Interface and

Data Format Specification

for sensors

(v3 sensor boards with Alpha sensor,

Firmware version 4.1)

Waggle Group

Waggle Sensor Array

November 2017, Version 1.0

Contents

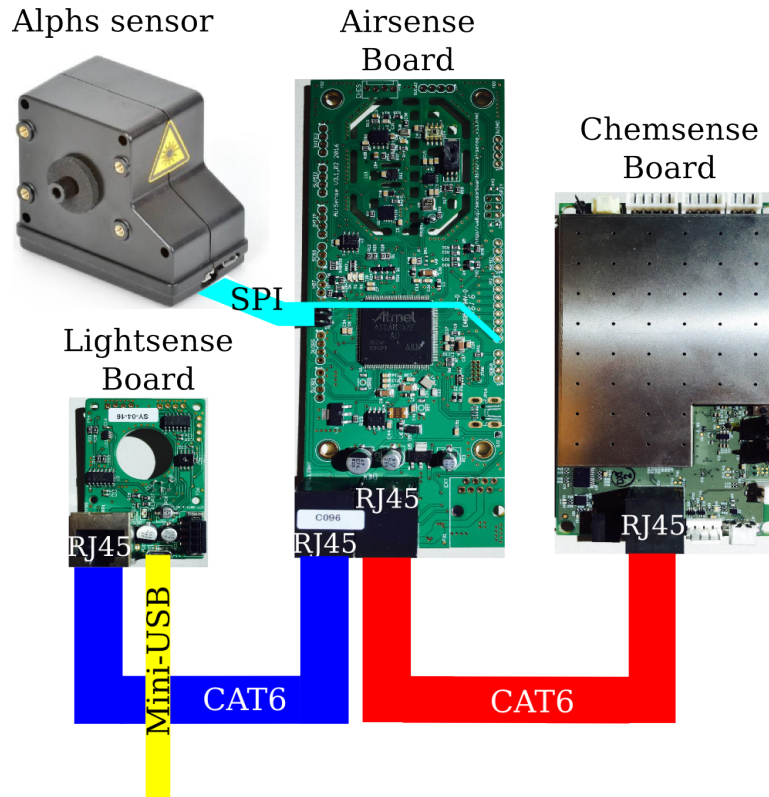# 1   Physical Connections and Interfaces

Figure 1: Connections between the sensor boards and the sensor

'v3 sensor boards with alpha sensor' means a set of sensors that are implemented on a v3.1 Metsense board, a v3.1 Lightsense board, and a Chemsense board and an independent alpha sensor.

Physical connections between sensor boards and an alpha sensor are shown in the Figure 1. A lightsense board and a chemsense board are connected to a metsense board through CAT6 cable. The metsense and the lightsense deliver data through I2C communication, and the chemsense board delivers data through serial3 communication. Alpha sensor is conncted to the metsense on SPI pins and one of GPIT pins. User requests and collects data from alpha sensor using SPI communication. All sensor data from metsense board, lightsense board, chemsense board, and alpha sensor are delivered to nodecontroller thourgh USB line attached on lightsense board using Serial communication.

2   Data Transmission

The data from the sensor boards are packetized in a transmission pacekt with the form of as they had read from the sensor. A transmission packet can be composed of several data sub-packets, each of which carries information pertaining to the parameter. The transmission packet format and the data sub-packets are described here.

2.1   Transmission Packet

A transmission packet can be separated into 6 segments. The structure of the transmission packet relies on positions of Bytes and predefined values for those Byte segments. Table 1 below illustrates how the segments are organized in a transmission packet.

Preamble	a [a]	b [b]	Data Length	Data	CRC	Postscript
1st Byte	2nd Byte	3rd Byte	4th Byte	next Bytes up to 256 Bytes	Penultimate Byte	Final Byte

[a] Packet Type | Protocol Version
[b] Last Packet Flag | Sequence

Table 1: Transmission Packet structure

A description how the segments are organized in a transmission packet is shown in table 2 below.

Field	Value	Segment	Length
Preamble	0xAA	1	1 Byte
Packet Type	0x00: request 0x01: sensor reading	2	1 Nibble
Protocol version	0x02		1 Nibble
Last Packet flag	0x01: last packet for one request 0x00: not last packet	3	1 bit
Packet sequence	Variable		7 bits
Length of data	Variable	4	1 Byte
Data	Variable	5	Variable
CRC of data	Variable	6	1 Byte
Postscript	0x55	7	1 Byte

Table 2: Transmission Packet Segments

The first segment is the start byte, or the preamble. The preamble is followed by the packet type and protocol

version, each of which are 4 bits long and are together packed into a single byte. Next, one byte field that reports the first 1 bit of last packet flag and 7 bit sequence number. Following byte reports length of the data which comes along until its immediately. The data segment is followed by a single CRC byte, and finally the packet ends with a one byte crc and postscript. Table 2 lists the packet and the static values, if any, for each of the segments.

2.2   Data Sub-packets

The data segment of the transmission packet can be further separated into many sub-packets. Two types of sub-packets are implemented, each of sub-packets are for sending request from coresense plugin to coresense firmware and visa versa.

Table 3 below shows the organization of a sub-packet requesting sensor data. The sub-pacekt starts with 4-bits call function id and 4-bits parameter length including source identifier, which is sensor id. The next bytes are parameters starting with target sensor id. Additional parameters can be attached after the sensor id. For more detail, refer sensor description file (SDF).

Table 4 below shows the organization of a sub-packet sending sensor reading. The sub-packet starts with a source identifier, which is sensor id. One bit validity field and seven bits "length of the sub-packet" field are packed together as the next byte. The length field counts the number of bytes following it which make up the sub-packet. The validity bit is set to 1 if the sensor reading is valid and set to 0 if the sensor is dead, disabled, unconnected, unresponsive or if data could not be collected from the sensor in the time window. The size of the sub-packet is restricted to 127 Bytes by the seven bits length field.

Call Function ID | Parameter Length including sensor ID	Source ID	Parameters
4 bits | 4 bits	1 Byte	up to 15 Bytes

Table 3: Transmission Packet Segments from plugin to firmware

Source ID	1-bit Validity [0: invalid, 1: valid] | 7-bits Data Length	Data
1 Byte	1 Byte	up to 128 Bytes

Table 4: Transmission Packet Segments

2.3   Data Packer CRC

To validate the data transmitted from and to the sensor board, a CRC value for the data is calculated and transmitted as part of the data packet. The Maxim 1-Wire CRC polynomial is used for calculating the CRC. On receiving the packet,

the CRC is recalculated and compared with the value transmitted as part of the packet. If the two CRC values match, the transmission is error-free. The equivalent polynomial function of the CRC is shown in Equation 1.

$$CRC = x^8 + x^5 + x^4 + 1 \tag{1}$$

Further description of the Maxim 1-Wire CRC is available in Maxim Application Note 27. Below are the Python and C implementations of the CRC calculator. The CRC implementations below take a data Byte and the previous CRC as inputs, and return the new CRC as return value.

Python Code:

```python
def calc_crc (data_Byte,CRC_Value)
    CRC_Value = ord(data_Byte) ^ CRC_Value
    for j in range(8):
    if (CRC_Value  & 0x01):
        CRC_Value  = (CRC_Value  >> 0x01) ^ 0x8C
    else:
        CRC_Value  =  CRC_Value  >> 0x01
return CRC_Value
```

C Code:

```c
unsigned char  CRC_CALC (unsigned char data, unsigned char crc)
{
        unsigned char i;
        crc ^= data;
        for (i=0x00; i < 0x08; i++)
        {
                if (crc & 0x01) { crc = (crc >> 0x01)^0x8C; }
                else { crc =  crc >> 0x01; }
        }
        return(crc);
}
```

3   Sub-packets from Coresense

As shortly explained in document section 2.2, data sub-packets from coresense are generated depending on its data reading from each sensor if valid. The first byte of the sub-packet from coresense is sensor ID for each parameter, and the second byte means validity of the packet and length of the sensor data as shown in Table **??**. Detail of sub-packet and sensor data will be explined in this section.

3.1   Parameters

The sensor boards output a set of parameters which are identified by a unique ID. Each parameter has a set of values associated with it which are encoded in an appropriate data format. The table below lists the various parameters produced by the sensor boards, the unique source ID used to identify them, the values produced by them.

Table 5: Data sub-packet structure (each row is a "chunk")

Parameter	Source ID	Values	Data Length
Firmware version	0xFF	Version (HW/SW)	2 bytes
		Build time	4 bytes
		Build git	2 bytes
Metsense board			
Metsense/Lightsense MAC address	0x00	MAC Address	6 bytes
TMP112	0x01	Temperature	2 bytes
HTU21D	0x02	Temperature	2 bytes
		Relative humidity	2 bytes
HIH4030	0x03	Relative humidity	2 bytes
BMP180	0x04	Temperature	2 bytes
		Pressure	3 bytes
PR103J2	0x05	Temperature	2 bytes
TSL250RD	0x06	Visible Light	2 bytes
MMA8452Q	0x07	Acceleration in X	2 bytes
		Acceleration in Y	2 bytes
		Acceleration in Z	2 bytes
SPV1840LR5H-B	0x08	RMS Sound Level	128 bytes
TSYS01	0x09	Temperature	2 bytes
Continued on next page			

Table5 – continued from previous page

Parameter	Source ID	Values	Data Length
Lightsense board			
HMC5883L	0x0A	Magnetic Field in Z	2 bytes
		Magnetic Field in Y	2 bytes
		Magnetic Field in Z	2 bytes
HIH6130	0x0B	Temperature	2 bytes
		Relative humidity	2 bytes
APDS-9006-020	0x0C	Ambient light intensity	3 bytes
TSL260RD	0x0D	IR intensity	3 bytes
TSL250RD	0x0E	Visible light intensity	3 bytes
MLX75305	0x0F	Light	3 bytes
ML8511	0x10	UV intensity	3 bytes
TMP421	0x13	Temperature	2 bytes
Chemsense board			
Chemsense configuration	0x16	Chemsense FW config	1514 bytes
Chemsense reading	0x2A	Raw reading	Varies
Alpha Sensor			
Histogram	0x28	Bin count	32 bytes
		Average time	4 bytes
		Sample flow rate	4 bytes
		Temp/Pressure(alter)	4 bytes
		Sampling period	4 bytes
		Sum of the counts	2 bytes
		PM 1	4 bytes
		PM 2.5	4 bytes
		PM 10	4 bytes
Serial	0x29	Serial number	20 bytes
Firmware	0x30	Firmware version	2 bytes
Configuration	0x31	Bin Boundaries	32 bytes
		Bin particle volumes	64 bytes
		Bin particle densities	64 bytes
		Bin sample volume weightings	64 bytes
		Gain scaling coefficient	4 bytes
Continued on next page			

Table5 – continued from previous page

Parameter	Source ID	Values	Data Length
continued	continued	Sample flow rate	4 bytes
		Laser DAC	1 byte
		Fan DAC	1 byte
		Conversion factor	1 byte
		Spare bytes	21 bytes

3.2   Data packets

The context of each parameter, its utility and the arrangement of its values is described below. In all the tables below, the validity bit is set to 1, which means the data is valid. The parameter described below are aggregated based on the sensor-board they are situated on - Metsense, Lightsense and Chemsense.

3.2.1   Firmware Version

This is a 8 bytes version information that identifies hardware version, software version, and build information of the waggle node. The build time and the build git are included to varify the effectiveness of the software. Firmware version is bit masked and encoded through format 1, and build git is encoded through format 1.

0xFD	0x88	Firmware version	Build time	Build git
Byte[0]	Byte[1]	Bytes[2 − 3]	Bytes[4 − 7]	Bytes[8 − 9]

Table 6: Sub-packet of Firmware version

3 bit major HW ver. | 3 bit minor HW ver. | 2 bit major SW ver.	Byte[2]
2 bit major SW ver. | minor SW ver. × 10 + sub SW ver.	Byte[3]

Table 7: Firmware version

3.2.2   Metsense

● **Metsense/Lightsense MAC address:**   This is a 6-byte ID that uniquely identifies each Airsense board. This MAC address is also applied to each Lightsense board which has the same board number. The ID is provided by a DS2401 1-Wire DSN chip. The 1-byte family ID and CRC provided by the DSN chip are omitted, and the rest 6 bytes are used as the Unique ID.

0x00	0x86	MAC address
Byte[0]	Byte[1]	Bytes[2 − 7]

Table 8: Sub-packet of met/lightsense board MAC address

● **TMP112, HIH4030, PR103J2, TSL250RD, TSYS01:**   TMP112, PR103J2, and TSYS01 are temperature sensors, HIH4030 is a humidity sensor, and TSL250RD is a light sensor. The coresense firmware collectes data from TMP112 through I2C, and from other sensors using analog read. All the reading values from the sensors are packetized as the raw value as they are collected. The raw reading will be converted relatively to temperature in centigrade, humidity in %RH, and light in lux.

Sensor ID (0x01, 0x03, 0x05, 0x06, 0x09)	0x82	Raw sensor reading
Byte[0]	Byte[1]	Bytes[2 − 3]

Table 9: Sub-packet for the sensor listed above

● **HTU21D:** HTU21D is a temperature and relative humidity sensor. The coresense firmware collects data from HTU21D through I2C and the readings are packetized as the raw value as they are collected. The raw readings will be converted to temperature in centigrate, humidity in %RH.

0x02	0x84	Raw temperature reading	Raw humidity reading
Byte[0]	Byte[1]	Bytes[2 − 3]	Bytes[4 − 5]

Table 10: Sub-packet of a temperature and relative humidity sensor, HTU21D

● **BMP180:** BMP180 is a temperature and barometric pressure sensor. The coresense firmware collects data from BMP180 through I2C and the readings are packetized as the raw value as they are collected. The raw readings will be converted to temperature in centigrade and barometric pressure in hPa.

0x04	0x84	Raw temperature reading	Raw pressure reading
Byte[0]	Byte[1]	Bytes[2 − 3]	Bytes[4 − 5]

Table 11: Sub-packet of a temperature and barometric pressure sensor, BMP180

● **MMA8452Q:** MMA8452Q is a three-axis accelerometer. The accelerations in three orthogonal directions, x, y and z, as a multiple of acceleration due to gravity (g) are obtained from the sensor. The coresense firmware collects data from this sensor through I2C and the readings are packetized as the raw value as they are collected. The raw reading will be converted to a vibration value (represented as multiple of g) and three directional acceleration in g.

0x07	0x86	Raw Ax reading	Raw Ay reading	Raw Az reading
Byte[0]	Byte[1]	Bytes[2 − 3]	Bytes[4 − 5]	Bytes[6 − 7]

Table 12: Sub-packet of a three-axis accelerometer, MMA8452Q

● **SPV1840LR5H-B:** SPV1840LR5H is a MEMS microphone that is sampled at high frequency to obtain the peaks and calculate the sound intensity for a time window. The coresense firmware collects data from this sensor through analog read and the readings are packetized as the raw value as they are collected. The raw readings will be converted to sound level in dB.

0x08	0xFF	64 times of Raw reading
Byte[0]	Byte[1]	Bytes[2 − 129]

Table 13: Sub-packet of a sound level sensor, SPV1840LR5H-B

3.2.3 Lightsense

• **HMC5883L:** HMC5883L is a three-axis magnetometer. The magnetic field strengths in three orthogonal directions, x, y and z are obtained from the sensor. The coresense firmware collects data from this sensor through I2C and the readings are packetized as the raw value as they are collected. The raw readings will be converted to three directional magnetic field in G.

0x0A	0x86	Raw Hx reading	Raw Hy reading	Raw Hz reading
Byte[0]	Byte[1]	Bytes[2 − 3]	Bytes[4 − 5]	Bytes[6 − 7]

Table 14: Sub-packet of a three-axis magnetometer, HMC5883L

• **HIH6130:** HIH6130 is a temperature and relative humidity sensor. The coresense firmware collects data from HIH6130 through I2C and the readings are packetized as the raw value as they are collected. The raw readings will be converted to temperature in centigrate, humidity in %RH.

0x0B	0x84	Relative Humidity in Format 6	Temperature in Format 6
Byte[0]	Byte[1]	Bytes[2 − 3]	Bytes[4 − 5]

Table 15: Sub-packet of a temperature and relative humidity sensor, HIH6130

• **APDS-9006-020, TSL260, TSL250, MLX75305, and ML8511:** APDS-9006-020, TSL260, TSL250, MLX75305, and ML8511 are light sensors that produce the analog voltage representing in general luminance, irradiance measured in $\mu W/cm^2$, or UV index. The coresense firmware collects data from sensors listing above through I2C and the reading is packetized as the raw value as it is collected. The raw reading will be converted to temperature in centigrate, humidity in %RH.

Sensor ID (0x0C ∼ 0x10)	0x82	Voltage output in Format 1
Byte[0]	Byte[1]	Bytes[2 − 3]

Table 16: Sub-packet of light intensity sensors, APDS-9006-020, TSL260, TSL250, MLX75305, and ML8511

• **TMP421:**　TMP421 is a temperature sensor. The coresense firmware collects data from TMP421 through I2C and the reading is packetized as the raw value as it is collected. The raw reading will be converted to temperature in centigrate.

0x13	0x82	Temperature in Format 6
Byte[0]	Byte[1]	Bytes[2 − 3]

Table 17: Sub-packet of a temperature sensor, TMP421

3.2.4　Chemsense:

　　Chemsense board sends data through Serial3 communication line in metsense board. All sensor values from chemsense board are caluated in its own firmware. The raw serial readings from the board are packetized as the raw value as they are collected, and the reading will be sorted out in database.

• **Chemsense reading:**　To fully collect all the data from chemsense board, the firmware collect data three times in a row, so that coresense firmware sends three packets when it gets request of reading of chemsense board.

0x2A	Varies	Raw reading
Byte[0]	Byte[1]	Bytes[Vareis]

Table 18: Sub-packet of a chemsense board

• **Chemsense firmware configuration:**　**THIS DATA DO NOT FOLLOW CORESENSE PACKET FORMAT, NO FORMAT AT ALL** Chemsense Fiwmware configuration is one-time collectable charactor data, when the board is powered on. The coresense firmware collectes data when the board is powered on, and store the data into an array. The user can call this data with their own plugin (request and answer collecting functions).

1st line	"Start sending Chemsense FW configuration"
2nd − 48th line	Each raw reading line
last line	"End sending Chemsense FW configuration"

Table 19: Sub-packet of chemsense firmware configuration

3.2.5　Alpha Sensor:

　　Coresense firmware communicates with alpha sensor through SPI. The raw SPI readings from the board are packetized as the raw value as they are collected, and the reading will be sorted out in database. In this document, concise Alpha sensor information is provided. For more information, refer to **'Firmware commands version 18.xls' and 'OPC-N2 Manual Issue December 2015.pdf'** at **'waggle/docs/alphasense-opc-n2'**.

● **Histogram**   Histogram of alpha sensor is 62 bytes of reading. This parameter provides various information as listed below. **All data from the alpha sensor is LSB first**.

0x28	0xBE	Various sensor information as listed above
Byte[0]	Byte[1]	Bytes[2 − 63]

Table 20: Sub-packet Alpha sensor histogram

○ Bin Counts (Bin0 - Bin15) are unsinged 16 bit integer variables (Bytes[2 − 33]).

○ An unsigned 8-bit integer represents the average amount of time that particles sized in the stated bin took cross the laser beam of the sensor. Value 10 represents 3.33 μs (Bytes[34 − 37]).

○ The sample flow rate in ml/s is provided as a float variable (Bytes[38 − 41]).

○ Temperature and Pressure alternating. Temperature is an unsigned 32-bit integer that represents temperature in Celsious multiplied by 10. Pressure is an unsigned 32-bit integer that represents pressure in pascals (Bytes[42 − 45]).

○ Actual sampling period of a measure of the historgam in seconds provided as 4 bytes of float variables (Bytes[46 − 49]).

○ The least significat 16-bit of the sum of the counts in all the histogram bins is provided by unsigned 16bit integers (Bytes[50 − 51]).

○ A float variable occupying 4 bytes for PM1. Unit is μg/m^3 (Bytes[52 − 55]).

○ A float variable occupying 4 bytes for PM2.5. Unit is μg/m^3 (Bytes[56 − 59]).

○ A float variable occupying 4 bytes for PM10. Unit is μg/m^3 (Bytes[60 − 63]).

Parameter	Data location
Bin Count (32 Bytes)	Bytes[2 - 33]
Average Time (4 Bytes)	Bytes[34 − 37]
Sample flow rate (4 Bytes)	Bytes[38 − 41]
Temp/Pressure(alter) (4 Bytes)	Bytes[42 − 45]
Sampling period (4 Bytes)	Bytes[46 − 49]
Sum of the counts (2 Bytes)	Bytes[50 − 51]
PM1 (4 Bytes)	Bytes[51 − 55]
PM2.5 (4 Bytes)	Bytes[56 − 59]
PM10 (4 Bytes)	Bytes[60 − 63]

Table 21: Detail sub-packet of Alpha sensor histogram

● **Serial**   Serial of alpha sensor is 20 bytes of reading. This parameter provides the serial number of the alpha sensor by character.

0x29	0x94	Raw values
Byte[0]	Byte[1]	Bytes[2 − 21]

Table 22: Sub-packet of Alpha sensor serial number

● **Firmware** Firmware of alpha sensor is 2 bytes of reading. This parameter provides the Firmware version of the alpha sensor by two unsigned 8-bit integer.

0x30	0x82	Raw values
Byte[0]	Byte[1]	Bytes[2 − 3]

Table 23: Sub-packet of Alpha sensor firmware

● **Configuration** **THIS DATA DO NOT FOLLOW CORESENSE PACKET FORMAT, NO FORMAT AT ALL** Configuration of alpha sensor is 256 bytes of reading. This parameter provides various information as listed below. Since this configuration data is too long to fit into a sub-packet, the data are not packed into coresense packet.

1st line	"Start sending alpha sensor configuration"
2nd line	256 bytes of raw reading
last line	"End sending alpha sensor configuration"

Table 24: Sub-packet of alpha sensor configuration

○ Bin Boundaries (BB0 - BB14) are unsinged 16 bit integer variables, and two spare bytes.

○ Bin Particle Volumes (BPV0 - BPV15) are float variables occupyting 4 bytes each.

○ Bin Particle Densities (BPD0 − BPD15) are float variables occupying 4 bytes each.

○ Bin Sample Volume Weightings (BSVW0 − BSVW15) are float variables occupying 4 bytes each.

○ Gain Scaling Coefficient (GSC) is float variable occupying 4 bytes.

○ Sample flow rate is a float variable occupying 4 bytes.

○ Laser DAC value is unsigned 8bit interger variable.

○ Fan DAC value is unsigned 8bit integer variable.

○ Time of Flight to Sample Flow Rate conversion factor is unsigned 8bit integer variable.

○ 21 spare bytes follow Configuration variables.

4 Sensor Data Units

4.1 Raw and Processed

The sensor boards output a set of raw readings which need to be calculated for proper units for each of data. The table below lists the various units for each sensor values. 'Current Units' in the table means the unit after conversion through designated equations stored in database currently (Nov, 2017), and 'Processed Units' means the unit which will be updated in the future.

Table 25: Sensor units

Sensor/Parameter	Current Units	Processed Units	Comments
Firmware version	No Units	No Units	
Airsense board			
Met/Lightsense MAC	No Units	No Units	
TMP112	°C	°C	
HTU21D	°C, %RH	°C, %RH	
BMP180	°C, Pa	°C, Pa	
PR103J2	°C	°C	
TSL250RD	Raw reading*	$\mu w/m^2$	
MMA8452Q	g, g, g	g, g, g, g	
SPV1840LR5H-B	Raw reading*		
TSYS01	°C	°C	
Lightsense board			
HMC5883L	G, G, G	G, G, G	
HIH6130	°C, %RH	°C, %RH	
APDS-9006-020	Raw reading	lux	
TSL260RD	Raw reading*	$\mu w/m^2$	
TSL250RD	Raw reading*	$\mu w/m^2$	
MLX75305	Raw reading*	$\mu w/m^2$	
ML8511	Raw reading*	UV index	
TMP421	°C	°C	
Chemsense board			
Chemsense configuration	No unit	No unit	Don't need conversions
Chemsense reading	Raw reading*	See following section	
			Continued on next page

Table 25 – continued from previous page

Sensor/Parameter	Current Units	Processed Units	Comments
Total reducing gases	No unit/Raw reading		Raw ADC reading
Nitrogen dioxide			
Ozone	AFE ADC counts		Raw ADC reading
Hydrogen sulphide			
Total oxidizing gases			
Carbon monoxide			
Sulfur dioxide			
SHT25	100ths of °C / %RH	°C, %RH	
LPS25H	100ths of °C, Pa	°C, Pa	
Si1145	Three fixed dummy value		Uncompleted FW
Intel MAC address	No Units	No Units	
CO ADC temp	100ths of °C	°C	
IAQ IRR ADC temp			
O3 NO2 ADC temp			
SO2 H2S ADC temp			
CO LMP temp			
Accelerometer	raw register		Raw reading
Gyro			
Alpha sensor			
Serial Number			
Serial	No unit	No unit	
Firmware			
Firmware	No unit	No unit	
Histogram			
Bin count	No unit	No unit	
Average time	Raw reading*	μs	value 10 = 3.33 μs
Sample flow rate	Raw reading*	ml/s	
Temp/Pressure(alter)	Raw reading*	°C / hPa (alter)	
Sampling period	Raw reading*	s	
Sum of the counts	No unit	No unit	
PM1	Raw reading*	μg/m^3	
PM2.5	Raw reading*	μg/m^3	
			Continued on next page

Table 25 – continued from previous page

Sensor/Parameter	Current Units	Processed Units	Comments
PM10	Raw reading*	$\mu g/m^3$	
Configuration			
Bin boundaries	No unit	No unit	
Bin particle volumes	Raw reading*		
Bin particle densities	Raw reading*		
Bin sample volume weightings	Raw reading*		
Gain scaling coefficient	Raw reading*		
Sample flow rate	Raw reading*		
Laser DAC	Raw reading*		
Fan DAC	Raw reading*		
Conversion factor	Raw reading*		
Spare bytes	No unit	No unit	

*Raw reading means the reading has no unit currently, and conversion equation for the sensor is not implemented in database. The data stored in database for this sensor is just integer form of raw reading.