
Irrelevant.

Returned values:
N/A.

cid.itime.spec

Numeric value: 7

Source: Host.

Destination: Spectrometer clock generator transputer.

Function:
Sets the integration time for the next frame.

Parameter values:
The integration time in milliseconds.

Returned values:
Not echoed.

cid.coadds.spec

Numeric value: 8

Source: Host

Destination: Spectrometer clock generator.

Function:
Tells the clock generator how many co-adds to do in the next frame.

Parameter values:
Number of co-adds.

Returned values:
Not echoed.

cid.sampmode.spec

Numeric value: 9

Source: Host

Destination: Spectrometer clock generator and data acquisition transputers.

Function:
Tells the spectrometer camera section what sample mode to use for the next frame. The choices are: single sampling (reset-integrate-read), double correlated sampling (reset-read-integrate-read), and multiple correlated sampling (reset-read multiple times-integrate-read multiple times). The latter is also referred to as Fowler sampling.

Parameter values:
1 = single sampling
2 = double correlated sampling
3 = multiple correlated sampling

Returned values:
Not echoed.

cid.daq.integ.start.spec

Numeric value: 10

Source: Spectrometer clock generator transputer.

Destination: Spectrometer data acquisition transputers.

Function:
Coordinates frame taking between clock generator and acquisition transputers. Each time a new co-add is to be started, this cid is sent by the clock to all the acquisition transputers. When it has passed through all of them and back to the clock generator, it starts clocking out the frame. The acquisition transputers have no knowledge of the number of co-adds to be taken. They wait for this cid or the abort or frame end cids.

Parameter values:
Irrelevant.

Returned values:
Irrelevant.

cid.daq.integ.end.spec

Numeric value: 11

Source: Spectrometer clock generator transputer.

Destination: Spectrometer data acquisition transputers.

Function:
Tells the acquisition transputers that the frame is over and there will be no more co-adds. They respond by sending their data back towards the host. The first acquisition transputer in line also generates a cid.frame.ready.spec.

Parameter values:
Irrelevant.

Returned values:
Irrelevant, although the clock generator does wait for it to come back from the acquisition transputers.

cid.itime.scam

Numeric value: 12

Source: Host

Destination: Slit-viewing camera clock generator transputer.

Function:
Sets the integration time for the next frame.

Parameter values:
The integration time in milliseconds.

Returned values:
Not echoed.

cid.coadds.scam

Numeric value: 13

Source: Host

Destination: Slit-viewing camera clock generator transputer.

Function:
Sets the number of co-adds for the next frame.

Parameter values:
The number of co-adds.

Returned values:
Not echoed.

cid.sampmode.scam

Numeric value: 14

Source: Host

Destination: Slit-viewing camera clock generator and data acquisition transputers.

Function:
Tells the slit-viewing camera section what sample mode to use for the next frame. The choices are: single sampling (reset-integrate-read), double correlated sampling (reset-read-integrate-read), and multiple correlated sampling (reset-read multiple times-integrate-read multiple times). The latter is also referred to as Fowler sampling.

Parameter values:
1 = single sampling
2 = double correlated sampling
3 = multiple correlated sampling

Returned values:
Not echoed.

cid.daq.integ.end.scam

Numeric value: 15

Source: Slit-viewing camera clock generator transputer.

Destination: Slit-viewing camera data acquisition transputers.

Function:
Tells the acquisition transputers that the frame is over and there will be no more co-adds. They respond by sending their data back towards the host. The first acquisition transputer in line also generates a cid.frame.ready.scam.

Parameter values:
Irrelevant.

Returned values:
Irrelevant, although the clock generator does wait for it to come back from the acquisition transputers.

cid.daq.integ.start.scam

Numeric value: 16

Source: Slit-viewing camera clock generator transputer.

Destination: Slit-viewing camera data acquisition transputers.

Function:
Coordinates frame taking between clock generator and acquisition transputers. Each time a new co-add is to be started, this cid is sent by the clock to all the acquisition transputers. When it has passed through all of them and back to the clock generator, it starts clocking out the frame. The acquisition transputers have no knowledge of the number of co-adds to be taken. They wait for this cid or the abort or frame end cids.

Parameter values:
Irrelevant.

Returned values:
Irrelevant.

cid.terminate.spec

Numeric value: 17

Source: Host

Destination: Spectrometer camera clock generator and acquisition transputers.

Function:
Tells the spectrometer camera section to terminate the current frame, i.e. to cut it short by finishing on the current co-add.

Parameter values:
Irrelevant.

Returned values:
Not echoed.

cid.terminate.scam

Numeric value: 18

Source: Host

Destination: Slit-viewing camera clock generator transputer.

Function:
Tells the slit-viewing camera section to terminate the current frame, i.e. to cut it short by finishing on the current co-add.

Parameter values:
Irrelevant.

Returned values:
Not echoed.

cid.multi.spec

Numeric value: 19

Source: Host

Destination: Spectrometer camera clock generator and data acquisition transputers.

Function:
Tells the spectrometer camera section how many samples to take in multiple correlated sampling mode.

Parameter values:
The number of samples.

Returned values:
Not echoed.

cid.multi.scam

Numeric value: 20

Source: Host.

Destination: Slit-viewing camera clock generator and data acquisition transputers.

Function:
Tells the slit-viewing camera section how many samples to take in multiple correlated sampling mode.

Parameter values:
The number of samples.

Returned values:
Not echoed.

cid.undeselect.spec

Numeric value: 21

Source: Host.

Destination: Spectrometer camera clock generator transputer.

Function:
The Aladdin 2 array has the ability to “deselect”, or turn off, specific rows (actually row pairs), so that they aren’t powered up as we clock through them. This can be used for instance to prevent a hot pixel from giving a lot of its neighboring pixels elevated background. As part of this functionality, the whole array powers up with *all* rows deselected (i.e. they don’t work). This command triggers the clock generator to send out the pattern to turn all the rows on.

Parameter values:
Irrelevant.

Returned values:
Not echoed.

cid.samprate.spec

Numeric value: 22

Source: Host

Destination: Spectrometer camera clock generator.

Function:
Tells the clock generator what A-D sample rate to use. The sampling rate is set by increasing the number of clock ticks in the output waveform. The maximum rate is 500 kHz, the limit of the A-D we use, and the lower limit is set by the capacity of the output FIFO buffer to hold a longer waveform.

Parameter values:
The sample rate in kHz.

Returned values:
Echoed unchanged.

cid.samprate.scam

Numeric value: 23

Source: Host.

Destination: Slit-viewing camera clock generator transputer.

Function:
Tells the clock generator what A-D sample rate to use. The sampling rate is set by increasing the number of clock ticks in the output waveform. The maximum rate is 500 kHz, the limit of the A-D we use, and the lower limit is set by the capacity of the output FIFO buffer to hold a longer waveform.

Parameter values:
Sample rate in kHz.

Returned values:
Echoed unchanged.

cid.quad1.offset.spec

Numeric value: 24

Source: Host.

Destination: Spectrometer camera clock generator transputer.

Function:
The Aladdin detector arrays have four distinct quadrants. The common DC pre-amp offset for all the analog channels in each quadrant is controllable via a 4-channel D-A chip. This cid is used to send the value for the D-A for quadrant 1.

Parameter values:
0 to 4095, covering the full range of the offset values.

Returned values:
Echoed unchanged.

cid.quad2.offset.spec

Numeric value: 25

Source: Host.

Destination: Spectrometer camera clock generator transputer.

Function:
The Aladdin detector arrays have four distinct quadrants. The common DC pre-amp offset for all the analog channels in each quadrant is controllable via a 4-channel D-A chip. This cid is used to send the value for the D-A for quadrant 2.

Parameter values:
0 to 4095, covering the full range of the offset values.

Returned values:
Echoed unchanged.

cid.quad3.offset.spec

Numeric value: 26

Source: Host.

Destination: Spectrometer camera clock generator transputer.

Function:
The Aladdin detector arrays have four distinct quadrants. The common DC pre-amp offset for all the analog channels in each quadrant is controllable via a 4-channel D-A chip. This cid is used to send the value for the D-A for quadrant 3.

Parameter values:
0 to 4095, covering the full range of the offset values.

Returned values:
Echoed unchanged.

cid.quad4.offset.spec

Numeric value: 27

Source: Host.

Destination: Spectrometer camera clock generator transputer.

Function:
The Aladdin detector arrays have four distinct quadrants. The common DC pre-amp offset for all the analog channels in each quadrant is controllable via a 4-channel D-A chip. This cid is used to send the value for the D-A for quadrant 4.

Parameter values:
0 to 4095, covering the full range of the offset values.

Returned values:
Echoed unchanged.

cid.quad1.offset.scam

Numeric value: 28

Source: Host.

Destination: Slit-viewing camera clock generator transputer.

Function:
The PICNIC detector arrays have four distinct quadrants. The DC pre-amp offset for each quadrant is controllable via a 4-channel D-A chip. This cid is used to send the value for the D-A for quadrant 1.

Parameter values:
0 to 4095, covering the full range of the offset values.

Returned values:
Echoed unchanged.

cid.quad2.offset.scam

Numeric value: 29

Source: Host.

Destination: Slit-viewing camera clock generator transputer.

Function:
The PICNIC detector arrays have four distinct quadrants. The DC pre-amp offset for each quadrant is controllable via a 4-channel D-A chip. This cid is used to send the value for the D-A for quadrant 2.

Parameter values:
0 to 4095, covering the full range of the offset values.

Returned values:
Echoed unchanged.

cid.quad3.offset.scam

Numeric value: 30

Source: Host.

Destination: Slit-viewing camera clock generator transputer.

Function:
The PICNIC detector arrays have four distinct quadrants. The DC pre-amp offset for each quadrant is controllable via a 4-channel D-A chip. This cid is used to send the value for the D-A for quadrant 3.

Parameter values:
0 to 4095, covering the full range of the offset values.

Returned values:
Echoed unchanged.

cid.quad4.offset.scam

Numeric value: 31

Source: Host.

Destination: Slit-viewing camera clock generator transputer.

Function:
The PICNIC detector arrays have four distinct quadrants. The DC pre-amp offset for each quadrant is controllable via a 4-channel D-A chip. This cid is used to send the value for the D-A for quadrant 4.

Parameter values:
0 to 4095, covering the full range of the offset values.

Returned values:
Echoed unchanged.

cid.set.gain.spec

Numeric value: 32

Source: Host

Destination: Spectrometer camera clock generator transputer.

Function:
Sets the gain for the spectrometer camera pre-amps by setting 2 bits of digital output port 0. These two bits are passed through the interface board to the pre-amp boards.

Parameter values:
1, 2, 3 or 4, to select the four possible gain values (which are *NOT* 1, 2, 3, and 4!)

Returned values:
Echoed unchanged.

cid.set.freq.spec

Numeric value: 33

Source: Host

Destination: Spectrometer camera clock generator transputer.

Function:
Sets the filter bandwidth for the spectrometer camera pre-amps by setting 2 bits of digital output port 0. These two bits are passed through the interface board to the pre-amp boards.

Parameter values:
1, 2, 3 or 4, to select the four possible filter cutoff values.

Returned values:
Echoed unchanged.

cid.set.gain.scam

Numeric value: 34

Source: Host

Destination: Slit-viewing camera clock generator transputer.

Function:
Sets the gain for the slit-viewing camera pre-amps by setting 2 bits of digital output port 0. These two bits are passed through the interface board to the pre-amp boards.

Parameter values:
1, 2, 3 or 4, to select the four possible gain values (which are *NOT* 1, 2, 3, and 4!)

Returned values:
Echoed unchanged.

cid.set.freq.scam

Numeric value: 35

Source: Host

Destination: Slit-viewing camera clock generator transputer.

Function:
Sets the filter bandwidth for the slit-viewing camera pre-amps by setting 2 bits of digital output port 0. These two bits are passed through the interface board to the pre-amp boards.

Parameter values:
1, 2, 3 or 4, to select the four possible filter cutoff values.

Returned values:
Echoed unchanged.

cid.detbias.spec

Numeric value: 36

Source: Host

Destination: Spectrometer camera clock generator transputer.

Function:
Uses a serial D-A converter to set the detector offset for the spectrometer camera.

Parameter values:
0 to 4095, covering the full voltage range available.

Returned values:
Echoed unchanged.

cid.fifo.test.spec

Numeric value: 37

Source: Host

Destination: Spectrometer camera clock generator transputer.

Function:
Sends out test patterns on the clock generator output port. Demonstrates the functionality of output via both direct and FIFO buffer registers. The parameter is the repeat count (called `repeatcount!`) for the pattern. First all 32 clock bits turn on and off in sync `repeatcount` times. Then they pulse once each in succession `repeatcount` times, then finally pulse in sync again `repeatcount` times. The first and last sections are written out via the direct output register, and the middle section via the FIFO repeat feature. This exercises all the hardware involved in sending out a normal clocking waveform. For details of this process see the clock generator code and the explanation in NSPN18.

Parameter values:
Repeat count as explained above.

Returned values:

Echoed unchanged if everything works OK.

cid.fifo.test.scam

Numeric value: 38

Source: Host

Destination: Slit-viewing camera clock generator transputer.

Function:
Sends out test patterns on the clock generator output port. Demonstrates the functionality of output via both direct and FIFO buffer registers. The parameter is the repeat count (called `repeatcount!`) for the pattern. First all 32 clock bits turn on and off in sync `repeatcount` times. Then they pulse once each in succession `repeatcount` times, then finally pulse in sync again `repeatcount` times. The first and last sections are written out via the direct output register, and the middle section via the FIFO repeat feature. This exercises all the hardware involved in sending out a normal clocking waveform. For details of this process see the clock generator code and the explanation in NSPN18.

Parameter values:
Repeat count as explained above.

Returned values:

Echoed unchanged if everything works OK.

cid.test.frames.spec

Numeric value: 39

Source: Host

Destination: Spectrometer clock generator transputer.

Function:
Sends out a read frame clock pattern the number of times specified by the parameter. Used to check bit assignment, signal continuity etc. No data will be returned because the clock generator sends no cids to the acquisition transputers to tell them to read.

Parameter values:
Repeat count for the clock waveform.

Returned values:
Echoed unchanged once the repeats are complete.

cid.test.frames.scam

Numeric value: 40

Source: Host

Destination: Slit-viewing camera clock generator transputer.

Function:
Sends out a read frame clock pattern the number of times specified by the parameter. Used to check bit assignment, signal continuity etc. No data will be returned because the clock generator sends no cids to the acquisition transputers to tell them to read.

Parameter values:
Repeat count for the clock waveform.

Returned values:
Echoed unchanged once the repeats are complete.

cid.sensors.reset

Numeric value: 41

Source: Host

Destination: Root transputer.

Function:
The root transputer reads Dallas Semiconductor DS1820 temperature sensor ICs distributed through the electronics cabinets to monitor the air temperature. This cid simply sends a reset signal to the sensors (they are on a 3-wire serial daisychain). If they respond properly then we know they are connected properly.

Parameter values:
Irrelevant.

Returned values:
1 for success, 0 for failure.

cid.sensors.getid

Numeric value: 42

Source: Host.

Destination: Root transputer.

Function:
The root transputer reads Dallas Semiconductor DS1820 temperature sensor ICs distributed through the electronics cabinets to monitor the air temperature. Each DS1820 has a unique 64-bit id code programmed into its ROM at the factory, and will only send back data if the request includes this id. This cid requests a sensor to send its id code. The cid has to be sent twice to get the upper and lower 32 bits of the 64. This sensor must be the only one connected otherwise all the sensors will respond and the result will be gibberish.

Parameter values:
0 for the lower 32 bits of the code, 1 for the upper 32 bits.

Returned values:
Upper or lower 32 bits of the 64-bit id code.

cid.sensor.read

Numeric value: 43

Source: Host.

Destination: Root transputer.

Function:
The root transputer reads Dallas Semiconductor DS1820 temperature sensor ICs distributed through the electronics cabinets to monitor the air temperature. The root code reads the sensors in sequence at intervals of several seconds. This reading can be turned on or off and the interval set via the parameter.

Parameter values:
0 = don't read
>0 = read every `param` seconds.

Returned values:
Sensor number and temperature data packaged together. Number is sensor number times 1000, plus the returned data which is in half degrees Celsius. So sensor 9, temperature 25.5 would give 9051.

cid.serial1A.baud

Numeric value: 50

Source: Host

Destination: Serial transputer 1.

Function:
Sets the baud rate for serial 1, port A. Used for testing only, baud rate is set to a default value at the start of the program.

Parameter values:
Baud rate/100, e.g. 300 baud is 3, 19.2 kbaud is 192.

Returned values:
Not echoed.

cid.serial1B.baud

Numeric value: 51

Source: Host

Destination: Serial transputer 1.

Function:
Sets the baud rate for serial 1, port B. Used for testing only, baud rate is set to a default value at the start of the program.

Parameter values:
Baud rate/100, e.g. 300 baud is 3, 19.2 kbaud is 192.

Returned values:
Not echoed.

cid.serial2A.baud

Numeric value: 52

Source: Host

Destination: Serial transputer 2.

Function:
Sets the baud rate for serial 2, port A. Used for testing only, baud rate is set to a default value at the start of the program.

Parameter values:
Baud rate/100, e.g. 300 baud is 3, 19.2 kbaud is 192.

Returned values:
Not echoed.

cid.serial2B.baud

Numeric value: 53

Source: Host

Destination: Serial transputer 2.

Function:
Sets the baud rate for serial 2, port B. Used for testing only, baud rate is set to a default value at the start of the program.

Parameter values:
Baud rate/100, e.g. 300 baud is 3, 19.2 kbaud is 192.

Returned values:
Not echoed.

cid.serial1A.handshake

Numeric value: 54

Source: Host

Destination: Serial transputer 1.

Function:
Turns on or off hardware (CTS/RTS) handshaking for serial 1, port A. Used for functional testing only, since we use a 3-wire connection from the serial ports to most devices.

Parameter values:
0 = handshaking off (data leads only).
1 = handshaking on.

Returned values:
Not echoed.

cid.serial1B.handshake

Numeric value: 55

Source: Host

Destination: Serial transputer 1.

Function:
Turns on or off hardware (CTS/RTS) handshaking for serial 1, port B. Used for functional testing only, since we use a 3-wire connection from the serial ports to most devices.

Parameter values:
0 = handshaking off (data leads only).
1 = handshaking on.

Returned values:
Not echoed.

cid.serial2A.handshake

Numeric value: 56

Source: Host

Destination: Serial transputer 2.

Function:
Turns on or off hardware (CTS/RTS) handshaking for serial 2, port A. Used for functional testing only, since we use a 3-wire connection from the serial ports to most devices.

Parameter values:
0 = handshaking off (data leads only).
1 = handshaking on.

Returned values:
Not echoed.

cid.serial2B.handshake

Numeric value: 57

Source: Host

Destination: Serial transputer 2.

Function:
Turns on or off hardware (CTS/RTS) handshaking for serial 2, port B. Used for functional testing only, since we use a 3-wire connection from the serial ports to most devices.

Parameter values:
0 = handshaking off (data leads only).
1 = handshaking on.

Returned values:
Not echoed.

cid.serial.1A.putchar

Numeric value: 58

Source: Host

Destination: Serial transputer 1.

Function:
Sends out a single character over serial 1, channel A. Used for testing.

Parameter values:
The ASCII value of the character, as a 32-bit integer.

Returned values:
Not echoed.

cid.serial.1B.putchar

Numeric value: 59

Source: Host

Destination: Serial transputer 1.

Function:
Sends out a single character over serial 1, channel B. Used for testing.

Parameter values:
The ASCII value of the character, as a 32-bit integer.

Returned values:
Not echoed.

cid.serial.2A.putchar

Numeric value: 60

Source: Host

Destination: Serial transputer 1.

Function:
Sends out a single character over serial 2, channel A. Used for testing.

Parameter values:
The ASCII value of the character, as a 32-bit integer.

Returned values:
Not echoed.

cid.serial.2B.putchar

Numeric value: 61

Source: Host

Destination: Serial transputer 1.

Function:
Sends out a single character over serial 2, channel B. Used for testing.

Parameter values:
The ASCII value of the character, as a 32-bit integer.

Returned values:
Not echoed.

cid.serial.1A.getchar

Numeric value: 62

Source: Host.

Destination: Serial transputer 1.

Function:
Reads back a single character from serial 1, channel A. Used for testing. Has one odd behavior; the RS232 driver code multiplexes I/O for both channels through a single virtual link to the rest of the code, so this request might prompt it to send a character for the other channel. However the byte is preceded by a tag denoting which port it came from, so the serial 1 code sends it back with the appropriate cid value so we know which port it was from.

Parameter values:
Irrelevant.

Returned values:
ASCII value of character read, as a 32-bit integer.

cid.serial.1B.getchar

Numeric value: 63

Source: Host.

Destination: Serial transputer 1.

Function:
Reads back a single character from serial 1, channel B. Used for testing. Has one odd behavior; the RS232 driver code multiplexes I/O for both channels through a single virtual link to the rest of the code, so this request might prompt it to send a character for the other channel. However the byte is preceded by a tag denoting which port it came from, so the serial 1 code sends it back with the appropriate cid value so we know which port it was from.

Parameter values:
Irrelevant.

Returned values:
ASCII value of character read, as a 32-bit integer.

cid.serial.2A.getchar

Numeric value: 64

Source: Host.

Destination: Serial transputer 2.

Function:
Reads back a single character from serial 2, channel A. Used for testing. Has one odd behavior; the RS232 driver code multiplexes I/O for both channels through a single virtual link to the rest of the code, so this request might prompt it to send a character for the other channel. However the byte is preceded by a tag denoting which port it came from, so the serial 2 code sends it back with the appropriate cid value so we know which port it was from.

Parameter values:
Irrelevant.

Returned values:
ASCII value of character read, as a 32-bit integer.

cid.serial.2B.getchar

Numeric value: 65

Source: Host.

Destination: Serial transputer 2.

Function:
Reads back a single character from serial 2, channel B. Used for testing. Has one odd behavior; the RS232 driver code multiplexes I/O for both channels through a single virtual link to the rest of the code, so this request might prompt it to send a character for the other channel. However the byte is preceded by a tag denoting which port it came from, so the serial 2 code sends it back with the appropriate cid value so we know which port it was from.

Parameter values:
Irrelevant.

Returned values:
ASCII value of character read, as a 32-bit integer.

cid.serial1A.test.string

Numeric value: 66

Source: Host.

Destination: Serial transputer 1.

Function:
Sends out a “Hello world” string identifying the output port as serial 1, port A.

Parameter values:
Irrelevant.

Returned values:
Not echoed.

cid.serial1B.test.string

Numeric value: 67

Source: Host.

Destination: Serial transputer 1.

Function:
Sends out a “Hello world” string identifying the output port as serial 1, port B.

Parameter values:
Irrelevant.

Returned values:
Not echoed.

cid.serial2A.test.string

Numeric value: 68

Source: Host.

Destination: Serial transputer 2.

Function:
Sends out a “Hello world” string identifying the output port as serial 2, port A.

Parameter values:
Irrelevant.

Returned values:
Not echoed.

cid.serial2B.test.string

Numeric value: 69

Source: Host.

Destination: Serial transputer 2.

Function:
Sends out a “Hello world” string identifying the output port as serial 2, port B.

Parameter values:
Irrelevant.

Returned values:
Not echoed.

cid.get.cryo.temp

Numeric value: 70

Source: Host.

Destination: Serial transputer

Function:
Serial transputer 2 reads a series of cryogenic temperatures from LakeShore model 208 readout devices. The code cycles through the different sensors and returns the temperatures. This cid turns the cycle on or off. There are 8 channels on each sensor. These are defined as channels 1-8 and 11-18 (channels 9 & 10 are read from another device).

Parameter values:
0 = don’t read temperatures.
1 = read and send back temperatures.

Returned values:
Since the parameter values have to be integer, we convert the temperatures to milliKelvin and package them with the channel number times 1 million. So channel 6, 78.5K would come back as 6078500. If there is a problem reading a channel it comes back as 0 Kelvin.

cid.get.detector.temp

Numeric value: 71

Source: Host.

Destination: Serial transputer 1.

Function:
Serial transputer 1 reads two cryogenic temperatures from the LakeShore 330 controller. One of these temperatures is the Aladdin detector. The temperature reading is done in a cycle, and this cid turns the reading on or off. The two channels are designated 9 and 10 (9 is the detector).

Parameter values:

`param` = 0	don't read temperatures.
`param` > 0	read and return temperatures every `param` seconds (the practical minimum is about 8 seconds)

Returned values:

Since the parameter values have to be integer, we convert the temperatures to milliKelvin and package them with the channel number times 1 million. So channel 9, 30.1K would come back as 9030100. If there is a problem reading a channel it comes back as 0 Kelvin.

cid.change.detector.temp

Numeric value: 72

Source: Host.

Destination: Serial transputer 1.

Function:
The temperature of the Aladdin detector (spectrometer camera) is controlled by a LakeShore model 33 controller. This cid changes the control setpoint.

Parameter values:
Degrees K (integer).

Returned values:
Not echoed.

cid.power.on

Numeric value: 73

Source: Host.

Destination: Serial transputer 1.

Function:
Power in the electronics cabinets is switched by a Pulizzi computer-controllable power strip, with 8 outlets. This cid tells the power strip to turn on one of the outlets, designated 1 to 8.

Parameter values:
1 to 8, the number of the outlet to be switched on.

Returned values:
0 = success
-1 = Error (wrong reply from power strip).
-2 = Error (invalid parameter).
-3 = Error (couldn’t establish communications with power strip).

cid.power.off

Numeric value: 74

Source: Host.

Destination: Serial transputer 1.

Function:
Power in the electronics cabinets is switched by a Pulizzi computer-controllable power strip, with 8 outlets. This cid tells the power strip to turn off one of the outlets, designated 1 to 8.

Parameter values:
1 to 8, the number of the outlet to be switched off.

Returned values:
0 = success
-1 = Error (wrong reply from power strip).
-2 = Error (invalid parameter).
-3 = Error (couldn’t establish communications with power strip).

cid.all.power.off

Numeric value: 75

Source: Host.

Destination: Serial transputer 1.

Function:
Power in the electronics cabinets is switched by a Pulizzi computer-controllable power strip, with 8 outlets. This cid tells the power strip to turn off all the outlets.

Parameter values:
Irrelevant.

Returned values:
0 = success (I'm not sure we ever see this one since turning off the power kills everything!)
-1 = Error (wrong reply from power strip).
-3 = Error (couldn't establish communications with power strip).

cid.cycle.all.power

Numeric value: 76

Source: Host.

Destination: Serial transputer 1.

Function:
Power in the electronics cabinets is switched by a Pulizzi computer-controllable power strip, with 8 outlets. This cid tells the power strip to cycle power off and back on, using its watchdog timer function. The code sets the power strip's timeout to the minimum value (supposed to be 12 seconds but it takes longer than that), then says nothing more so that the watchdog timer triggers.

Parameter values:
Irrelevant.

Returned values:
0 = success
-1 = Error (wrong reply from power strip).
-2 = Error (invalid parameter).
-3 = Error (couldn't establish communications with power strip).

cid.lamps.command

Numeric value: 77

Source: Host.

Destination: Root transputer.

Function:
Turns on the required lamps in the calibration unit. There are 4 arc lamps (neon, xenon, argon and krypton) and 3 quartz-halogen bulbs. The code checks which bulbs are on by reading the status information from the monitor circuit.

Parameter values:
The 7 least significant bits each turn on or off one of the lamps.

Returned values:
The cid returned is `cid.lamps.status`, with the bit pattern returned by the monitor hardware.

cid.lamps.status

Numeric value: 78

Source: Host.

Destination: Root transputer.

Function:
Checks the input status bits reading the lamp monitor circuit, and returns them to the host.

Parameter values:
Irrelevant.

Returned values:
Bit pattern representing which lamps which are on and which are off (a 1 means on).

cid.rotts1

Numeric value: 100

Source:

Destination:

Function:

Parameter values:

Returned values:

cid.rotts2

Numeric value: 101

Source:

Destination:

Function:

Parameter values:

Returned values:

cid.rotpdst

Numeric value: 102

Source:

Destination:

Function:

Parameter values:

Returned values:

cid.rotnew

Numeric value: 103

Source:

Destination:

Function:

Parameter values:

Returned values:

cid.mot.irot.step

Numeric value: 150

Source: Host.

Destination: Motor transputer 3.

Function:
Commands the transputer to move the image rotator mechanism a specific number of steps (i.e. a relative move).

Parameter values:
The number of steps. If the required number of steps is negative (i.e. in the counterclockwise direction) the parameter is the number of steps plus 999999. This is because of a limitation in the host-transputer protocol that prevents negative integers going from the host to the transputers.

Returned values:
If the move was successful and no limit switches were encountered, the value is `move.success`. If a limit switch was encountered, the value is `move.limit.switch`. If the move was aborted by a command from the host, the value is `move.abort`.

cid.mot.irot.pos

Numeric value: 151

Source: Host.

Destination: Motor transputer 3.

Function:
Tells the code to move the image rotator mechanism to a specific position (i.e. an absolute move).

Parameter values:
The required position in increments of 0.01°. Minimum position is 9000 (or 90°) and maximum is 27000 (or 270°).

Returned values:

`move.success`	the move was completed successfully
`move.pos.unknown`	the mechanism initial position wasn't known, so we can't move
`motor.invalid.parameter`	the requested position was out of range
`move.abort`	the move was aborted on command from the host
`move.limit.switch`	the move ended prematurely when a limit switch was tripped

cid.mot.irot.init

Numeric value: 152

Source: Host.

Destination: Motor 3 transputer.

Function:
Tells the transputer to initialize the image rotator mechanism. First it seeks until it hits a limit switch (searching for the secondary one if it doesn't find the primary), then it moves the mechanism to the straight up and down (180°) position.

Parameter values:

Returned values:

cid.mot.irot.initloc

Numeric value: 153

Source: Host.

Destination: Motor transputer 3.

Function:
Sets the image rotator position (in steps) to a new value. An engineering-level command, used in the case where the limit switches aren't working but we actually know where the mechanism is.

Parameter values:
Position in steps. Must be between zero and `irot.total.steps`, the upper limit of motion.

Returned values:
Echoed unchanged unless the step value is outside the range, in which case it sends back `motor.invalid.parameter`.

cid.mot.irot.location

Numeric value: 154

Source: Host.

Destination: Motor transputer 3.

Function:
Requests the code to send back the location of the image rotator mechanism.

Parameter values:
Irrelevant.

Returned values:
The position in steps (not degrees). A value of -1 indicates the mechanism position is unknown (i.e. it hasn't been initialized).

cid.mot.irot.switch

Numeric value: 155

Source: Host.

Destination: Motor transputer 3.

Function:
Prompts the code to check the status of the limit and home microswitches in the image rotator mechanism.

Parameter values:
Irrelevant.

Returned values:
An integer containing 4 bits set to reflect the state of the switches. A 1 means the switch is open and a 0 means it's closed.

cid.mot.irot.abort

Numeric value: 156

Source: Host.

Destination: Motor transputer 3.

Function:
Tells the mechanism to abort the current move.

Parameter values:
Irrelevant.

Returned values:
Echoed unchanged. Also, if a move or initialization was in progress, that cid will also come back with `move.abort` as parameter.

cid.mot.irot.track

Numeric value: 157

Source: Host.

Destination: Motor transputer 3.

Function:
Passes a track rate to the image rotator code. If the rate is non-zero, the code enters the track routine and moves at that speed until another update arrives. If the rate is zero, it exits track mode.

Parameter values:
Track rate in steps per second.

Returned values:

cid.mot.filt1.step

Numeric value: 160

Source: Host.

Destination: Motor transputer 1.

Function:
Commands the transputer to move filter wheel 1 a specific number of steps (i.e. a relative move).

Parameter values:
The number of steps. If the required number of steps is negative (i.e. in the counterclockwise direction) the parameter is the number of steps plus 999999. This is because of a limitation in the host-transputer protocol that prevents negative integers going from the host to the transputers.

Returned values:
If the move was successful, value is `move.success`. If the move was aborted by a command from the host, the value is `move.abort`.

cid.mot.filt1.pos

Numeric value: 161

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the code to move filter wheel 1 to a specific position (i.e. an absolute move). The mechanism checks position switches as it moves to ensure it goes through all the intervening positions.

Parameter values:
The filter number. Minimum value is 3 and maximum 11.

Returned values:

`move.success`	the move was completed successfully
`move.pos.unknown`	the mechanism initial position wasn't known, so we can't move
`move.pos.switch`	the wrong number of position switches were activated over the range of the move
`motor.invalid.parameter`	the requested position was out of range

`move.abort`	the move was aborted on command from the host

cid.mot.filt1.init

Numeric value: 162

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the code to initialize filter wheel 1. First it seeks the home switch or its backup, and if one of those is successfully found, moves the mechanism to the `init.home` position (in this case a blank position).

Parameter values:
Irrelevant.

Returned values:

cid.mot.filt1.initloc

Numeric value: 163

Source: Host.

Destination: Motor transputer 1.

Function:
Sets the filter wheel 1 position (in steps) to a new value. An engineering-level command, used in the case where the home switches aren't working but we actually know where the mechanism is.

Parameter values:
Position in steps. Must be between zero and `filt1.total.steps`, the number if steps in a full rotation of the wheel.

Returned values:
Echoed unchanged unless the step value is outside the range, in which case it sends back `motor.invalid.parameter`.

cid.mot.filt1.location

Numeric value: 164

Source: Host.

Destination: Motor transputer 1.

Function:
Requests the code to send back the location of filter wheel 1.

Parameter values:
Irrelevant.

Returned values:
The position in steps (not filter positions). A value of -1 indicates the mechanism position is unknown (i.e. it hasn't been initialized).

cid.mot.filt1.switch

Numeric value: 165

Source: Host.

Destination: Motor transputer 1.

Function:
Prompts the code to check the status of the position and home microswitches in filter wheel 1.

Parameter values:
Irrelevant.

Returned values:
An integer containing 4 bits set to reflect the state of the switches. A 1 means the switch is open and a 0 means it's closed.

cid.mot.filt1.abort

Numeric value: 166

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the mechanism to abort the current move.

Parameter values:
Irrelevant.

Returned values:
Echoed unchanged. Also, if a move or initialization was in progress, that cid will also come back with `move.abort` as parameter.

cid.mot.filt2.step

Numeric value: 170

Source: Host.

Destination: Motor transputer 1.

Function:
Commands the transputer to move filter wheel 2 a specific number of steps (i.e. a relative move).

Parameter values:
The number of steps. If the required number of steps is negative (i.e. in the counterclockwise direction) the parameter is the number of steps plus 999999. This is because of a limitation in the host-transputer protocol that prevents negative integers going from the host to the transputers.

Returned values:
If the move was successful, the value is `move.success`. If the move was aborted by a command from the host, the value is `move.abort`.

cid.mot.filt2.pos

Numeric value: 171

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the code to move filter wheel 2 to a specific position (i.e. an absolute move). The mechanism checks position switches as it moves to ensure it goes through all the intervening positions.

Parameter values:
The filter number. Minimum value is 0 and maximum 10.

Returned values:

`move.success`	the move was completed successfully
`move.pos.unknown`	the mechanism initial position wasn't known, so we can't move
`move.pos.switch`	the wrong number of position switches were activated over the range of the move
`motor.invalid.parameter`	the requested position was out of range

`move.abort`	the move was aborted on command from the host

cid.mot.filt2.init

Numeric value: 172

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the code to initialize filter wheel 2. First it seeks the home switch or its backup, and if one of those is successfully found, moves the mechanism to the `init.home` position (in this case a blank position).

Parameter values:
Irrelevant.

Returned values:

cid.mot.filt2.initloc

Numeric value: 173

Source: Host.

Destination: Motor transputer 1.

Function:
Sets the filter wheel 2 position (in steps) to a new value. An engineering-level command, used in the case where the home switches aren't working but we actually know where the mechanism is.

Parameter values:
Position in steps. Must be between zero and `filt2.total.steps`, the number if steps in a full rotation of the wheel.

Returned values:
Echoed unchanged unless the step value is outside the range, in which case it sends back `motor.invalid.parameter`.

cid.mot.filt2.location

Numeric value: 174

Source: Host.

Destination: Motor transputer 1.

Function:
Requests the code to send back the location of filter wheel 2.

Parameter values:
Irrelevant.

Returned values:
The position in steps (not filter positions). A value of -1 indicates the mechanism position is unknown (i.e. it hasn't been initialized).

cid.mot.filt2.switch

Numeric value: 175

Source: Host.

Destination: Motor transputer 1.

Function:
Prompts the code to check the status of the position and home microswitches in filter wheel 2.

Parameter values:
Irrelevant.

Returned values:
An integer containing 4 bits set to reflect the state of the switches. A 1 means the switch is open and a 0 means it's closed.

cid.mot.filt2.abort

Numeric value: 176

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the mechanism to abort the current move.

Parameter values:
Irrelevant.

Returned values:
Echoed unchanged. Also, if a move or initialization was in progress, that cid will also come back with `move.abort` as parameter.

cid.mot.slit.step

Numeric value: 180

Source: Host.

Destination: Motor transputer 1.

Function:
Commands the transputer to move the slit wheel a specific number of steps (i.e. a relative move).

Parameter values:
The number of steps. If the required number of steps is negative (i.e. in the counterclockwise direction) the parameter is the number of steps plus 999999. This is because of a limitation in the host-transputer protocol that prevents negative integers going from the host to the transputers.

Returned values:
If the move was successful, the value is `move.success`. If the move was aborted by a command from the host, the value is `move.abort`.

cid.mot.slit.pos

Numeric value: 181

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the code to move the slit wheel to a specific position (i.e. an absolute move). The mechanism checks position switches as it moves to ensure it goes through all the intervening positions.

Parameter values:
The slit number. Minimum value is 0 and maximum 11.

Returned values:

`move.success`	the move was completed successfully
`move.pos.unknown`	the mechanism initial position wasn't known, so we can't move
`move.pos.switch`	the wrong number of position switches were activated over the range of the move
`motor.invalid.parameter`	the requested position was out of range

`move.abort`	the move was aborted on command from the host

cid.mot.slit.init

Numeric value: 182

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the code to initialize the slit wheel. First it seeks the home switch or its backup, and if one of those is successfully found, moves the mechanism to the `init.home` position (in this case position 0).

Parameter values:
Irrelevant.

Returned values:

cid.mot.slit.initloc

Numeric value: 183

Source: Host.

Destination: Motor transputer 1.

Function:
Sets the slit wheel position (in steps) to a new value. An engineering-level command, used in the case where the home switches aren't working but we actually know where the mechanism is.

Parameter values:
Position in steps. Must be between zero and `slit.total.steps`, the number if steps in a full rotation of the wheel.

Returned values:
Echoed unchanged unless the step value is outside the range, in which case it sends back `motor.invalid.parameter`.

cid.mot.slit.location

Numeric value: 184

Source: Host.

Destination: Motor transputer 1.

Function:
Requests the code to send back the location of the slit wheel.

Parameter values:
Irrelevant.

Returned values:
The position in steps (not slit positions). A value of -1 indicates the mechanism position is unknown (i.e. it hasn't been initialized).

cid.mot.slit.switch

Numeric value: 185

Source: Host.

Destination: Motor transputer 1.

Function:
Prompts the code to check the status of the position and home microswitches in the slit wheel.

Parameter values:
Irrelevant.

Returned values:
An integer containing 4 bits set to reflect the state of the switches. A 1 means the switch is open and a 0 means it's closed.

cid.mot.slit.abort

Numeric value: 186

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the mechanism to abort the current move.

Parameter values:
Irrelevant.

Returned values:
Echoed unchanged. Also, if a move or initialization was in progress, that cid will also come back with `move.abort` as parameter.

cid.mot.echl.step

Numeric value: 190

Source: Host.

Destination: Motor transputer 2.

Function:
Commands the transputer to move the echelle mechanism a specific number of steps (i.e. a relative move).

Parameter values:
The number of steps. If the required number of steps is negative (i.e. in the counterclockwise direction) the parameter is the number of steps plus 999999. This is because of a limitation in the host-transputer protocol that prevents negative integers going from the host to the transputers.

Returned values:
If the move was successful and no limit switches were encountered, the value is `move.success`. If a limit switch was encountered, the value is `move.limit.switch`. If the move was aborted by a command from the host, the value is `move.abort`.

cid.mot.echl.pos

Numeric value: 191

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the code to move the echelle mechanism to a specific position (i.e. an absolute move). There is one position switch at the low-res flat position. The code checks it either goes through or arrives at this switch position.

Parameter values:
The position in increments of 0.01 °. Minimum position is 5000 (50 °) and maximum is 18200 (182°).

Returned values:

`move.success`	the move was completed successfully
`move.pos.unknown`	the mechanism initial position wasn't known, so we can't move
`move.pos.switch`	the wrong number of position switches were activated
`motor.invalid.parameter`	the requested position was out of range
`move.limit.switch`	the move ended prematurely when a limit switch was tripped
`move.abort`	the move was aborted on command from the host

cid.mot.echl.init

Numeric value: 192

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the code to initialize the echelle mechanism. First it seeks the home switch or its backup, and if one of those is successfully found, moves the mechanism to the `init.home` position (63.5°).

Parameter values:
Irrelevant.

Returned values:

cid.mot.echl.initloc

Numeric value: 193

Source: Host.

Destination: Motor transputer 2.

Function:
Sets the echelle mechanism position (in steps) to a new value. An engineering-level command, used in the case where the home switches aren't working but we actually know where the mechanism is.

Parameter values:
Position in steps. Must be between zero and `echl.total.steps`, the number if steps in the full range of motion of the mechanism.

Returned values:
Echoed unchanged unless the step value is outside the range, in which case it sends back `motor.invalid.parameter`.

cid.mot.echl.location

Numeric value: 194

Source: Host.

Destination: Motor transputer 2.

Function:
Requests the code to send back the location of the echelle mechanism.

Parameter values:
Irrelevant.

Returned values:
The position in steps (not degrees). A value of -1 indicates the mechanism position is unknown (i.e. it hasn't been initialized).

cid.mot.echl.switch

Numeric value: 195

Source: Host.

Destination: Motor transputer 2.

Function:
Prompts the code to check the status of the limit and home microswitches in the echelle mechanism.

Parameter values:
Irrelevant.

Returned values:
An integer containing 4 bits set to reflect the state of the switches. A 1 means the switch is open and a 0 means it's closed.

cid.mot.echl.abort

Numeric value: 196

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the mechanism to abort the current move.

Parameter values:
Irrelevant.

Returned values:
Echoed unchanged. Also, if a move or initialization was in progress, that cid will also come back with `move.abort` as parameter.

cid.mot.disp.step

Numeric value: 200

Source: Host.

Destination: Motor transputer 2.

Function:
Commands the transputer to move the cross-disperser grating mechanism a specific number of steps (i.e. a relative move).

Parameter values:
The number of steps. If the required number of steps is negative (i.e. in the counterclockwise direction) the parameter is the number of steps plus 999999. This is because of a limitation in the host-transputer protocol that prevents negative integers going from the host to the transputers.

Returned values:
If the move was successful and no limit switches were encountered, the value is `move.success`. If a limit switch was encountered, the value is `move.limit.switch`. If the move was aborted by a command from the host, the value is `move.abort`.

cid.mot.disp.pos

Numeric value: 201

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the code to move the cross-disperser mechanism to a specific position (i.e. an absolute move).

Parameter values:
The position in increments of 0.01 °. Minimum position is 0 (0 °) and maximum is 5800 (58°).

Returned values:

`move.success`	the move was completed successfully
`move.pos.unknown`	the mechanism initial position wasn't known, so we can't move
`motor.invalid.parameter`	the requested position was out of range
`move.limit.switch`	the move ended prematurely when a limit switch was tripped
`move.abort`	the move was aborted on command from the host

cid.mot.disp.init

Numeric value: 202

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the code to initialize the cross-disperser mechanism. First it seeks the home switch or its backup, and if one of those is successfully found, moves the mechanism to the `init.home` position (35°).

Parameter values:
Irrelevant.

Returned values:

cid.mot.disp.initloc

Numeric value: 203

Source: Host.

Destination: Motor transputer 2.

Function:
Sets the cross-disperser mechanism position (in steps) to a new value. An engineering-level command, used in the case where the home switches aren't working but we actually know where the mechanism is.

Parameter values:
Position in steps. Must be between zero and `disp.total.steps`, the number if steps in the full range of motion of the mechanism.

Returned values:
Echoed unchanged unless the step value is outside the range, in which case it sends back `motor.invalid.parameter`.

cid.mot.disp.location

Numeric value: 204

Source: Host.

Destination: Motor transputer 2.

Function:
Requests the code to send back the location of the cross-disperser mechanism.

Parameter values:
Irrelevant.

Returned values:
The position in steps (not degrees). A value of -1 indicates the mechanism position is unknown (i.e. it hasn't been initialized).

cid.mot.disp.switch

Numeric value: 205

Source: Host.

Destination: Motor transputer 2.

Function:
Prompts the code to check the status of the limit and home microswitches in the cross-disperser mechanism.

Parameter values:
Irrelevant.

Returned values:
An integer containing 4 bits set to reflect the state of the switches. A 1 means the switch is open and a 0 means it's closed.

cid.mot.disp.abort

Numeric value: 206

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the mechanism to abort the current move.

Parameter values:
Irrelevant.

Returned values:
Echoed unchanged. Also, if a move or initialization was in progress, that cid will also come back with `move.abort` as parameter.

cid.mot.calm.step

Numeric value: 210

Source: Host.

Destination: Motor transputer 2.

Function:
Commands the transputer to move the calibration unit pickoff mirror mechanism a specific number of steps (i.e. a relative move).

Parameter values:
The number of steps. If the required number of steps is negative (i.e. in the counterclockwise direction) the parameter is the number of steps plus 999999. This is because of a limitation in the host-transputer protocol that prevents negative integers going from the host to the transputers.

Returned values:
If the move was successful and no limit switches were encountered, the value is `move.success`. If a limit switch was encountered, the value is `move.limit.switch`. If the move was aborted by a command from the host, the value is `move.abort`.

cid.mot.calm.pos

Numeric value: 211

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the code to move the calibration unit mirror in or out of the beam.

Parameter values:
0 or 1, where 0 is out and 1 is in.

Returned values:

`move.success`	the move was completed successfully
`move.pos.unknown`	the mechanism initial position wasn't known, so we can't move
`motor.invalid.parameter`	the requested position was out of range
`move.limit.switch`	the move ended prematurely when a limit switch was tripped
`move.abort`	the move was aborted on command from the host
`move.pos.switch`	the move didn't end on the correct position switch

cid.mot.calm.init

Numeric value: 212

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the code to initialize the calibration unit mirror mechanism. First it seeks the home switch or its backup, and if one of those is successfully found, moves the mechanism to the `init.home` position (in this case the out position).

Parameter values:
Irrelevant.

Returned values:

cid.mot.calm.initloc

Numeric value: 213

Source: Host.

Destination: Motor transputer 2.

Function:
Sets the calibration unit flip mirror position (in steps) to a new value. An engineering-level command, used in the case where the home switches aren't working but we actually know where the mechanism is.

Parameter values:
Position in steps. Must be between zero and `calm.total.steps`, the number if steps in the full range of motion of the mechanism.

Returned values:
Echoed unchanged unless the step value is outside the range, in which case it sends back `motor.invalid.parameter`.

cid.mot.calm.location

Numeric value: 214

Source: Host.

Destination: Motor transputer 2.

Function:
Requests the code to send back the location of the calibration unit mirror.

Parameter values:
Irrelevant.

Returned values:
The position in steps (not 0 or 1 for in or out of the beam). A value of -1 indicates the mechanism position is unknown (i.e. it hasn't been initialized).

cid.mot.calm.switch

Numeric value: 215

Source: Host.

Destination: Motor transputer 2.

Function:
Prompts the code to check the status of the limit and home microswitches in the calibration unit mirror mechanism.

Parameter values:
Irrelevant.

Returned values:
An integer containing 4 bits set to reflect the state of the switches. A 1 means the switch is open and a 0 means it's closed.

cid.mot.calm.abort

Numeric value: 216

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the mechanism to abort the current move.

Parameter values:
Irrelevant.

Returned values:
Echoed unchanged. Also, if a move or initialization was in progress, that cid will also come back with `move.abort` as parameter.

cid.mot.calp.step

Numeric value: 220

Source: Host.

Destination: Motor transputer 2.

Function:
Commands the transputer to move the calibration unit pinhole mechanism a specific number of steps (i.e. a relative move).

Parameter values:
The number of steps. If the required number of steps is negative (i.e. in the counterclockwise direction) the parameter is the number of steps plus 999999. This is because of a limitation in the host-transputer protocol that prevents negative integers going from the host to the transputers.

Returned values:
If the move was successful and no limit switches were encountered, the value is `move.success`. If a limit switch was encountered, the value is `move.limit.switch`. If the move was aborted by a command from the host, the value is `move.abort`.

cid.mot.calp.pos

Numeric value: 221

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the code to move the calibration unit pinhole slide in or out of the beam.

Parameter values:
0 or 1, where 0 is out and 1 is in.

Returned values:

`move.success`	the move was completed successfully
`move.pos.unknown`	the mechanism initial position wasn't known, so we can't move
`motor.invalid.parameter`	the requested position was out of range
`move.limit.switch`	the move ended prematurely when a limit switch was tripped
`move.abort`	the move was aborted on command from the host
`move.pos.switch`	the move didn't end on the correct position switch

cid.mot.calp.init

Numeric value: 222

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the code to initialize the mechanism. First it seeks the home switch or its backup, and if one of those is successfully found, moves the mechanism to the `init.home` position (in this case the out position).

Parameter values:
Irrelevant.

Returned values:

cid.mot.calp.initloc

Numeric value: 223

Source: Host.

Destination: Motor transputer 2.

Function:
Sets the calibration unit pinhole mechanism position (in steps) to a new value. An engineering-level command, used in the case where the home switches aren't working but we actually know where the mechanism is.

Parameter values:
Position in steps. Must be between zero and `calp.total.steps`, the number if steps in the full range of motion of the mechanism.

Returned values:
Echoed unchanged unless the step value is outside the range, in which case it sends back `motor.invalid.parameter`.

cid.mot.calp.location

Numeric value: 224

Source: Host.

Destination: Motor transputer 2.

Function:
Requests the code to send back the location of the calibration unit pinhole mechanism.

Parameter values:
Irrelevant.

Returned values:
The position in steps (not 0 or 1 for in or out of the beam). A value of -1 indicates the mechanism position is unknown (i.e. it hasn't been initialized).

cid.mot.calp.switch

Numeric value: 225

Source: Host.

Destination: Motor transputer 2.

Function:
Prompts the code to check the status of the limit and home microswitches in the calibration unit pinhole slide mechanism.

Parameter values:
Irrelevant.

Returned values:
An integer containing 4 bits set to reflect the state of the switches. A 1 means the switch is open and a 0 means it's closed.

cid.mot.calp.abort

Numeric value: 226

Source: Host.

Destination: Motor transputer 2.

Function:
Tells the mechanism to abort the current move.

Parameter values:
Irrelevant.

Returned values:
Echoed unchanged. Also, if a move or initialization was in progress, that cid will also come back with `move.abort` as parameter.

cid.mot.calc.step

Numeric value: 230

Source: Host.

Destination: Motor transputer 1.

Function:
Commands the transputer to move the calibration unit dust cover mechanism a specific number of steps (i.e. a relative move).

Parameter values:
The number of steps. If the required number of steps is negative (i.e. in the counterclockwise direction) the parameter is the number of steps plus 999999. This is because of a limitation in the host-transputer protocol that prevents negative integers going from the host to the transputers.

Returned values:
If the move was successful and no limit switches were encountered, the value is `move.success`. If a limit switch was encountered, the value is `move.limit.switch`. If the move was aborted by a command from the host, the value is `move.abort`.

cid.mot.calc.pos

Numeric value: 231

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the code to open or close the calibration unit dust cover.

Parameter values:
0 or 1, where 0 is open and 1 is closed.

Returned values:

`move.success`	the move was completed successfully
`move.pos.unknown`	the mechanism initial position wasn't known, so we can't move
`motor.invalid.parameter`	the requested position was out of range
`move.limit.switch`	the move ended prematurely when a limit switch was tripped
`move.abort`	the move was aborted on command from the host
`move.pos.switch`	the move didn't end on the correct position switch

cid.mot.calc.init

Numeric value: 232

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the code to initialize the calibration unit dust cover mechanism. First it seeks the home switch or its backup, and if one of those is successfully found, moves the mechanism to the `init.home` position (in this case the out position).

Parameter values:
Irrelevant.

Returned values:

cid.mot.calc.initloc

Numeric value: 233

Source: Host.

Destination: Motor transputer 1.

Function:
Sets the calibration unit dust cover mechanism position (in steps) to a new value. An engineering-level command, used in the case where the home switches aren't working but we actually know where the mechanism is.

Parameter values:
Position in steps. Must be between zero and `calc.total.steps`, the number if steps in the full range of motion of the mechanism.

Returned values:
Echoed unchanged unless the step value is outside the range, in which case it sends back `motor.invalid.parameter`.

cid.mot.calc.location

Numeric value: 234

Source: Host.

Destination: Motor transputer 1.

Function:
Requests the code to send back the location of the calibration unit dust cover mechanism.

Parameter values:
Irrelevant.

Returned values:
The position in steps (not 0 or 1 for closed or open). A value of -1 indicates the mechanism position is unknown (i.e. it hasn't been initialized).

cid.mot.calc.switch

Numeric value: 235

Source: Host.

Destination: Motor transputer 1.

Function:
Prompts the code to check the status of the limit and home microswitches in the calibration unit dust cover mechanism.

Parameter values:
Irrelevant.

Returned values:
An integer containing 4 bits set to reflect the state of the switches. A 1 means the switch is open and a 0 means it's closed.

cid.mot.calc.abort

Numeric value: 236

Source: Host.

Destination: Motor transputer 1.

Function:
Tells the mechanism to abort the current move.

Parameter values:
Irrelevant.

Returned values:
Echoed unchanged. Also, if a move or initialization was in progress, that cid will also come back with `move.abort` as parameter.

cid.mot.abort

Numeric value: 240

Source: Host.

Destination: Motor transputer 1.

Function:
Tells transputer 1 to abandon the current move. OBSOLETE

Parameter values:
Irrelevant.

Returned values:

cid.mot.quit

Numeric value: 241

Source: Host

Destination: All motor transputers.

Function:
Requests total shutdown of all motor processes.

Parameter values:
Irrelevant.

Returned values:
Comes back with parameter = 1 when all motor processes have reported back that they got the quit command. Motor 2 waits for all its own sections plus motor 3 to reply, then it sends the command to motor 1 which waits for all its own sections to report back also, then replies to the host.

cid.mot.busy

Numeric value: 242

Source: Any motor transputer.

Destination: Host.

Function:
Tells the host that a command can't be handled because the transputer is already busy servicing a previous command. Now OBSOLETE; instead the value `motor.busy` is sent back as the parameter to the cid that can't be serviced.

Parameter values:
Irrelevant

Returned values:
N/A

cid.mot.invalid.command

Numeric value: 243

Source: Any motor transputer.

Destination: Host.

Function:
Signals that an invalid cid was received. Should never happen, since the command routing is hard-coded. Now OBSOLETE; instead we return the value `mot.invalid.command` as the parameter of the command that caused the problem.

Parameter values:
Irrelevant.

Returned values:
N/A

cid.mot.invalid.command.parameter

Numeric value: 244

Source: Any motor transputer.

Destination: Host.

Function:
Signals to the host that the parameter of a recently sent cid was invalid. Now OBSOLETE; instead we return the value `motor.invalid.parameter` with the cid that carried the bad parameter.

Parameter values:
Irrelevant

Returned values:
N/A

cid.test

Numeric value: 255

Source: Host.

Destination: Any transputer, depending on the parameter value.

Function:
Used to check the integrity of the transputer network. The cid is routed according to the value of the parameter to a particular transputer, which adds 1000 to the parameter and sends it back. For parameter values 5 & 23 the added 1000 doesn't happen. These values are just passed around the loopbacks connecting the acquisition transputers and the clock generators in each camera. This cid allows us to figure out which link is broken or which transputer is hung up. For details see NSPN21.

Parameter values:
1 = root, 2 = slit-viewing camera clock, 3,4 = slit-viewing camera acquisition, 5 = slit-viewing camera loopback, 6 = spectrometer clock, 7 - 22 = spectrometer acquisition, 23 = spectrometer loopback, 24 & 25 = serial 1 & serial 2, and 26, 27 & 28 = motor 1, motor 2, & motor 3.

Returned values:
Original parameter + 1000, except for parameter values 5 & 23 as noted above.