Z5 command protocol

V02.1

Anthony Lee

2014/11/6

Revision

Version no.	Date	Description
V02. 1	2014/11/06	1st formal release.

Preparation

1. Connect the board (SD1220) to PC via USB and UART ports.
2. Serial port setting:

   Baud Rate : 9600

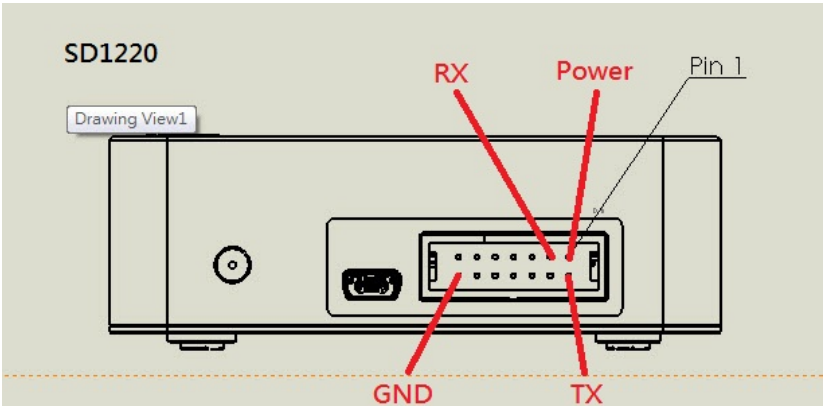
   Parity Bit : NONE

   Data Bit : 8

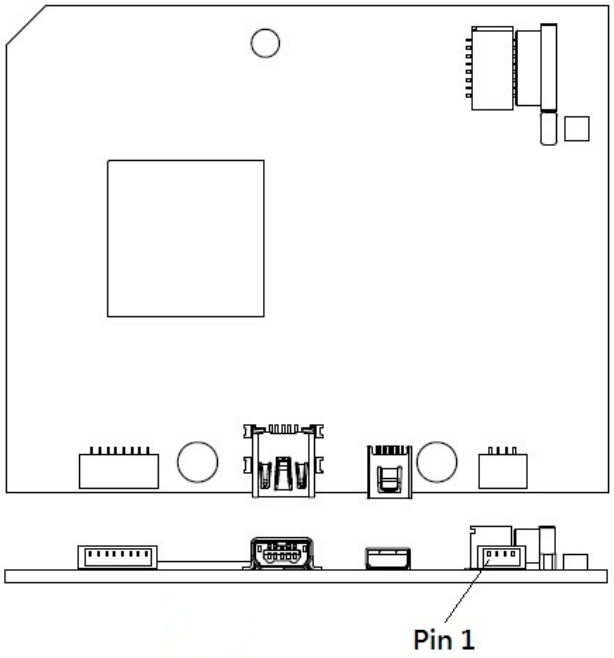
   Stop Bit : 1
3. Commands can be sent to the board via UART after getting message "*READYREADY*" as 10 bytes.
4. Command syntax looks like the following format:

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_FW_VERSION Example: 0x56464F09	None	None	None
DEVCMD_SET_EXTERNEL_PORT Example: 0x70654F09	operation 0x00000002	None	None
DEVCMD_SET_TRIGGER_IO Example: 0x69744F09	trigger_type 0x80000002	trigger_edge 0x00000001	None
DEVCMD_SET_SPECTRUM_RANGE Example 1: 0x72734F09 Example 2: 0x72734F09	type 0x00000000 0x0000000A	Number 0xFFFFFFFF 0x00000001	position[number] None 0x030C017C

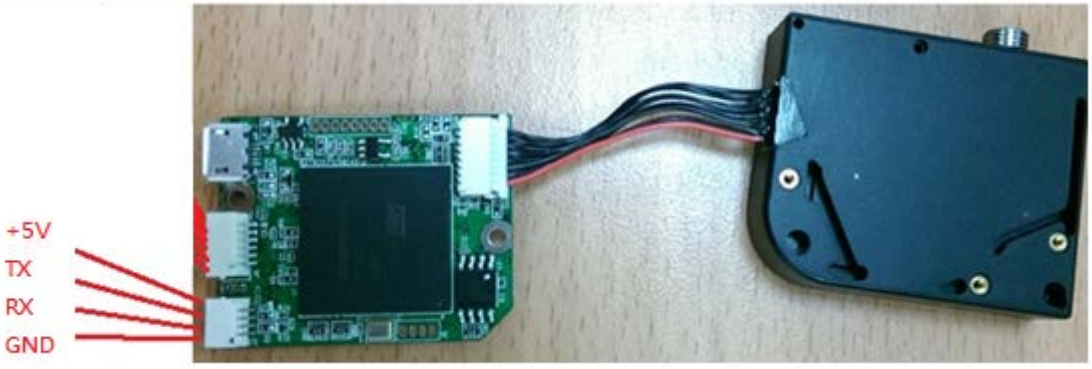
5. Physical UART connection is illustrated as below. The first plot illustrates for SD1220, the second plot illustrates for board model CB-5xH1, and the third plot (picture) illustrates for board model UM1280/UM2280. For more board information, please contact the Sales Department.



CB-5xH2



UM2280/UM1280



Pin No.	Direction	Pin Name
1	+5V	+5V
2	Output	TX
3	Input	RX
4	GND	GND

Commands

/*command symbol*/ /*command code*/

DEVCMD_GET_FW_VERSION ‘0x09’ ‘0x4F’ ‘0x46’ ‘0x56’

DEVCMD_GET_FW_BUILD ‘0x09’ ‘0x4F’ ‘0x46’ ‘0x42’

DEVCMD_SPECTRUM_ACQUIRE ‘0x09’ ‘0x4F’ ‘0x53’ ‘0x51’

DEVCMD_SPECTRUM_ONESHOT ‘0x09’ ‘0x4F’ ‘0x53’ ‘0x4F’

DEVCMD_SPECTRUM_CONTINUE ‘0x09’ ‘0x4F’ ‘0x53’ ‘0x43’ start/stop

//start:1; stop:0

DEVCMD_SET_EXTERNEL_PORT ‘0x09’ ‘0x4F’ ‘0x65’ ‘0x70’ operation

// operation:

//BIT0: Triggered (Input)

//BIT1: Lamp On (Output)

//BIT2: GPIO3 (Output)

//BIT3: GPIO2 (Output)

//BIT4: GPIO1 (Output)

//BIT5: GPIO0 (Output)

//BIT6~BIT31: Reserved

DEVCMD_GET_EXTERNEL_PORT ‘0x09’ ‘0x4F’ ‘0x45’ ‘0x50’

**DEVCMD_SET_TRIGGER_IO ‘0x09’ ‘0x4F’ ‘0x74’ ‘0x69’ trigger_type

trigger_edge

// trigger_type:

//BIT0: continuous triggered

//BIT1: discrete triggered

//BIT2~3: reserved

//BIT4: XYZ

//BIT5: reserved

//BIT6: xyz

//BIT7: reserved

//BIT8: Lab

//BIT9: reserved

//BIT10: reserved

//BIT11: reserved

//BIT12: reserved

//BIT13: 1960UCS

//BIT14: 1960ucs

//BIT15: 1976UCS

//BIT16: 1976ucs

//BIT17: CCT

//BIT18: DominantWavelength
//BIT19: Purity
//BIT20: CIEWhiteness
//BIT21: CIETin
//BIT22: CRI
//BIT23: CQS
//BIT24: Luminance
//BIT25: RadiantPower
//BIT26: CentralWavelength
//BIT27: reserved
//BIT28: LambdaP
//BIT29: PPFD
// BIT30: NormalFactor

// BIT31: raw intensity

// trigger_edge:
//0: Falling edge triggered
//1: Raising edge triggered

DEVCMD_SET_INTEGRATION_TIME	'0x09' '0x4F' '0x69' '0x74'	it(us)
DEVCMD_GET_INTEGRATION_TIME	'0x09' '0x4F' '0x49' '0x54'	
DEVCMD_SET_AVERAGE	'0x09' '0x4F' '0x61' '0x76'	average
DEVCMD_GET_AVERAGE	'0x09' '0x4F' '0x41' '0x56'	
DEVCMD_SET_RUNNING_AVERAGE	'0x09' '0x4F' '0x72' '0x76'	average
DEVCMD_GET_RUNNING_AVERAGE	'0x09' '0x4F' '0x52' '0x56'	
DEVCMD_SET_BOXCAR	'0x09' '0x4F' '0x62' '0x63'	boxcar
DEVCMD_GET_BOXCAR	'0x09' '0x4F' '0x42' '0x43'	
DEVCMD_SET_LINEARITY	'0x09' '0x4F' '0x6C' '0x61'	on/off
DEVCMD_GET_LINEARITY	'0x09' '0x4F' '0x4C' '0x41'	
DEVCMD_SET_INTENSITY	'0x09' '0x4F' '0x69' '0x69'	mode
DEVCMD_GET_INTENSITY	'0x09' '0x4F' '0x49' '0x49'	
DEVCMD_SET_STRAYLIGHT	'0x09' '0x4F' '0x73' '0x6C'	on/off
DEVCMD_GET_STRAYLIGHT	'0x09' '0x4F' '0x53' '0x4C'	

// mode:
// off:0
//absolute:1
//contrast:2

DEVCMD_SET_SPECTRUM_RANGE	‘0x09’ ‘0x4F’ ‘0x73’ ‘0x72’	type number position[number]
		//type:
		//0:pixel mode;
		//type>0: wavelength;
		//number: number of position
		//position: pixel of wavelength range
DEVCMD_GET_SPECTRUM_RANGE	‘0x09’ ‘0x4F’ ‘0x53’ ‘0x52’	
DEVCMD_SET_GROUP_AVERAGE	‘0x09’ ‘0x4F’ ‘0x67’ ‘0x76’	on/off
DEVCMD_GET_GROUP_AVERAGE	‘0x09’ ‘0x4F’ ‘0x47’ ‘0x56’	
DEVCMD_SET_BACKGROUND	‘0x09’ ‘0x4F’ ‘0x62’ ‘0x67’	user/on/off
		//user:2; on:1; off:0
DEVCMD_GET_BACKGROUND	‘0x09’ ‘0x4F’ ‘0x42’ ‘0x47’	
DEVCMD_GET_BACKGROUND_VALUE	‘0x09’ ‘0x4F’ ‘0x42’ ‘0x56’	
DEVCMD_SET_STD_WHITE	‘0x09’ ‘0x4F’ ‘0x73’ ‘0x77’	user/on/off
		//user:2; on:1; off:0
DEVCMD_GET_STD_WHITE	‘0x09’ ‘0x4F’ ‘0x53’ ‘0x57’	
DEVCMD_GET_ STD_WHITE _VALUE	‘0x09’ ‘0x4F’ ‘0x53’ ‘0x56’	
DEVCMD_WAVELENGTH_ACQUIRE	‘0x09’ ‘0x4F’ ‘0x57’ ‘0x51’	
DEVCMD_WAVELENGTH_PIECE_ACQUIRE	‘0x09’ ‘0x4F’ ‘0x57’ ‘0x50’	pix_range
DEVCMD_FRAMESIZE_ACQUIRE	‘0x09’ ‘0x4F’ ‘0x46’ ‘0x4F’	
DEVCMD_SET_AUTO_INTEGRATION_TIME	‘0x09’ ‘0x4F’ ‘0x61’ ‘0x74’	
*DEVCMD_SET_COLOR_REFERENCE	‘0x09’ ‘0x4F’ ‘0x63’ ‘0x72’	
*DEVCMD_GET_COLOR_REFERENCE	‘0x09’ ‘0x4F’ ‘0x43’ ‘0x52’	
*DEVCMD_SET_COLOR_INFO	‘0x09’ ‘0x4F’ ‘0x63’ ‘0x69’	irradiance observer illuminant
		// irradiance:
		//contrast emissive:0
		//contrast reflective:1
		//absolute emissive:2
		// absolute reflective:3
		// observer:
		//2 degree:0
		//10 degree:1
		// illuminant:
		//cie illuminant A:0
		//cie illuminant B:1

//cie illuminant C:2
//cie illuminant D50:3
//cie illuminant D55:4
//cie illuminant D65:5
//cie illuminant D75:6
//cie illuminant E: 7
//cie illuminant F1:8
//cie illuminant F2:9
//cie illuminant F3:10
//cie illuminant F4:11
//cie illuminant F5:12
//cie illuminant F6:13
//cie illuminant F7:14
//cie illuminant F8:15
//cie illuminant F9:16
//cie illuminant F10:17
//cie illuminant F11:18
//cie illuminant F12:19

*DEVCMD_GET_COLOR_INFO ‘0x09’ ‘0x4F’ ‘0x43’ ‘0x49’

*DEVCMD_GET_COLOR_VALUE ‘0x09’ ‘0x4F’ ‘0x43’ ‘0x56’ color_space

//color_space
//BIT0: XYZ
//BIT1: reserved
//BIT2: xyz
//BIT3: reserved
//BIT4: Lab
//BIT5: reserved
//BIT6: reserved
//BIT7: reserved
//BIT8: reserved
//BIT9: 1960UCS
//BIT10: 1960ucs
//BIT11: 1976UCS
//BIT12: 1976ucs
//BIT13: CCT
//BIT14: DominantWavelength
//BIT15: Purity
//BIT16: CIEWhiteness

//BIT17: CIETin
//BIT18: ColorRenderingIndex
//BIT19: ColorQualityScale
//BIT20 Luminance
//BIT21: RadiantPower
//BIT22: CentralWavelength
//BIT23: reserved
//BIT24: LambdaP
//BIT25: PPFD
//BIT26~28: reserved
//BIT29: AutoIntTimeSet
//BIT30: NormalFactor
// BIT31: raw intensity

*DEVCMD_SPECTRUM_RATIO_ACQUIRE ‘0x09’ ‘0x4F’ ‘0x52’ ‘0x51’

*DEVCMD_TRANSMISSIOM_ACQUIRE ‘0x09’ ‘0x4F’ ‘0x54’ ‘0x51’

*DEVCMD_REFLECTION_ACQUIRE ‘0x09’ ‘0x4F’ ‘0x46’ ‘0x51’

DEVCMD_SET_BAUDRATE ‘0x09’ ‘0x4F’ ‘0x62’ ‘0x64’ baud_rate

DEVCMD_GET_BAUDRATE ‘0x09’ ‘0x4F’ ‘0x42’ ‘0x44’

DEVCMD_INTENSITY_CALIBRATE ‘0x09’ ‘0x4F’ ‘0x69’ ‘0x63’

DEVCMD_INTENSITY_CAL_DATE_GET ‘0x09’ ‘0x4F’ ‘0x49’ ‘0x47’

DEVCMD_INTENSITY_STDTABLE_SET ‘0x09’ ‘0x4F’ ‘0x69’ ‘0x6C’ length
table[2][length] (table[0]:wavelength,
table[1]:intensity)
//wavelength:4 bytes (integer) with unit 0.1nm,
//intensity: 8 bytes as INT32 fraction;
INT8 integer;
INT8 exponent;
UINT8 pad[2];

DEVCMD_INTENSITY_STDTABLE_GET ‘0x09’ ‘0x4F’ ‘0x49’ ‘0x4C’

DEVCMD_LAMBDA_TO_PIX ‘0x09’ ‘0x4F’ ‘0x4C’ ‘0x50’ wavelength
(scaled with 65536)

DEVCMD_GET_SERIAL_NUMBER ‘0x09’ ‘0x4F’ ‘0x53’ ‘0x4E’

DEVCMD_GET_MODEL_NAME ‘0x09’ ‘0x4F’ ‘0x4D’ ‘0x4E’

DEVCMD_GET_SLIT_TYPE ‘0x09’ ‘0x4F’ ‘0x53’ ‘0x54’

DEVCMD_GET_WAVELENGTH_START_END ‘0x09’ ‘0x4F’ ‘0x57’ ‘0x45’

DEVCMD_READ_USER_ROM ‘0x09’ ‘0x4F’ ‘0x52’ ‘0x55’ length address

DEVCMD_WRITE_USER_ROM ‘0x09’ ‘0x4F’ ‘0x77’ ‘0x75’ length
address

	data[length]
DEVCMD_ERASE_USER_ROM	'0x09' '0x4F' '0x65' '0x75'
DEVCMD_POWER_SAVE	'0x09' '0x4F' '0xFF' '0xFF'
DEVCMD_POWER_WAKE	'0x09' '0x4F' '0x00' '0x00'
DEVCMD_INTENSITY_ABS_NRML_FAC_GET	'0x09' '0x4F' '0x41' '0x4E'

*Marked with * is an optional function. Please contact sales department for the detail.

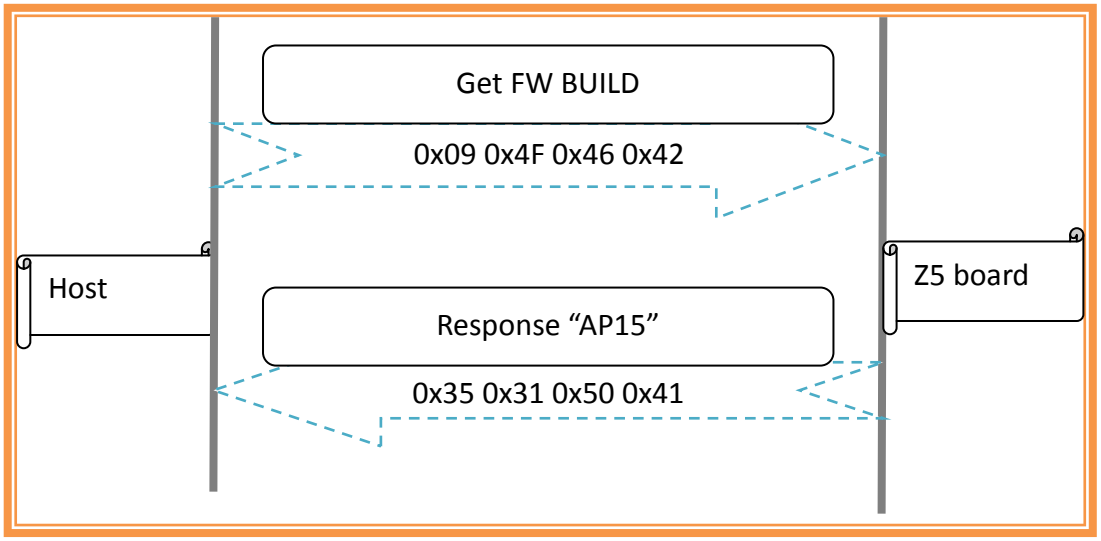
Color operation in command ***DEVCMD_SET_TRIGGER_IO* is an option. Please contact sales department for the detail.

Detailed Protocol

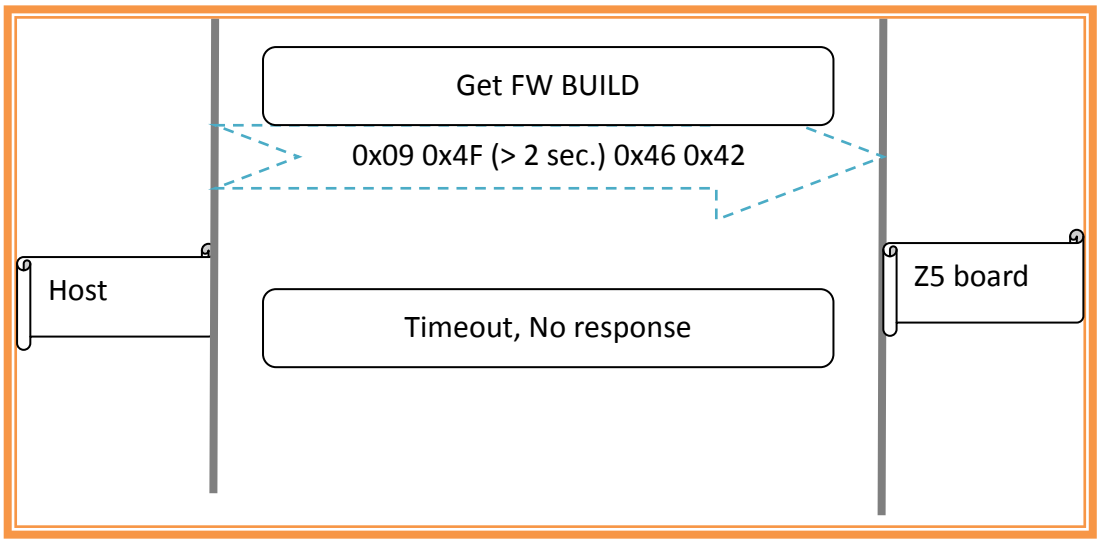
Each byte of the command should be completed within 2 seconds, otherwise, timeout mechanism is triggered and the command should be resend from the beginning. One example is shown as below:

Ex.

To get the FW BUILD, send the command sequence '0x09' '0x4F' '0x46' '0x42' then get the response '0x31' '0x30' '0x30' '0x42' ("B001"). This hand shaking completes the command ***DEVCMD_GET_FW_BUILD***.



Timeout mechanism is trigger if one of these 4 characters, 0x09, 0x4F, 0x46, or 0x42 is received or missed after 2 sec.



Detailed Commands

DEVCMD_GET_FW_VERSION

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_FW_VERSION 0x56464F09	None	None	None

Sending '0x09' '0x4F' '0x46' '0x56' sequentially gets the 4-byte response to show the firmware version.

DEVCMD_GET_FW_BUILD

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_FW_BUILD 0x42464F09	None	None	None

Sending '0x09' '0x4F' '0x46' '0x42' sequentially gets the 4-byte response to show the firmware build version.

DEVCMD_SPECTRUM_ACQUIRE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SPECTRUM_ACQUIRE 0x51534F09	None	None	None

Sending '0x09' '0x4F' '0x53' '0x51' sequentially gets the sensor data (2-byte each with LSB). These data may be compensated by "background calibration (with straylight compensation)", "linearity calibration", and "intensity calibration", with commands ***DEVCMD_SET_BACKGROUND (and DEVCMD_SET_STRALIGHT)***, ***DEVCMD_SET_LINEARITY***, and ***DEVCMD_SET_INTENSITY***, respectively. The total data count can be checked with command ***DEVCMD_FRAMESIZE_ACQUIRE***. The corresponding wavelength can be checked with command ***DEVCMD_WAVELENGTH_ACQUIRE***.

Please be noted that the spectra data may not be reliable once one or more than one pixel data are presented as 65535.

DEVCMD_SPECTRUM_ONESHOT

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SPECTRUM_ONESHOT 0x4F534F09	None	None	None

Sending 0x09 0x4F 0x53 0x4F sequentially gets one shot (without any average) spectrum data. Sensor data start to acquire after this command. Thus, for integration time = 50ms, sensor data will be received about 50ms after sending this command.

Please be noted that the spectra data may not be reliable once one or more than one pixel data are presented as 65535.

DEVCMD_SPECTRUM_CONTINUE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SPECTRUM_CONTINUE 0x43534F09	On/Off	None	None

Sending 0x09 0x4F 0x53 0x43 0x01 0x00 0x00 0x00 (4-byte LSB necessary) serially results acquiring sensor data automatically with time period as integration time (set by ***DEVCMD_SET_INTEGRATION_TIME***). Sending 0x09 0x4F 0x53 0x43 0x00 0x00 0x00 0x00 (4-byte LSB necessary) serially stops data auto acquiring.

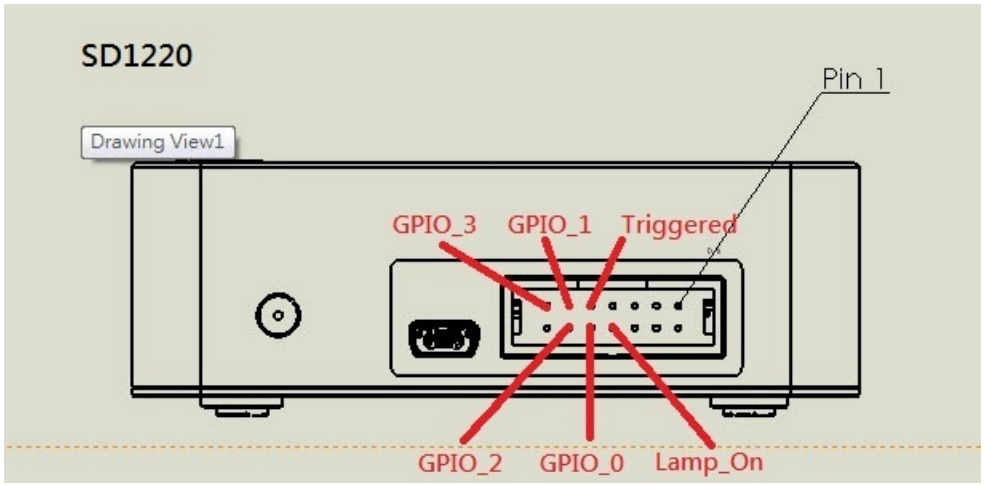
Please be noted that only the stop command (0x09 0x4F 0x53 0x43 0x00 0x00 0x00 0x00) can be accepted after command sequence 0x09 0x4F 0x53 0x43 0x01 0x00 0x00 0x00.

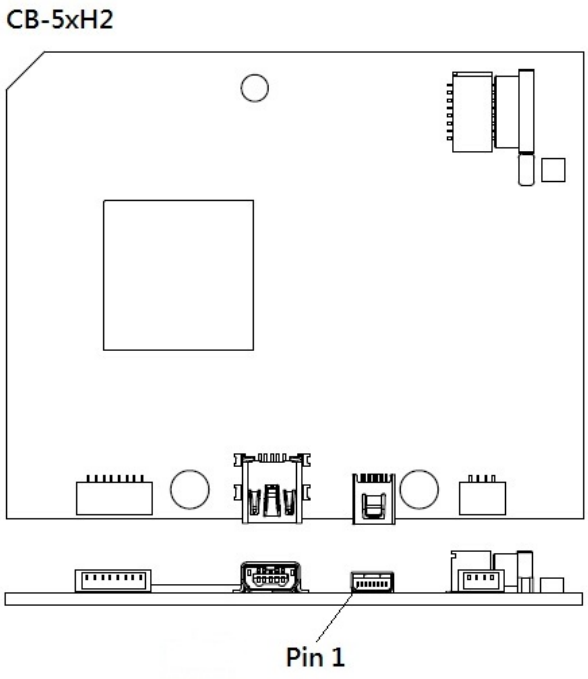
Please be noted that the spectra data may not be reliable once one or more than one pixel data are presented as 65535.

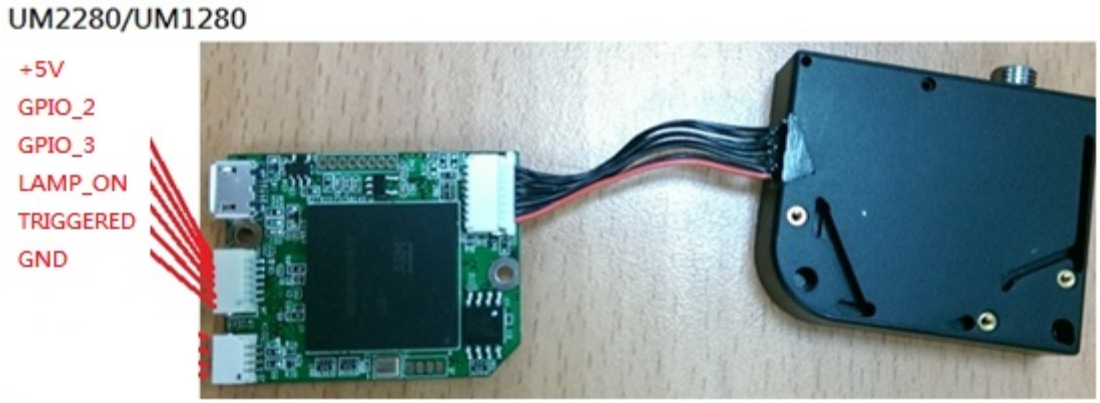
DEVCMD_SET_EXTERNAL_PORT

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)
DEVCMD_SET_EXTERNAL_PORT 0x70654F09	Operation BIT0: Input (Trigger) BIT1: Output (Lamp_On) BIT2: Output (GPIO3) BIT3: Output (GPIO2) BIT4: Output (GPIO1) BIT5: Output (GPIO0) BIT6~BIT31: reserved	None

There are 6 digital input/output (1 input and 5 outputs) can be read/set from the host. The plots below show the position of each pin. The first plot is illustrated for SD1220, the second plot is illustrated for board model CB-5xH1, and the third plot (picture) is illustrated for board model UM1280/UM2280. For more board information, please contact the Sales Department.







Pin No.	Direction	Pin Name
1	Power	+5V
2	Output	GPIO0
3	Output	GPIO1
4	Output	GPIO2
5	Output	GPIO3
6	Output	Lamp_ON
7	Input	Triggered
8	GND	GND

Sending 0x09 0x4F 0x65 0x70 sequentially followed by operation value (4-byte with LSB) operates the output pins including Lamp_On, GPIO_0, GPIO_1, GPIO_2, and GPIO_3 simultaneously. For example, sending the command sequence 0x09 0x4F 0x65 0x70 0x06 0x00 0x00 0x00 sets the Lamp_On pin (BIT1) and GPIO_3 pin (BIT2) ON, and sets the

GPIO_2 pin (BIT3), GPIO_1 pin (BIT4), and GPIO_0 pin (BIT5) OFF at the same time. Please be noted that it may cause some problem by operating other bits of the operation value.

DEVCMD_GET_ EXTERNAL_PORT

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_EXTERNAL_PORT 0x50454F09	None	None	None

Sending 0x09 0x4F 0x45 0x50 sequentially gets the status (4-byte with LSB) of pins Triggered, Lamp_On, GPIO_0, GPIO_1, GPIO_2, and GPIO_3. The list below shows the bit position for each pin.

//BIT0: Triggered (Input)

//BIT1: Lamp On (Output)

//BIT2: GPIO3 (Output)

//BIT3: GPIO2 (Output)

//BIT4: GPIO1 (Output)

//BIT5: GPIO0 (Output)

//BIT6~BIT31: Reserved

Please be noted that the status of BIT6~BIT31 should be ignored.

DEVCMD_SET_TRIGGER_IO *(color operation is an option)*

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)
DEVCMD_SET_TRIGGER_IO 0x69744F09	Trigger type BYTE0: mode BIT0: continuous triggered BIT1: discrete triggered BIT2~3: reserved BYTE4~31: color operation BIT4: XYZ (12 bytes) (**F**) BIT5: reserved BIT6: xyz (12 bytes) BIT7: reserved BIT8: CIE Lab (12 bytes) BIT9: reserved BIT10: reserved	Trigger edge 0: Falling edge triggered 1: Raising edge triggered

	BIT11: reserved BIT12: reserved BIT13: 1960UCS (12 bytes) BIT14:1960ucs (12 bytes) BIT15: 1976UCS (12 bytes) BIT16: 1976ucs (12 bytes) BIT17: CCT (4 bytes) BIT18: Dominant Wavelength (4 bytes) BIT19: Purity (4 bytes) BIT20: CIE Whiteness (4 bytes) BIT21: CIE Tin (4 bytes) BIT22: Color Rendering Index (17x4 bytes) BIT23: Color Quality Scale (16x4 bytes) BIT24: Luminance (4 bytes) (**F**) BIT25: Radiant Power (4 bytes) (**F**) BIT26: Central Wavelength (4 bytes) BIT27: reserved BIT28 Lambda Peak (4 bytes) BIT29: PPFD BIT30: Normal Factor (4 bytes) (**F**) BIT31: raw intensity	

Spectra data can be acquired with external trigger signal. Sending 0x09 0x4F 0x74 0x69 sequentially followed by 2 arguments, “trigger type” and “trigger edge” (4-byte each with LSB), turns this function ON. For example, sending 0x09 0x4F 0x74 0x69 0x01 0x00 0x00 0x80 0x00 0x00 0x00 0x00 sequentially turns on the external trigger function and the data acquired at the falling edge of the trigger signal. On the other hand, sending 0x09 0x4F 0x74 0x69 0x01 0x00 0x00 0x80 0x01 0x00 0x00 0x00 sequentially turns on the external trigger function and the data acquired at the raising edge of the triggered signal.

The pin of the triggered signal is the same as the Triggered pin in ***DEVCMD_GET_EXTERNAL_PORT*** command.

Please be noted that only command sequence ‘0x09 0x4F 0x74 0x69 trigger_type trigger_edge’ can be accepted after command sequence 0x09 0x4F 0x74 0x69 0x01 0x00 0x00 0x80 0x01 0x00 0x00 0x00 or command sequence 0x09 0x4F 0x74 0x69 0x01 0x00

0x00 0x80 0x00 0x00 0x00 0x00. To send commands other than ***DEVCMD_SET_TRIGGER_IO***, send command sequence 0x09 0x4F 0x74 0x69 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 to stop the triggered function firstly.

There are 2 types of triggered mode in command ***DEVCMD_SET_TRIGGER_IO***. One is called continuous type and the other one is called discrete type. To start the continuous mode with raising edge triggered, send 0x09 0x4F 0x74 0x69 0x01 0x00 0x00 0x80 0x01 0x00 0x00 0x00 sequentially. Spectra data will be sent out each triggered edge continuously. To start the discrete mode with raising edge triggered, send 0x09 0x4F 0x74 0x69 0x02 0x00 0x00 0x80 0x01 0x00 0x00 0x00 sequentially. Spectra data will be sent out just once in the 1st triggered edge after the command. Send 0x09 0x4F 0x74 0x69 0x02 0x00 0x00 0x80 0x01 0x00 0x00 0x00 sequentially again to get another triggered spectrum data.

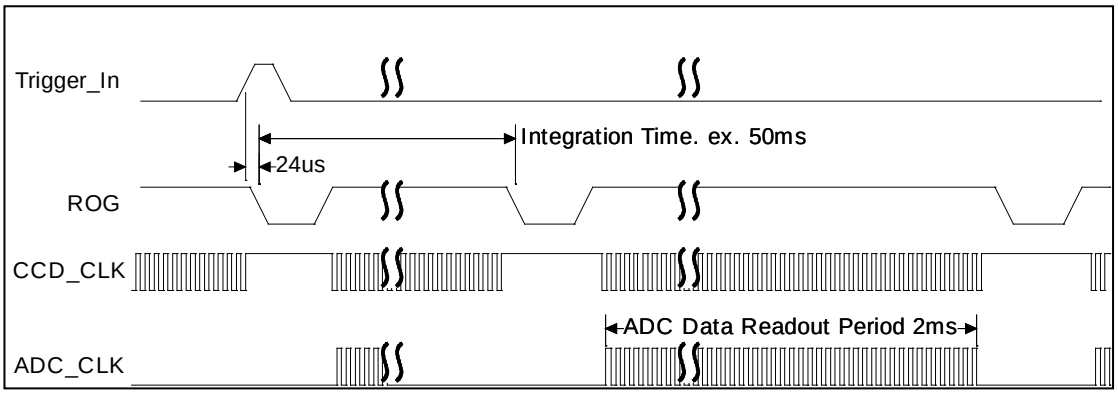
DEVCMD_SET_TRIGGER_IO can also generate the color values which are the same as command ***DEVCMD_GET_COLOR_VALUE***. Please be noted that command ***DEVCMD_SET_COLOR_INFO*** should be sent to set the related color configuration firstly before trying to acquire the color values. The bit assignment from BIT4 to BIT28 of the 'trigger_type' is the same as the bit assignment from BIT0 to BIT24 of the 'color_space' argument in command ***DEVCMD_GET_COLOR_VALUE*** respectively.

For example, Sending command sequence 0x09 0x4F 0x74 0x69 0x01 0x0F 0x00 0x00 0x00 0x00 0x00 0x00 gets 48 bytes response (4 color values, XYZ, XYZ_reference, xyz and xyz_reference, respectively) continuously after the color calculation. Each value is presented as a signed integer of 4-byte with LSB. To get the actual value, divide the value with 65536.0f. Please be noted that spectra data cannot be acquired in this example since the BIT31 of the 'trigger_type' is off.

Values marked with (**F**) have a special format for the real value. In these values, the significant 2 bytes stand for the exponent term of the real value, and the least 2 bytes stand for the factor term of the real value. For example 4 bytes 0x01 0x05 0xF6 0xFF defines the value $1.281x10^{-7}$. 0x01 and 0x05 form the value 0x0501 which is 1281 in digits. 0xF6 and 0xFF form the value 0xFFF6 which is -10. Thus the real value is $1281x10^{-10}$, which can be expressed as $1.281x10^{-7}$.

For bit 30, NormalFactor, please refer to command ***DEVCMD_INTENSITY_ABS_NRML_FAC_GET.***

Timing chart for the hardware signal is shown as below:



All related calculations (including background compensation, linearity compensation, color calculations, and so on) should be started after “ADC Data Readout Period”.

DEVCMD_SET_INTEGRATION_TIME

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SET_INTEGRATION_TIME 0x74694F09	Integration time (micro-second)	None	None

Sending 0x09 0x4F 0x69 0x74 sequentially followed by integration time (4-byte with LSB) sets the integration time of the sensor. The unit of integration time here is defined as micro-second (μs).

For example, sending the command sequence 0x09 0x4F 0x69 0x74 0x50 0xC3 0x00 0x00 (0x0000C350 = 50000) sets the integration time as 50 milliseconds.

DEVCMD_GET_INTEGRATION_TIME

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_INTEGRATION_TIME 0x5449F09	None	None	None

Sending 0x09 0x4F 0x49 0x54 sequentially gets the current integration time (4-byte with LSB) of the sensor. The unit of integration time here is defined as micro-second (μs).

DEVCMD_SET_AVERAGE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SET_AVERAGE 0x7661F09	Average	None	None

Sending 0x09 0x4F 0x61 0x76 sequentially followed by average (4-byte with LSB) sets the counts of samples for average. For example, sending command sequence 0x09 0x4F 0x61 0x76 0x04 0x00 0x00 0x00 sets 4 samples average for data acquiring. After command ***DEVCMD_SET_AVERAGE***, sending ***DEVCMD_SPECTRUM_ACQUIRE*** command gets the data with 4 samples average. That means, it should wait for about 40 ms to get the data after sending command ***DEVCMD_SPECTRUM_ACQUIRE*** as integration time = 10ms.

DEVCMD_GET_AVERAGE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_AVERAGE 0x5641F09	None	None	None

Sending 0x09 0x4F 0x41 0x56 sequentially gets the current counts of samples (4-byte with LSB) for the average.

DEVCMD_SET_RUNNING_AVERAGE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SET_RUNNING_AVERAGE 0x76724F09	Average	None	None

Sending 0x09 0x4F 0x72 0x76 sequentially followed by average (4-byte with LSB) sets the counts of samples for average. For example, sending command sequence 0x09 0x4F 0x61 0x76 0x04 0x00 0x00 0x00 sets 4 samples average for data acquiring. After command ***DEVCMD_SET_RUNNING_AVERAGE***, sending ***DEVCMD_SPECTRUM_ACQUIRE*** command gets the data with 4 samples average. Comparing with the ***DEVCMD_SET_AVERAGE*** command, it does not need to wait for about 40 ms to get the data after sending command ***DEVCMD_SPECTRUM_ACQUIRE*** as integration time = 10ms. The averaging data is calculated by sum of the current spectra data, the past 1st spectra data, the past 2nd spectra data, and the 3rd spectra data, which are stored in the system.

DEVCMD_GET_RUNNING_AVERAGE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_RUNNING_AVERAGE 0x56524F09	None	None	None

Sending 0x09 0x4F 0x52 0x56 sequentially gets the current counts of samples (4-byte with LSB) for the running average.

DEVCMD_SET_BOXCAR

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SET_BOXCAR 0x63624F09	boxcar	None	None

Sending 0x09 0x4F 0x62 0x63 sequentially followed by boxcar (4-byte with LSB) sets the neighborhood of samples for average. For example, sending command sequence 0x09 0x4F 0x62 0x63 0x04 0x00 0x00 0x00 sets totally 9 samples average (including the sample itself with 4 left and 4 right neighbors) for data acquiring. 4-byte LSB “0x04 0x00 0x00 0x00” represents as boxcar = 4.

Following rules should be noted,

1. With boxcar = 4 the samples to average are the sample itself with 4 left neighbors and 4 right neighbors. However, for the 1st sample in a frame, there is no neighbor in the left side. Meanwhile, there is no neighbor in the right side for the last sample in the frame. Thus, only 5 samples (4 right neighbors with itself) get into the average for the 1st sample in the frame, 6 samples (1 left neighbor and 4 right neighbors with itself) get into the average for the 2nd sample in the frame, and so on. Likewise, 5 samples (4 left neighbors with itself) get into the average for the last sample in the frame, and 6 samples (1 right neighbor and 4 left neighbors with itself) get into the average for the last 2nd sample in the frame.
2. The maximum boxcar to be set is 10. Of course, command with boxcar > 10 can be set. That means, sending command sequence 0x09 0x4F 0x62 0x63 0x10 0x00 0x00 0x00 is legal, but only 21 samples (10 left neighbors and 10 right neighbors with itself) get into the average.

DEVCMD_GET_BOXCAR

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_BOXCAR 0x43424F09	None	None	None

Sending 0x09 0x4F 0x42 0x43 sequentially gets the current boxcar of samples (4-byte with LSB) for the average. Please be noted that sending command 0x09 0x4F 0x62 0x63 0x10 0x00 0x00 0x00 to set the boxcar will get 0x10 0x00 0x00 0x00 (boxcar = 0x00000010 = 16)

with this command even the actual boxcar is 10 for computation.

DEVCMD_SET_LINEARITY

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SET_LINEARITY 0x616C4F09	On/Off	None	None

Sending 0x09 0x4F 0x6C 0x61 sequentially followed by on/off (4-byte with LSB) switches on/off the linearity compensation. For example, sending command sequence 0x09 0x4F 0x6C 0x61 0x01 0x00 0x00 0x00 turns on the linearity compensation for the measured spectrum. In other words, sending command sequence 0x09 0x4F 0x6C 0x61 0x00 0x00 0x00 0x00 turns off the linearity compensation for the measured spectrum.

DEVCMD_GET_LINEARITY

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_LINEARITY 0x414C4F09	None	None	None

Sending 0x09 0x4F 0x4C 0x41 sequentially checks the on/off status for the linearity compensation. "on/off" value (4-byte with LSB) will be responded after the command.

DEVCMD_SET_INTENSITY

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SET_INTENSITY 0x69694F09	Mode 0: off 1: Absolute 2: Contrast	None	None

Sending 0x09 0x4F 0x69 0x69 sequentially followed by mode (4-byte with LSB) switches on or switches off the intensity compensation. 2 modes (contrast mode and absolute mode) of the intensity calibration can be selected. For example, sending command sequence 0x09 0x4F 0x69 0x69 0x01 0x00 0x00 0x00 turns on the absolute intensity compensation for the measured spectrum, and sending command sequence 0x09 0x4F 0x69 0x69 0x02 0x00 0x00 0x00 turns on the contrast intensity compensation for the measured spectrum. In other words, sending command sequence 0x09 0x4F 0x69 0x69 0x00 0x00 0x00 0x00 turns off the intensity

compensation for the measured spectrum.

DEVCMD_GET_INTENSITY

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_INTENSITY 0x49494F09	None	None	None

Sending 0x09 0x4F 0x49 0x49 sequentially checks the mode status for the intensity compensation. "mode" value (4-byte with LSB) will be responded after the command.

DEVCMD_SET_STRAYLIGHT

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SET_STRAYLIGHT 0x6C734F09	On/Off	None	None

Sending 0x09 0x4F 0x73 0x6C sequentially followed by on/off (4-byte with LSB) switches on/off the stray light compensation. For example, sending command sequence 0x09 0x4F 0x73 0x6C 0x01 0x00 0x00 0x00 turns on the stray light compensation for the measured spectrum. In other words, sending command sequence 0x09 0x4F 0x73 0x6C 0x00 0x00 0x00 0x00 turns off the stray light compensation for the measured spectrum.

DEVCMD_GET_STRAYLIGHT

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_STRAYLIGHT 0x4C534F09	None	None	None

Sending 0x09 0x4F 0x53 0x4C sequentially checks the on/off status for the stray light compensation. "on/off" value (4-byte with LSB) will be responded after the command.

DEVCMD_SET_SPECTRUM_RANGE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SET_SPECTRUM_RANGE 0x72734F09	Type	Number	Position[number]

Sending 0x09 0x4F 0x73 0x72 sequentially sets the desired data that acquired by commands ***DEVCMD_SPECTRUM_ACQUIRE***, ***DEVCMD_SPECTRUM_ONESHOT***, and ***DEVCMD_SPECTRUM_CONTINUE***. 3 arguments (each one gets 4-byte with LSB) can be set with this command:

type

There are 2 data types according to the position of the spectrum. One is the spectrum vs. pixel and the other is the spectrum vs. wavelength. "**type**=0" defines the data is the spectrum vs. pixel and "**type**>=1" defines the data is spectrum vs. wavelength. With **type**>=1, **type** becomes the resolution for the wavelength, and the unit for this resolution is 0.1 nanometer (nm). That means **type**=10 set the resolution of wavelength mode to 1nm.

number (1~2095 or 0xFFFFFFFF)

This argument defines how many values will be acquired. This value should be constrained within **1~2095**, otherwise the result will be unexpected except 0xFFFFFFFF. 0xFFFFFFFF is used to acquire all data. The total data number can be got by command ***DEVCMD_FRAMESIZE_ACQUIRE*** for pixel mode or calculated from the wave length range of the device in wavelength mode. For example in wavelength mode, the resolution is set as 10nm (set 100 in "type" argument) and the wavelength range of the device is 300nm~1000nm. Then, the total data number for each data acquire is (1000-300)/10+1=71.
Argument "**position[number]**" should be omitted if the number=0xFFFFFFFF.

position[number]

This argument defines the position of the data for the data acquiring. These values should be **constrained by the value of frame size in pixel mode or by the wavelength range of the device in wavelength mode**; otherwise the result will be unexpected. Count of **position** should be exactly the same as **number**. Otherwise the result will be unexpected. **Position** can be set as a range of pixel in pixel mode (or a range of wavelength in wavelength mode). For example, set a position as 0x00000190 (0x90 0x01 0x00 0x00 sequentially and the value is 400 in decimal) in pixel mode means only the data of pixel 400 will be acquired. It means only the data of wavelength 400nm will be acquired in wavelength mode. Set **position** as 0x012C0190 (0x90 0x01 0x2C 0x01 sequentially) in pixel mode means the data in pixel range 300 (0x012C) ~ 400 (0x0190) will be acquired. Likewise, Set **position** as 0x012C0190 (0x90 0x01 0x2C 0x01 sequentially) in wavelength mode means the data in wavelength range 300nm~400nm with some resolution (set in **type**) will be acquired. Of course, the data size for acquiring is depended on the resolution.

Some examples are listed below:
Ex1. (pass)

Sending “0x09 0x4F 0x73 0x72” “0x00 0x00 0x00 0x00” “0xFF 0xFF 0xFF 0xFF” serially results all data output with acquiring commands (***DEVCMD_SPECTRUM_ACQUIRE***, ***DEVCMD_SPECTRUM_ONESHOT***, and ***DEVCMD_SPECTRUM_CONTINUE***) in pixel mode.

Ex2. (pass)

Suppose the wavelength range of the device is 300nm~1000nm. Sending “0x09 0x4F 0x73 0x72” “0x0A 0x00 0x00 0x00” “0xFF 0xFF 0xFF 0xFF” serially results all data output with acquiring commands (***DEVCMD_SPECTRUM_ACQUIRE***, ***DEVCMD_SPECTRUM_ONESHOT***, and ***DEVCMD_SPECTRUM_CONTINUE***) in wavelength mode. The wavelength of each data is 300nm, 301nm, 302nm… 998nm, 999nm, and 1000nm since the resolution is set as 1nm (**type** = 0x0000000A). The total data count is 701.

Ex3. (fail)

Sending “0x09 0x4F 0x73 0x72” “0x00 0x00 0x00 0x00” “0xFF 0xFF 0xFF 0xFF” “0x12 0x34 0x56 0x78” might get unexpected results.

Ex4. (pass)

Sending “0x09 0x4F 0x73 0x72” “0x00 0x00 0x00 0x00” “0x05 0x00 0x00 0x00” “0x10 0x00 0x00 0x00” “0x11 0x00 0x00 0x00” “0x20 0x00 0x00 0x00” “0x22 0x00 0x00 0x00” “0x39 0x00 0x00 0x00” serially results only 5 data of pixel 16 (0x00000010), 17 (0x00000011), 32 (0x00000020), 34 (0x00000022), and 57 (0x00000039) output with acquiring commands (***DEVCMD_SPECTRUM_ACQUIRE***, ***DEVCMD_SPECTRUM_ONESHOT***, and ***DEVCMD_SPECTRUM_CONTINUE***).

Ex5. (pass)

Sending “0x09 0x4F 0x73 0x72” “0x00 0x00 0x00 0x00” “0x02 0x00 0x00 0x00” “0x10 0x00 0x11 0x00” “0x20 0x00 0x39 0x00” serially results totally 28 (2+26) data of pixel 16 (0x00000010) ~ 17 (0x00000011) and pixel 32 (0x00000020) ~ 57 (0x00000039) output with acquiring commands (***DEVCMD_SPECTRUM_ACQUIRE***, ***DEVCMD_SPECTRUM_ONESHOT***, and ***DEVCMD_SPECTRUM_CONTINUE***).

Ex6. (pass)

With wavelength range of the device is 300nm~1000nm, sending “0x09 0x4F 0x73 0x72” “0x01 0x00 0x00 0x00” “0x02 0x00 0x00 0x00” “0x2C 0x01 0x36 0x01” “0x90 0x01 0x95 0x01” serially results totally 152 (101+51) data of wavelength 300nm (0x0000012C) ~ 310nm (0x00000136) and wavelength 400nm (0x00000190) ~ 405nm (0x00000195) output with acquiring commands (***DEVCMD_SPECTRUM_ACQUIRE***, ***DEVCMD_SPECTRUM_ONESHOT***, and ***DEVCMD_SPECTRUM_CONTINUE***).

Ex7. (shortage fail)

Sending “0x09 0x4F 0x73 0x72” “0x00 0x00 0x00 0x00” “0x05 0x00 0x00 0x00” “0x10 0x00 0x00 0x00” “0x11 0x00 0x00 0x00” “0x20 0x00 0x00 0x00” “0x22 0x00 0x00 0x00” might get unexpected results.

Ex8. (overflow fail)

Sending “0x09 0x4F 0x73 0x72” “0x00 0x00 0x00 0x00” “0x05 0x00 0x00 0x00” “0x10 0x00 0x00 0x00” “0x11 0x00 0x00 0x00” “0x20 0x00 0x00 0x00” “0x22 0x00 0x00 0x00” “0x22 0x00 0x00 0x00” “0x22 0x00 0x00 0x00” might get unexpected results.

DEVCMD_GET_SPECTRUM_RANGE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_SPECTRUM_RANGE 0x52534F09	None	None	None

Sending 0x09 0x4F 0x53 0x52 sequentially gets all 3 arguments (4-byte each with LSB) of the command ***DEVCMD_GET_SPECTRUM_RANGE***.

DEVCMD_SET_GROUP_AVERAGE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SET_GROUP_AVERAGE 0x76674F09	On/Off	None	None

Sending 0x09 0x4F 0x67 0x76 followed by an argument on_off sequentially sets on/off group average function according to the command ***DEVCMD_SET_SPECTRUM_RANGE***. Following is the detail description:

Command ***DEVCMD_SET_SPECTRUM_RANGE*** can set the desired acquiring data group by group in the argument **position**. For example, set **position** as “0x012C0190” is the pixel 300 ~ pixel 400 in pixel mode or 300nm ~ 400nm in wavelength mode. This called one group of the spectrum range. Sending acquiring commands (***DEVCMD_SPECTRUM_ACQUIRE***, ***DEVCMD_SPECTRUM_ONESHOT***, or ***DEVCMD_SPECTRUM_CONTINUE***) gets the spectra data (should be 101 data in pixel mode) of the group. However, with command ***DEVCMD_SET_GROUP_AVERAGE*** to set this function ON, the output by the acquiring command (***DEVCMD_SPECTRUM_ACQUIRE***, ***DEVCMD_SPECTRUM_ONESHOT***, or ***DEVCMD_SPECTRUM_CONTINUE***) is just one data which is the average of the group.

DEVCMD_GET_GROUP_AVERAGE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_GROUP_AVERAGE 0x56474F09	None	None	None

Sending 0x09 0x4F 0x47 0x56 gets the on/off status of the command ***DEVCMD_SET_GROUP_AVERAGE***.

DEVCMD_SET_BACKGROUND

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)
DEVCMD_SET_BACKGROUND 0x67624F09	Mode: 0: off 1: default background 2: user acquired background	None

Sending 0x09 0x4F 0x62 0x67 sequentially with one argument (4-byte with LSB) sets the background compensation model. There are 2 models of the background compensation. “USER” mode (0x02 0x00 0x00 0x00) acquires the signal immediately and use these signal data as the background compensation values. “ON” mode (0x01 0x00 0x00 0x00) use the default background compensation values stored in the ROM. “OFF” mode (0x00 0x00 0x00 0x00) turns off the background compensation. Check background values with the command ***DEVCMD_GET_BACKGROUND_VALUE***.

DEVCMD_GET_BACKGROUND

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_BACKGROUND 0x47424F09	None	None	None

Sending 0x09 0x4F 0x42 0x47 sequentially checks which mode of background compensation which is being used currently. Argument “mode” (4 bytes LSB) of command ***DEVCMD_SET_BACKGROUND*** will be output immediately

DEVCMD_GET_BACKGROUND_VALUE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_BACKGROUND_VALUE 0x56424F09	None	None	None

Sending 0x09 0x4F 0x42 0x56 sequentially checks the values of background compensation used currently. The background compensation values will be output after this command. Size of the background compensation values is equal to the frame size of the command

DEVCMD_FRAMESIZE_ACQUIRE. Each value contains 2 bytes LSB.

DEVCMD_SET_STD_WHITE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)
DEVCMD_SET_STD_WHITE 0x77734F09	Mode: 0: off 1: default standard white reflect. 2: user defined std_white reflect.	None

This command is used for color calculation. Sending 0x09 0x4F 0x73 0x77 sequentially with one argument (4-byte with LSB) sets the standard white reflectance mode. There are 2 mode of the standard white reflectance. “USER” mode (0x02 0x00 0x00 0x00) should be followed by the user defined standard white reflectance. “ON” mode (0x01 0x00 0x00 0x00) use the default standard white reflectance values stored in the ROM. “OFF” mode (0x00 0x00 0x00 0x00) turns off the standard white reflectance. Check the standard white reflectance values with the command ***DEVCMD_GET_STD_WHITE_VALUE***.

Please be noted that a formatted header should be send before putting the user defined standard white reflectance in mode 2 (‘USER’ mode). Following is the table of the header.

Header Code (Fixed 4bytes)	Date (2 bytes)	Wave Length start (2 bytes)	Wave Length end (2 bytes)	Wave Length unit (2 bytes)	Reserved (4 bytes)
0x12345678	User defined format	Eg. 380.0nm 0x0ED8	Eg. 780.0nm 0x1E78	Eg. 5.0nm 0x0032	-

The example in the header table shows that the wavelength of the standard whit reflectance will be from 380 (3800/10) nm to 780 (7800/10) nm with interval 5 (50/10) nm. With the example shown in the table, 81 user-defined reflectance values should be input since there are 81 values from 380nm~780nm with interval 5nm (start and end points are included).

DEVCMD_GET_ STD_WHITE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_ STD_WHITE 0x57534F09	None	None	None

Sending 0x09 0x4F 0x53 0x57 sequentially checks which mode of standard white reflectance which is being used currently. Argument “mode” (4 bytes LSB) of command ***DEVCMD_SET_STD_WHITE*** will be output immediately

DEVCMD_GET_ STD_WHITE _VALUE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_STD_WHITE_VALUE 0x56534F09	None	None	None

Sending 0x09 0x4F 0x53 0x56 sequentially checks the values of standard white reflectance used currently. The standard white reflectance values will be output after this command. Size of the standard white reflectance values is equal to the frame size of the command ***DEVCMD_FRAMESIZE_ACQUIRE***. Each value contains 4 bytes LSB with the format as “unsigned 32 bits”. And the unit of the value is 0.00001. For example, one value got from this command is 0x000184B9. That means the reflectance is 99513(0x000184B9) X 0.00001 = 0.99513 = 99.513%.

DEVCMD_WAVELENGTH_ACQUIRE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_WAVELENGTH_ACQUIRE 0x51574F09	None	None	None

Sending 0x09 0x4F 0x57 0x51 sequentially gets the wavelength value (4-byte each with LSB) of each pixel. Following the command ***DEVCMD_FRAMESIZE_ACQUIRE***, the total pixel count can be received. Each pixel represents its own wavelength. Wavelength value in this command is defined as a 4-byte (LSB) unsigned integer. Divide this unsigned integer with 65536.0f to get the actual wavelength value (float type).

DEVCMD_WAVELENGTH_PIECE_ACQUIRE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)
DEVCMD_WAVELENGTH_PIECE_ACQUIRE 0x50574F09	Pixel range	None

This command gets the piecewise wavelength values. Sending 0x09 0x4F 0x57 0x50 sequentially followed by argument pixel_range gets the wavelength value (4-byte each with LSB) of each pixel in pixel_range. For example, sending 0x09 0x4F 0x57 0x50 0x00 0x00 0x0A 0x00 sequentially acquire the wavelength of the pixel 0 (0x0000) ~ pixel 10 (0x000A). Wavelength value in this command is defined as a 4-byte (LSB) signed integer. Divide this signed integer with 65536.0f to get the actual wavelength value (float type).

DEVCMD_FRAMESIZE_ACQUIRE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_FRAMESIZE_ACQUIRE 0x4F464F09	None	None	None

Depending on the specification and the performances, the frame size of the spectrometers may be different. Sending 0x09 0x4F 0x46 0x4F sequentially gets the actual frame size (4-byte with LSB).

DEVCMD_SET_AUTO_INTEGRATION_TIME

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)
DEVCMD_SET_AUTO_INTEGRATION_TIME 0x74614F09	None	None

To start the color measurement, sending command sequence ‘0x09’ ‘0x4F’ ‘0x61’ ‘0x74’ can find the integration time for the measurement automatically. Status response (4-byte with LSB) will be received after finding integration time done. A non-zero 4-byte value (which means the integration time) shows the result of the integration time. This value is limited in the range of 1000 (0x000003E8) ~ 1000000 (0x000F4240) with the unit micro-second (us).

Please be noted that since the value is limited in 1000~1000000 us, following situations should be aware:

1. The returned value is 1000 us. This may be caused by the too strong power of light source, and the CCD signal cannot be adjusted by tuning the integration time (1000us is the minimum integration time in current system).
2. On the other hand, the returned value of 1000000us may be caused by too low power of light source. The CCD signal cannot be adjusted by tuning the integration time (1000000us is the maximum integration time in current system).

***DEVCMD_SET_COLOR_REFERENCE** (Option)*

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)
DEVCMD_SET_COLOR_REFERENCEE 0x72634F09	None	None

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x63’ ‘0x72’ acquires the spectrum as the reference for the color calculation. This spectrum will be saved in DRAM and can be used here and after. All the background, linearity, and intensity compensation can be applied for this reference spectrum.

***DEVCMD_GET_COLOR_REFERENCE** (Option)*

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)
DEVCMD_GET_COLOR_REFERENCEE 0x52434F09	None	None

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x43’ ‘0x52’ gets the spectrum which is acquired by command ***DEVCMD_SET_COLOR_REFERENCE***. The frame size of this spectrum can be received by the command ***DEVCMD_FRAMESIZE_ACQUIRE***. Each data of the spectrum is represented as 2 bytes LSB.

***DEVCMD_SET_COLOR_INFO** (Option)*

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SET_COLOR_INFO 0x69634F09	Irradiance 0: contrast emissive 1: contrast reflective 2: absolute emissive 3: absolute reflective	Observer 0: 2 degree 1: 10 degree	Illuminant 0: CIE A 1: CIE B 2: CIE C 3: CIE D50 4: CIE D55 5: CIE D65 6: CIE D75 7: CIE E 8: CIE F1 9: CIE F2 10: CIE F3 11: CIE F4 12: CIE F5 13: CIE F6 14: CIE F7 15: CIE F8 16: CIE F9 17: CIE F10

			18: CIE F11 19: CIE F12

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x63’ ‘0x69’ sets the information for color calculation. 3 values (4-byte each with LSB), irradiance, observer, and illuminant, should be set sequentially after the command. For example, set the related information ‘irradiance = contrast emissive’, ‘observer = 2 degree’, and ‘illuminant = cie illuminant A’ by sending command sequence as ‘0x09’ ‘0x4F’ ‘0x63’ ‘0x69’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x00’.
Please be noted that sending command ***DEVCMD_GET_COLOR_VALUE*** without setting color information in advanced will get an unexpected result.

DEVCMD_GET_COLOR_INFO *(Option)*

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_COLOR_INFO 0x49434F09	None	None	None

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x43’ ‘0x49’ gets the information set by command ***DEVCMD_SET_COLOR_INFO***.

DEVCMD_GET_COLOR_VALUE *(Option)*

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)
DEVCMD_GET_COLOR_VALUE 0x56434F09	Color space BIT0: XYZ (12 bytes) (**F**) BIT1: reserved BIT2: xyz (12 bytes) BIT3: reserved BIT4: CIE Lab (12 bytes) BIT5: reserved BIT6: reserved BIT7: reserved BIT8: reserved BIT9: 1960UCS (12 bytes) BIT10:1960ucs (12 bytes) BIT11: 1976UCS (12 bytes) BIT12: 1976ucs (12 bytes)	None

	BIT13: CCT (4 bytes) BIT14: Dominant Wavelength (4 bytes) BIT15: Purity (4 bytes) BIT16: CIE Whiteness (4 bytes) BIT17: CIE Tin (4 bytes) BIT18: Color Rendering Index (17x4 bytes) BIT19: Color Quality Scale (16x4 bytes) BIT20: Luminance (4 bytes) (**F**) BIT21: Radiant Power (4 bytes) (**F**) BIT22: Central Wavelength (4 bytes) BIT23: reserved BIT24: Lambda Peak (4 bytes) BIT25: PPFD(4 bytes) BIT26~28: reserved BIT29: Set Auto Integration Time (4 bytes returned) BIT30: Normal Factor (4 bytes) (**F**) BIT31: raw intensity	

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x43’ ‘0x56’ gets the calculated color values for each color space. Some calculated color information such as CCT or Dominant Wavelength can also be received by this command. The required color value or color information should be set by ‘color_space’ argument (4-byte with LSB). Please be noted that the response value with this command is depended on the value of ‘color_space’. For example, Sending command sequence ‘0x09’ ‘0x4F’ ‘0x43’ ‘0x56’ ‘0x0F’ ‘0x00’ ‘0x00’ ‘0x00’ gets 48 bytes response (4 color values, XYZ (BIT0), XYZ_reference (BIT1), xyz (BIT2) and xyz_reference (BIT3), respectively) after the color calculation. Each value is presented as a signed integer of 4-byte with LSB. To get the actual value, divide the value with 65536.0f.

Please be noted that values marked with (**F**) have a special format for the real value. In these values, the significant 2 bytes stand for the exponent term of the real value, and the least 2 bytes stand the factor term of the real value. For example 4 bytes 0x01 0x05 0xF6 0xFF defines the value $1.281x10^{-7}$. 0x01 and 0x05 form the value 0x0501 which is 1281 in digits. 0xF6 and 0xFF form the value 0xFFF6 which is -10. Thus the real value is $1281x10^{-10}$, which can be expressed as $1.281x10^{-7}$.

Auto Integration Time Set function can be performed before getting color value with setting the bit 29. The auto-adjusted integration time, 4-byte LSB value, will be returned after the command being finished. The data format of the returned integration is the same as command ***DEVCMD_SET_AUTO_INTEGRATION_TIME***.

For bit 30, NormalFactor, please refer to command ***DEVCMD_INTENSITY_ABS_NRML_FAC_GET.***

DEVCMD_SPECTRUM_RATIO_ACQUIRE *(Option)*

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SPECTRUM_RATIO_ACQUIRE 0x51524F09	None	None	None

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x52’ ‘0x51’ gets the calculated ratio of current spectrum over the reference got by command ***DEVCMD_SET_COLOR_REFERENCE***. Total data size of the ratio is the same as the value of frame size got by command ***DEVCMD_FRAMESIZE_ACQUIRE***. Each ratio value is presented as a signed integer of 4-byte with LSB. To get the actual value, divide the value with 65536.0f.

DEVCMD_TRANSMISSION_ACQUIRE *(Option)*

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_TRANSMISSION_ACQUIRE 0x51544F09	None	None	None

Do the following steps to get the transmission of the spectrum:

1. Get the reference spectrum with command ***DEVCMD_SET_COLOR_REFERENCE*** firstly.
2. Put the sample between the illuminant and the spectrum meter.
3. Send command sequence ‘0x09’ ‘0x4F’ ‘0x54’ ‘0x51’ to get the calculated transmission of the sample.

Total data size of the transmission is the same as the value of frame size got by command ***DEVCMD_FRAMESIZE_ACQUIRE***. Each transmission value is presented as a signed integer of 4-byte with LSB. To get the actual data, divide the value with 65536.0f.

DEVCMD_REFLECTION_ACQUIRE *(Option)*

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_REFLECTION_ACQUIRE 0x51464F09	None	None	None

Do the following steps to get the reflection of the spectrum:

1. Get the spectrum for the reference white with command ***DEVCMD_SET_COLOR_REFERENCE*** firstly.
2. Measure the object by sending command sequence '0x09' '0x4F' '0x54' '0x51' to get the calculated reflection of the sample.

Total data size of the reflection is the same as the value of frame size got by command ***DEVCMD_FRAMESIZE_ACQUIRE***. Each reflection value is presented as a signed integer of 4-byte with LSB. To get the actual data, divide the value with 65536.0f.

DEVCMD_SET_BAUDRATE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_SET_BAUDRATE 0x64624F09	Baud rate	None	None

The baud rate of the communication port can be changed by sending command sequence '0x09' '0x4F' '0x62' '0x64' with one argument baudrate.

Please be noted that the baud rate will be changed immediately after sending this command. The communication after this command should be changed to the new baud rate. Currently, the legal range of the baud rate is 4800, 9600, 14400, 19200, 38400, 57600, 115200, and 230400.

This baud rate setting will not be kept after power off the device.

DEVCMD_GET_BAUDRATE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_BAUDRATE 0x44424F09	None	None	None

Sending command sequence '0x09' '0x4F' '0x42' '0x44' gets the current baud rate (4bytes LSB) on board.

DEVCMD_INTENSITY_CALIBRATE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_INTENSITY_CALIBRATE 0x63694F09	Date	None	None

Sending command sequence '0x09' '0x4F' '0x69' '0x63' with argument Date is going to do the intensity calibration by the device itself. The 'Date' value will be stored in NVRAM for mark which can be received by command ***DEVCMD_INTENSITY_CAL_DATE_GET****. Following description states detail procedures for the calibration.*

1. Acquire spectrum data.
2. Get the reference data from table
3. Calculate the gain for the absolute intensity.
4. Store gain values to NVRAM.

Please be noted that this action will modify the intensity calibration data in NVRAM. Do not interrupt the power during the whole process.

DEVCMD_INTENSITY_CAL_DATE_GET

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_INTENSITY_CALIBRATE 0x47494F09	None	None	None

Sending command sequence '0x09' '0x4F' '0x49' '0x47' to get the 'Date' value which is set by command ***DEVCMD_INTENSITY_CALIBRATE***.

DEVCMD_INTENSITY_STDTABLE_SET

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (N bytes LSB)	arg3 (N bytes LSB)
DEVCMD_INTENSITY_STDTABLE_SET 0x6C694F09	Length	Table[0][Length]	Table[1][Length]

Sending command sequence '0x09' '0x4F' '0x69' '0x6C' with arguments length and tables sets the reference table for absolute intensity calibration. The value of the Length should be less than 2095. It will cause fatal errors if Length is more than 2095.
Table[0][Length] shows the wavelength for the reference, and table Table[1][Length] shows counts of the intensity with respect to the wavelength. The unit of the wavelength is 0.1nanometer. That means a value 4052 of wavelength in Table[0][Length] indicate the wavelength as 405.2 nanometer. Byte length of each table (Table[0][Length] or Table[1][Length]) should be 8xLength bytes. Thus, total byte size in this command should be equal to 4+4+4xLength+8xLength.

DEVCMD_INTENSITY_STDTABLE_GET

Command Code	arg1	arg2	arg3

(4bytes LSB)	(4bytes LSB)	(4bytes LSB)	(4bytes LSB)
DEVCMD_INTENSITY_STDTABLE_GET 0x4C494F09	None	None	None

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x49’ ‘0x4C’ acquires the reference table for the intensity calibration. The total byte length of the response will be 4+4xLength+8xLength. The first 4 bytes show the value of ‘Length’ actually.

DEVCMD_LAMBDA_TO_PIX

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_LAMBDA_TO_PIX 0x504C4F09	Wavelength Scaled with 65536	None	None

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x4C’ ‘0x50’ with wavelength (4 bytes LSB) gets the pixel number which is closest to the wavelength.
For example, the wavelength of pixel 200 is 504.13nm and pixel 201 is 504.98nm. Sending command sequence ‘0x09’ ‘0x4F’ ‘0x4C’ ‘0x50’ ‘0x00’ ‘0x80’ ‘0xF8’ ‘0x01’ gets the response ‘0x00‘ ‘0x00’ ‘0x00’ ‘0xC8’ which means pixel 200. Wavelength ‘0x00’ ‘0x80’ ‘0xF8’ ‘0x01’ equals to 0x01F88000 which is 33062912 in decimal. Divide 33062912 with 65536 gets the actual wavelength is 504.5nm.
Thus, the meaning of the example above is to get the pixel number which is closest to 504.5nm. And the answer is pixel 200 since the distance between pixel 200 and 504.5nm is 0.37nm but the distance between pixel 201 and 504.5 is 0.48nm.

DEVCMD_GET_SERIAL_NUMBER

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_SERIAL_NUMBER 0x4E534F09	None	None	None

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x53’ ‘0x4E’ gets the 16 bytes serial number of the device.

DEVCMD_GET_MODEL_NAME

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_MODEL_NAME	None	None	None

0x4E4D4F09			

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x4D’ ‘0x4E’ gets the 16 bytes model name of the device.

DEVCMD_GET_SLIT_TYPE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_GET_SLIT_TYPE 0x54534F09	None	None	None

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x53’ ‘0x54’ gets the 16 bytes slit type of the device.

DEVCMD_GET_WAVELENGTH_START_END

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)
DEVCMD_ GET_WAVELENGTH_START_END 0x45574F09	None	None

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x57’ ‘0x45’ gets the start wavelength value (4 bytes LSB) and the end wavelength value (4 bytes LSB) of the device.
For example, the specification of the device is to measure the spectrum of the wavelength range 380nm~780nm. Sending command sequence ‘0x09’ ‘0x4F’ ‘0x57’ ‘0x45’ gets the 8 bytes response ‘0x7C’ ‘0x01’ ‘0x00’ ‘0x00’ ‘0x0C’ ‘0x03’ ‘0x00’ ‘0x00’. The first 4 bytes stand for 380nm and the last 4 bytes stand for 780nm.

DEVCMD_READ_USER_ROM

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_ READ_USER_ROM 0x55524F09	Length	Address	None

There are 512 bytes spaces in ROM for saving user defined values. Sending this command gets the values of the ROM which are located from the assigned address. The address of the space is from 0x00000000 to 0x000001FF. And the maximum length should be less than 0x00000200. Any mistake for the length or address input gets no response from the device. For example, sending command sequence ‘0x09’ ‘0x4F ‘0x52’ ‘0x55’ ‘0x01’ ‘0x00’ ‘0x00’ ‘0x00’

‘0x03’ ‘0x00’ ‘0x00’ ‘0x00’ gets 1 byte response which is the value of the address 0x03 in user ROM.

Sending command sequence ‘0x09’ ‘0x4F ‘0x52’ ‘0x55’ ‘0x00’ ‘0x10’ ‘0x00’ ‘0x00’ ‘0x03’ ‘0x00’ ‘0x00’ ‘0x00’ gets no response since the length is over the limitation of 512 bytes.

DEVCMD_WRITE_USER_ROM

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3… (each byte)
DEVCMD_ WRITE_USER_ROM 0x75774F09	Length	Address	Data [length]

Sending this command writes the values to the user ROM. The address of the space is from 0x00000000 to 0x000001FF. And the maximum length should be less than 0x00000200. Any mistake for the length or address input gets some unexpected response from the device.

For example, sending command sequence ‘0x09’ ‘0x4F ‘0x77’ ‘0x75’ ‘0x01’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x03’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0xAB’ writes 1 byte ‘0xAB’ to the address 0x03 in user ROM.

Sending command sequence ‘0x09’ ‘0x4F ‘0x77’ ‘0x75’ ‘0x00’ ‘0x10’ ‘0x00’ ‘0x00’ ‘0x03’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0xAB’… writes only the first 510 bytes to the address from 0x03 to 0x1FF and ignores the rest bytes.

Sending command sequence ‘0x09’ ‘0x4F ‘0x77’ ‘0x75’ ‘0x10’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x00’ ‘0x30’ ‘0x00’ ‘0x00’ ‘0xAB’… writes no byte of data to user ROM since the address is beyond the legal range.

DEVCMD_ERASE_USER_ROM

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_ ERASE_USER_ROM 0x75654F09	None	None	None

Send this command ‘0x09’ ‘0x4F’ ‘0x65’ ‘0x75’ sets the whole (512 bytes) user ROM to 0xFF which means the user ROM was reset.

DEVCMD_POWER_SAVE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_POWER_SAVE 0xFFFF4F09	None	None	None

Sending command sequence ‘0x09’ ‘0x4F’ ‘0xFF’ ‘0xFF’ set the device to Power Save Mode. 4 bytes ‘0x20’ ‘0x4F’ ‘0x46’ ‘0x46’ will be responded after the device getting into the Power Save Mode.

DEVCMD_POWER_WAKE

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_WAKE 0x00004F09	None	None	None

Sending command sequence ‘0x09’ ‘0x4F’ ‘0x00’ ‘0x00’ wake the device up from the Power Save Mode. 4 bytes ‘0x20’ ‘0x20’ ‘0x4F’ ‘0x4E’ will be responded after this command.

DEVCMD_INTENSITY_ABS_NRML_FAC_GET

Command Code (4bytes LSB)	arg1 (4bytes LSB)	arg2 (4bytes LSB)	arg3 (4bytes LSB)
DEVCMD_WAKE 0x4E414F09	None	None	None

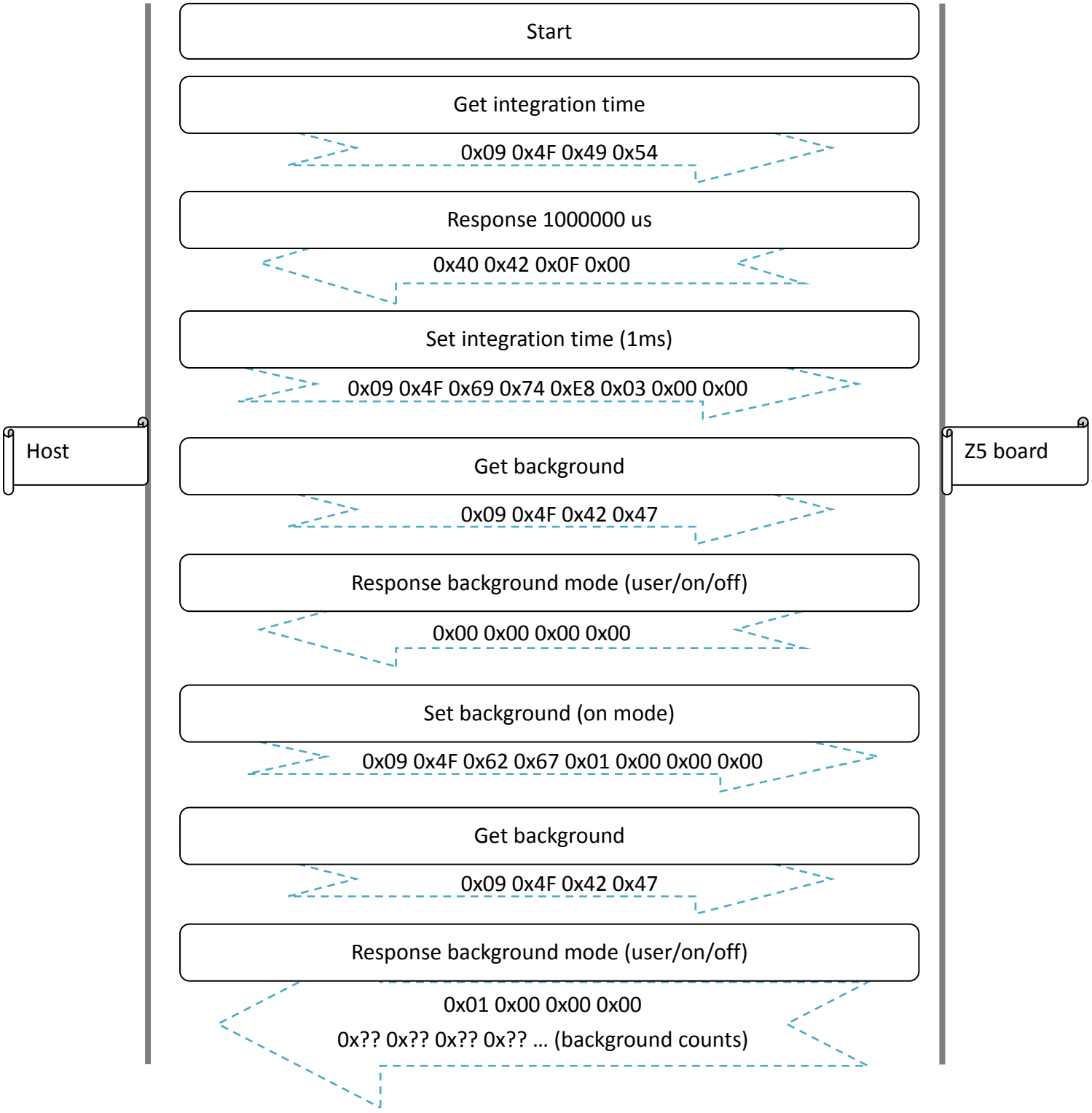
Sending command sequence ‘0x09’ ‘0x4F’ ‘0x41’ ‘0x4E’ gets the Normal Factor for absolute intensity. 4 bytes LSB value of the Normal Factor will be responded after this command. This Factor is used for compensating the raw of intensity acquire by the Spectra Acquiring Commands.

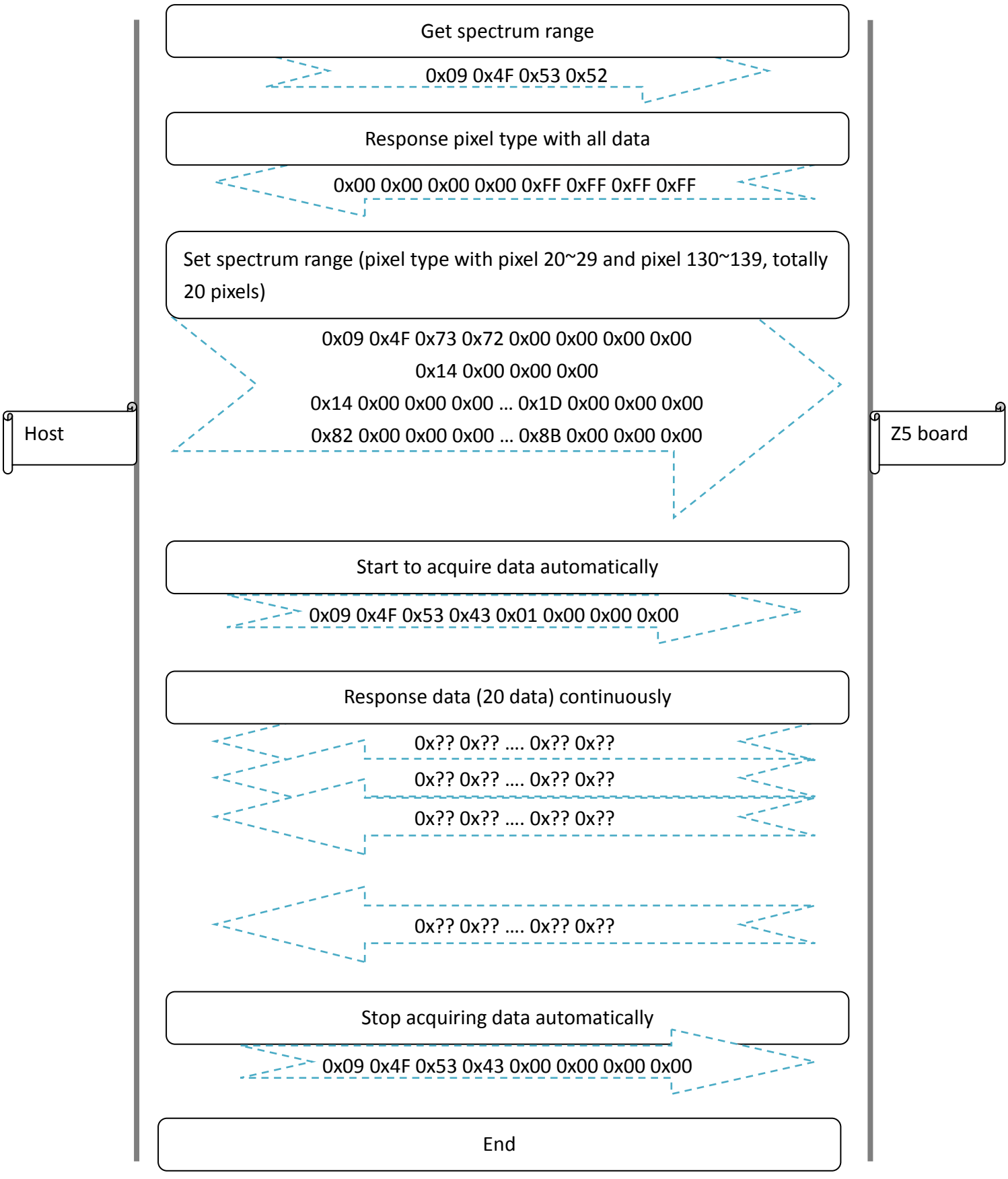
As the intensity calibration sets ON (‘mode’ value in command ***DEVCMD_SET_INTENSITY*** is 1 or 2), acquiring the spectrum gets the contrast intensity calibrated values even the ‘mode’ value setting in ***DEVCMD_SET_INTENSITY*** is 1 (absolute intensity). To obtain the actual absolute intensity calibrated values, divide all the output values with this factor getting by command ***DEVCMD_INTENSITY_ABS_NRML_FAC_GET***.

The format of the Normal Factor is the same as the color values noted as (**F**). Following is the detail example:

Values marked with (**F**) have a special format for the real value. In these values, the significant 2 bytes stand for the exponent term of the real value, and the least 2 bytes stand the factor term of the real value. For example 4 bytes 0x01 0x05 0xF6 0xFF defines the value $1.281x10^{-7}$. 0x01 and 0x05 form the value 0x0501 which is 1281 in digits. 0xF6 and 0xFF form the value 0xFFF6 which is -10. Thus the real value is $1281x10^{-10}$, which can be expressed as $1.281x10^{-7}$.

Example flow




Host
Z5 board
Get spectrum range
0x09 0x4F 0x53 0x52
Response pixel type with all data
0x00 0x00 0x00 0x00 0xFF 0xFF 0xFF 0xFF
Set spectrum range (pixel type with pixel 20~29 and pixel 130~139, totally 20 pixels)
0x09 0x4F 0x73 0x72 0x00 0x00 0x00 0x00
0x14 0x00 0x00 0x00
0x14 0x00 0x00 0x00 … 0x1D 0x00 0x00 0x00
0x82 0x00 0x00 0x00 … 0x8B 0x00 0x00 0x00
Start to acquire data automatically
0x09 0x4F 0x53 0x43 0x01 0x00 0x00 0x00
Response data (20 data) continuously
0x?? 0x?? …. 0x?? 0x??
0x?? 0x?? …. 0x?? 0x??
0x?? 0x?? …. 0x?? 0x??
0x?? 0x?? …. 0x?? 0x??
Stop acquiring data automatically
0x09 0x4F 0x53 0x43 0x00 0x00 0x00 0x00
End