INTELLIGENT SERIES GAS SENSORS iSERIES

Intelligent Series Gas Sensors (iseries): Communication Protocol

SDCS: SMART DEVICE COMMUNICATION STANDARD

Contents

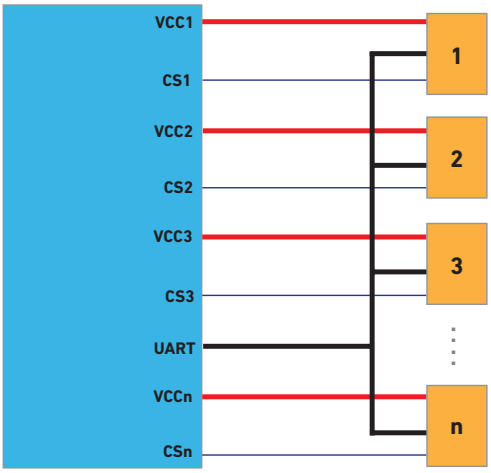
Honeywell

COMMUNICATION PROTOCOL
SDCS SMART DEVICE COMMUNICATION STANDARD

1.0 INTRODUCTION

SDCS (Smart Device Communication Standard) is a digital protocol that is used to establish communication between the iseries sensors and an instrument (microcontroller, personal computer, etc). The iseries sensors allows to connect several sensors to a single instrument, the following schematic describes the electrical connection between the instrument (large blue rectangle) and sensors (small orange rectangles).

Figure 1. The UART bus (black line) consists of the receiver (Rx) and transmitter (Tx) lines. Chip Select (blue line) alternates the communicate between sensors. The red line represents the sensor supply voltage.



The following sections will describe in detail how to operate the sensor. Refer to the Technical Communication Appendix for information about the power consumption in sleep mode, wake up times and sensor time out.

IMPORTANT

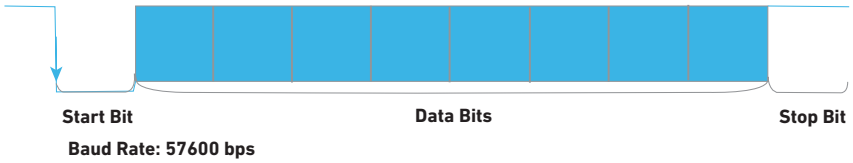
The iseries sensors should be powered continuously while they are in an instrument.
*Otherwise, UF, write-protect and RTC would revert to their default values.

1.1 SDCS FORMAT:

TABLE 1. SDCS FORMAT

DESCRIPTION	VALUE
Baud rate	57600 bps
Data bits	8
Start bit	1, always low
Stop bit	1, always high
Parity	None

Figure 2. Data Bit Representation



1.2 CHIP SELECT ACTIVATION

To initiate communication with the sensor, a falling edge (a high jump to low) must be sent to Chip Select (CS); this will activate the communication port.

Figure 3. Falling Edge

In the same way, to deactivate the communication port a rising edge (a low jump to high) must be sent to CS.

Figure 4. Rising Edge

1.2.1 CS response: waiting time

The instrument should wait for 500 µs before establishing communication with the sensor, otherwise the information sent to it will not be received correctly, causing data loss.

Figure 5. Chip Select Response

	Time	Description
Chip Select activation	To	Falling edge to activate CS
Waiting time (CS)		Data can be sent to the sensor after 500 us
Rx (from instrument to sensor)	To	Data
Sensor timeout		Time out: the sensor should respond within a 250 ms timeframe
Tx (from sensor to instrument)	To	Data

IMPORTANT

Chip Select should be controlled by the instrument, even when an instrument is set to a single sensor configuration. CS should not be grounded; failing to do so will increase the current on stand-by mode by approximately 100 µA.

1.3 COMMUNICATION MODES

There are two main communication modes:

- Responder mode: This is the **default operating mode** at which the **sensor** will not initiate communication. It only **responds to commands** sent by the instrument, i.e. the instrument sends a command, and the sensor will answer back.
- Aloha mode: The sensor will constantly send gas readings (data packets) to the instrument once this mode is configured. The sampling frequency can be configured through Aloha mode (consult Aloha commands). Additionally, the sensor will also send a warning (data packet) if a gas threshold has been reached. One purpose of the Aloha mode is to **allow power to be saved** by the instrument being able to go to sleep and be woken up by the sensor when it detects gas. Note that this mode can only be used when the instrument is interfacing with a single sensor.

1.4 BRIEF LIST OF COMMANDS

The external controlling device can request information from the sensor with commands. The external controlling device (instrument) can request the following instructions:

- Acquisition of parameters, concentration readings from sensors
- Calibration of sensors
- Request of calibration parameters and data from sensor
- Set alarms, deadband, real time clock, OEM lock and other sensor parameters

Additionally, it is possible to set the sensors parameters. The following table contains a summarized list of the commands that are available for the iseries Intelligent Gas Sensors.

TABLE 2. AVAILABLE COMMANDS

COMMANDS	GET (REQUEST INFORMATION)	SET (CONFIGURE)
Sensor readings: Gas concentration and temperature Sensor flags: Status, alarms and errors	●	
Product name	●	
Firmware version	●	
Serial number	●	
Type of sensor: CO/H_2S/LEL/etc.	●	
Manufacturing date	●	
*Diagnostic test log report: data from previous diagnostic tests can be requested	●	
*[2]End of life	●	
Real-time clock: Date and time		●
User factor (UF will help identify what type of instrument it is, and in this manner, the installed sensor will be fully compensated and calibrated)		●
Work mode, sleep mode (stand-by) and reset sensor		●
*[2]Reset STEL & TWA alarm counter		●
*[2]Predictive calibration: The output of this function will depend on the accuracy	●	●
*[2]Accuracy	●	●
*[3]Reading units: ppm, ppb, % lel, % vol	●	●
Calibration: set calibration points/ get calibration parameters	●	●
Calibration: time interval	●	●
Bump test: set time interval, get bump test due days	●	●
OEM lock will be set during sensor manufacture so won't be user settable. Second level lock will be settable by OEM/user	●	●
Sensor parameters (Span for Calibration, Short Term Exposure Limit, Time-Weighted Average, Low and High Alarm Levels, etc)	●	●
Dead band configuration (Dead band is a region where a change in gas concentration produces no variation in the sensor measurement output)	●	●
Sampling rate where the maximum sampling rate is 1 sample per second (refer to Aloha mode by period)	●	●
Alarm setting by average over time (STEL & TWA) and ceiling alarm (threshold)	●	●
*[4]Compliance standard measurement	●	●
*[4]Calibration gas and target gas	●	●

* Applies only for electrochemical sensors
*[2] Applies only on toxic gas sensors
*[3] Available units depend on the type of sensor
*[4] Applies only for LEL sensors

1.5 PACKET FORMAT

Table 3 depicts the typical architecture of a received (or sent) data packet. The format of the data packets will remain the same for transmitted and received packets; nevertheless, the size of these would vary from command to command.

TABLE 3. PACKET ARCHITECTURE

ITEM	SOP	VER-SION	LENGTH	AI	CMD	DATA	CRC	EOP
Size (Bytes)	1	1	1	2	1	0-n	2	1
Content	0x7B	0x59	Number of bytes from **AI to EOP**	Auto increment from 0 to 65535	Command code: various	Various (maximum 128 bytes)	CRC-16: Count from SOP to data	0x7D

An SDCS packet is divided into eight different subsections: SOP, Version, Length, SI, CMD, Data, CRC and EOP; where **SOP** is the start of packet, **Version** corresponds to the firmware version, **Length** describes the size of the packet (from SI to EOP), **AI** is an auto-increment index that goes from 0 to 65535 (0xFF FF), **CMD** the command code, **Data** the data requested by the command code, **CRC** the error-detecting and correction code used to detect accidental change to raw data and **EOP** the end of packet.

1.5.1 Cyclic redundancy check implementation

A Cyclic Redundancy Check is a verification method used to ensure that data being sent is not corrupted during transfer.

The sensor system calculates the verification or check code and adds it to the packet (CRC subsection). In the instrument, the data must go through the same process. If the CRC produced at the instrument does not match the sent CRC, then the data is corrupt. At this point, the instrument can request to either ignore the data or retransmit the request.

The polynomial key for the iseries sensors is described in the CRC-16 Appendix, where the key is translated into a binary expression by expanding the polynomial.

1.5.2 Instrument and sensor indexing

The auto-increment index featured in the sensor can be used together with a second instrument auto-increment index implemented at the instrument level. In this manner, if a packet is lost (either from the instrument to the sensor or the other way around), it will be possible to track the packet number that was not sent/received.

2. SDCS COMMANDS

The communication between the instrument and sensor is made by SDCS commands. In this document, the commands are divided into three different categories: Set, Get and Aloha commands.

Get commands are used to obtain data out of the sensor, Set commands are used to modify the data in the sensor and Aloha mode is used to automatically receive gas measurements once Aloha mode has been set. Consult section 2.3 for additional information about Aloha mode.

2.1 GET COMMANDS

These commands are used to request and retrieve information contained within the digital sensor. Some of these commands can request the information simply by sending the command to the sensor module.

Note that the commands are expressed in hexadecimal for convenience; however, the instrument must send the command in its binary equivalent.

Other commands require more detailed information apart from the command, such as sensor index (some sensors are able to detect more than one target gas).

In other cases, the instrument is required to send a more specific request. For instance, when the user requests the last calibration date it is necessary to specify the sensor index (some sensors can detect two gas types) and the type of calibration point (zero or span).

Table 4. GET Commands

GET COMMANDS	INSTRUMENT REQUEST (DATA)	DESCRIPTION AND SENSOR RESPONSE
Product name GET_PROD_NAME CMD 0x11		Displays the product name of the sensor in HEX to ASCII format. e.g. “iCO”, “iH2S”, etc. Note: The product name data string ends with 0x00
Sensor firmware GET_FW_VER CMD 0x12		Displays the firmware version of the sensor in HEX to ASCII format. e.g. “V0.44CB”
Serial number GET_SEN_SN CMD 0x13		Displays the sensor serial number in HEX to ASCII format Note: The serial number data string ends with 0x00
Number of sensors GET_SEN_SUM CMD 0x15		Some sensors can detect more than one gas. This command requests the number of gases that the sensor can detect. [0] Maximum number of sensor types [1-2] Bitmap index (installed sensor index) **Note:** In Bitmap index, each bit represents one different gas type e.g. 0x5A = 0 1 0 1 1 0 1 0 (which would mean that the sensor index 1,3,4 and 6 correspond to installed gas types supported by the sensor)
Production date GET_ PROD_DATE CMD 0x37		Displays the production date of the sensor in the following order: [0] Year (0*-99) *Where 0 stands for 2000 [1] Month (1-12) [2] Day (1-31) [3-4] Reserved
OEM code GET_OEM_CODE CMD 0x3B		Sensors can have an OEM specific code programmed in, which is written during manufacture and cannot be modified. Instrument can check that the sensor has the unique code – if not, then sensor has not been sourced through them and instrument can refuse the sensor. This command displays the EOM ID code in HEX to ASCII string (no more than 6 characters). This value is set on request by Honeywell and corresponds to the first OEM code level. The default value is NoLock **An example can be found in the supplementary documentation (Appendix II: Data decoding examples, i.3)*
OEM code GET_PARTNER_CODE CMD 0x40		Displays the OEM code in HEX to ASCII string (no more than 6 characters). This command is the **second OEM code** and it can be set by the user. **Important:** Once the code has been assigned, it won’t be able to be modified.

COMMUNICATION PROTOCOL
SDCS SMART DEVICE COMMUNICATION STANDARD

GET COMMANDS	INSTRUMENT REQUEST (DATA)	DESCRIPTION AND SENSOR RESPONSE
Data pack GET_DATA_PACK CMD 0x30	**This is the main command to obtain gas measurements.** [0] Sensor Index Some iseries devices can measure more than one gas type. This byte will specify which is the one that is being measured (refer to CMD GET_SEN_SUM 0x15) [1] Bitmap; High B15-B9: Reserved B8: Negative Gas Reading (for debugging purposes only) [2] Bitmap; Low B7: Uncompensated Gas Reading (for debugging purposes only) B6: Humidity (if applicable) B5: Temperature B4: Sensor Raw Counts **B3: Gas Reading** B2: Sensor Error B1: Sensor Alarm B0: Sensor Status Note: The instrument can request information such as the sensor status, sensor errors, gas measurements and temperature/humidity. For example, to request the status (B0), gas reading (B3) and temperature (B5): [2] = 0x29 = 0010 1001	It displays the sensor's status, error, gas reading and temperature/humidity measurements. The length of the response will depend on the number of parameters requested. **The order of appearance of the requested bytes is as follows:** * Note that the numeration is in brackets, not parenthesis. (0) Status: B0: Reserved B1: In warm up B2: Reserved B3: In calibration B4-B5: Reserved B6: In sleep mode B7: Reserved (1) Alarm: B0: Over range B1: User factor hasn't been set B2: Time is not synchronized (RTC hasn't been set) B3: High alarm B4: Low alarm B5: STEL (Short Term Exposure Limit) B6: TWA (Time-weighted average) B7: Drift (flagged when the apparent gas concentration reading has gone too far negative) (2) Error code: Byte 0: Number of error codes Byte 1-n: Error codes (1 byte each error code) List of error codes (**NOT** in order of priority): Error code 001 = Diagnostic electrode failure Error code 101 = Sensing electrode impedance too high Error code 102 = Reference electrode failure Error code 103 = Electrolyte too dry Error code 104 = End of life Error code 105 = Counter electrode failure Error code 106 = Broken bead/short circuit (LEL) Error code 108 = LED/PD Failure (NDIR) Error code 109 = Span calibration is due Error code 110 = Bump test is due Error code 111 = User factor not valid Error code 112 = Operational temp out of range Error code 113 = Electrolyte too wet The order or priority is found in troubleshooting appendix. (3) Gas reading: Concentration measurement is 4 bytes **signed integer**. Gas measurement= Hex to Dec/100 (4) Raw counts gas readings: (for debugging purposes only) The **first byte** represents the number of raw counts and the rest is the raw counts (**each raw reading is 2 bytes**). (5) Temperature reading = Decimal value - 127 (1 unsigned byte) (6) Humidity = 0xFF when is not supported in sensor (1 unsigned byte) (7) Uncompensated Reading: (For debugging purposes only). The uncompensated gas concentration measurement is **4 bytes signed integer**

GET COMMANDS	INSTRUMENT REQUEST (DATA)	DESCRIPTION AND SENSOR RESPONSE
		Uncompensated Gas measurement= Hex to Dec/100 (8) Negative Gas Reading: (For debugging purposes only). After this bit is set, the user will be able to read negative values, i.e. the sensor will output negative reading if the actual reading is below zero. The negative gas concentration measurement is **4 bytes signed integer.** It is recommended to deactivate the dead band to be able to read this measurement. Negative compensated gas measurement= Hex to Dec/100 _____ For instance, if only the status, gas reading and temperature are requested, the order of data will be: [0] Status [1-4] Gas reading [5] Temperature **A few examples can be found in the supplementary documentation (Appendix II: Data decoding examples: ii.1, ii.2 & ii.3)*
Sensor data format GET_DATA_FMT CMD 0x31	[0] Sensor Index	Displays the unit of the current gas reading (such as the measurement unit in ppm, ppm, % lel, % vol, etc.) and reading resolution coefficients (needed to obtain the sensor resolution). [0] Unit codes 0x00: **ppm** (parts per million) 0x01: **%** (gas percentage) 0x02: **ppb** (parts per billion) 0x27: **% LEL** (lower explosive limit) 0x28: **% VOL** (volume percentage) [1] Reading resolution integer (from 1 to 255) [2] Reading resolution exponent (from -4 to 4) [3] Mask parameter; High (Reserved) B12-B15: Reserved B11: Drift (Read-only) B9-B10: Reserved B8: Zero calibration (**only for oxygen sensor**) Equivalent to oxygen percentage in clean air (default value is 20.9% O_2) [4] Mask parameter; Low B7: Reserved B6: TWA B5: STEL B4: Over range (Read-only) B3: Span high B2: High alarm B1: Low alarm B0: Span **Note:** The mask parameter determines whether a parameter is enabled or not (where 0 is disabled and 1 is enabled) **Note 2:** The resolution can be calculated with the following formula: Resolution = (Res Integer) * (10^Res Exponent) **An example can be found in the supplementary documentation (Appendix II: Data decoding examples, i.6)*

COMMUNICATION PROTOCOL
SDCS SMART DEVICE COMMUNICATION STANDARD

GET COMMANDS	INSTRUMENT REQUEST (DATA)	DESCRIPTION AND SENSOR RESPONSE
Sensor name GET_TARGET_GAS CMD 0x35	[0] Sensor Index	Displays the target gas in HEX to ASCII e.g. CO, H_2S, O_2, etc. **Note:** The sensor name data string ends with 0x00. **An example can be found in the supplementary documentation (Appendix II: Data decoding examples, iii.1)*
Sensor parameter GET_SEN_PARA CMD 0x33	The instrument requires to specify the sensor index and parameter that is requesting to get. For instance, if the byte corresponding to the low parameter mask [2] is FF(HEX) it would request all the parameters available (1111 1111 in binary). [0] Sensor Index [1] Mask Parameter; High B15: Reserved B14: Reserved B13: Reserved B12: Reserved B11: Drift B10: Reserved B9: Reserved B8: Zero calibration: **only for oxygen sensor** (Equivalent to oxygen percentage in clean air) [2] Mask Parameter; Low B7: Reserved B6: Time-weighted average (TWA) B5: Short term exposure limit (STEL) B4: Over range (read-only) B3: Span High B2: High B1: Low B0: Span	Displays the sensor parameters such as the short-term exposure limit, over range, time-weighted average, span high and low concentration alarm levels. The length of the response will depend on the number of parameters requested. The order of appearance of the requested bytes are as follows: Mask Parameter; Low (0) Span (1) Low (2) High (3) Span high (4) Over range (read-only) (5) Short term exposure limit (STEL) (6) Time-weighted average (TWA) (7) Reserved Mask Parameter; High: Reserved (8) Zero calibration: **only for oxygen sensor** (Equivalent to oxygen percentage in clean air) (9) Reserved (10) Reserved (11) Drift (12) Reserved (13) Reserved (14) Reserved (15) Reserved _____ For instance, if Overrange, STEL and TWA are requested, the order of these the bytes sent from the sensor to the instrument would be (1st)=Over Range, (2nd)=STEL and (3rd)=TWA. Note: All Mask Parameters have **4 unsigned bytes each**, and they can be decoded by using the following formula: Parameter Value= Hex to Dec/100 _____ **An example can be found in the supplementary documentation (Appendix II: Data decoding examples, v.1)*

GET COMMANDS	INSTRUMENT REQUEST (DATA)	DESCRIPTION AND SENSOR RESPONSE
End of life GET_END_OF_LIFE CMD 0x41	[0]= Sensor Index	Due days before the sensor expiries. For electrochemical sensors, the time is calculated based on the historical environmental conditions (it does not account for poisoning). For LEL, it is a countdown from the production date to the last day of the expected operating life of the sensor (it does not account for poisoning) [0-1]= Due days before end of life **Note:** The End-of-Life algorithm takes into account historical data such as the temperature, electrolyte concentration, sensitivity and time. It is important to pay special attention into setting the RTC so it reflects the actual date and time. Otherwise, end of life measurement will be invalid. The End-of-Life calculation is made automatically 60 seconds after the sensor is set to sleep mode. The test lasts approximately 30 seconds. If the sensor is kept in sleep mode, the test will be repeated every 24 hours. **An example can be found in the supplementary documentation (Appendix II: Data decoding examples, i.7)*
Predictive calibration GET_PREDCAL_DUE_DAYS CMD 0x42	[0] Sensor Index	Estimated days remaining to calibrate sensor [0-1] = Due days for calibration (countdown timer or predictive calibration timer). The sensor will respond with whichever is lower, the countdown timer or the predictive calibration reading. For non-electrochemical sensors, the function is solely a countdown beginning from the last calibration date. The countdown depends on the calibration period (configurable through CMD 0x8F). **Non-electrochemical: When the countdown reaches 0 days, an error code will be flagged in datapack (0x30) and Aloha datapack (0xA3).** For electrochemical sensors, the function has two timers: a countdown timer and a predictive calibration timer. For the predictive calibration, a calculation taking historical measurements such as the temperature, electrolyte concentration, sensitivity, accuracy and time are taken into consideration. **Electrochemical: When any of the counters reach 0 days, an error code will be flagged in datapack (0x30) and Aloha datapack (0xA3).** **Note:** It is important to pay special attention into setting the RTC so it reflects the actual date and time. Otherwise, predictive calibration measurement will be invalid. The predictive calibration timer is calculated automatically 60 seconds after the sensor is set to sleep mode. The test lasts approximately 30 seconds. If the sensor is kept in sleep mode, the test will be repeated every 24 hours. **An example can be found in the supplementary documentation (Appendix II: Data decoding examples, i.8)*

COMMUNICATION PROTOCOL
SDCS SMART DEVICE COMMUNICATION STANDARD

GET COMMANDS	INSTRUMENT REQUEST (DATA)	DESCRIPTION AND SENSOR RESPONSE
Calibration Time GET_CAL_TIME CMD 0x43	[0] Sensor Index	Requests the calibration time used to calibrate the sensor [0-1] = Calibration time in seconds **An example can be found in the supplementary documentation (Appendix II: Data decoding examples, vi.1)*
Dead Band GET_ DEADBAND CMD 0x45	[0] Sensor Index	Dead band is a region of gas concentrations where a change in concentration produces no change in measurement output. This function allows to configure a gas concentration dead band. Consult SDCS appendix for additional information. [0] = Set status 0x00 Disable 0x01 Enable [1-4] = Dead band outgoing limit [5-8] = Dead band incoming limit **Note:** The dead band values can be read by using the gas reading format (**4 bytes**) Dead band values= Hex to Dec/100

GET COMMANDS	INSTRUMENT REQUEST (DATA)	DESCRIPTION AND SENSOR RESPONSE
Calibration data GET_CAL_ DATA CMD 0x46	[0] Sensor Index	Displays the last calibration date of the sensor. [0] Number of channels supported by sensor B0-B3: Number of raw sensor channels B4-B7: Reserved **Zero calibration** [1...4]: Calibration point value (in gas reading format) [5...n]: Calibration raw counts, each count has a length of 2 bytes. **n** will depend on the number of channels supported by sensor [n+1]: Cal Temperature - 127 [n+2]: Cal Humidity, 0xFF when the humidity measurement is not supported in sensor [n+3]: Year (0*-99) *Where 0 stands for 2000 [n+4]: Month (1-12) [n+5]: Day (1-31) [n+6]: Hour (0-23) [n+7]: Minute (0-59) [n+8]: Second (0-59) **Span calibration** [n+9...n+12]: Calibration point value (in gas reading format) [n+13...**m**]: Calibration raw counts, each count has a length of 2 bytes. **m** will depend on the number of gases supported by sensor [m+1]: Cal Temperature - 127 [m+2]: Cal Humidity, 0xFF when the humidity measurement is not supported in sensor [m+3]: Year (0*-99) *Where 0 stands for 2000 [m+4]: Month (1-12) [m+5]: Day (1-31) [m+6]: Hour (0-23) [m+7]: Minute (0-59) [m+8]: Second (0-59) **Span High calibration** [m+9...m+12]: Calibration value (in gas reading format) [m+13...**p**]: Calibration raw counts, each count has a length of 2 bytes. **p** will depend on the number of gases supported by sensor [p+1]: Cal Temperature - 127 [p+2]: Cal Humidity, 0xFF when the humidity measurement is not supported in sensor [p+3]: Year (0*-99) *Where 0 stands for 2000 [p+4]: Month (1-12) [p+5]: Day (1-31) [p+6]: Hour (0-23) [p+7]: Minute (0-59) [p+8]: Second (0-59)
Bump test due days GET_BUMP_ DUE_DAYS CMD 0x47	[0] Sensor Index	Due days to get a new bump test [0-1] = Due days for bump test _____ When the counter reach 0 days, an error code will be flagged in datapack (0x30) and Aloha datapack (0xA3)

GET COMMANDS	INSTRUMENT REQUEST (DATA)	DESCRIPTION AND SENSOR RESPONSE
Predictive calibration GET_ PRED_ CAL_DUE_DAYS CMD 0x48	[0] Sensor Index	Estimated days remaining to calibrate sensor. [0-1] = Due days for calibration (countdown timer) [2-3] = Due days for calibration (predictive calibration) This command can be used ONLY on electrochemical sensors. The function has two timers: a countdown timer and a predictive calibration timer. For the predictive calibration, a calculation taking historical measurements such as the temperature, electrolyte concentration, sensitivity, accuracy and time are taken into consideration. The countdown timer starts the countdown from the last calibration date. The countdown depends on the calibration period, which is configurable through CMD 0x8F. **Note:** It is important to pay special attention into setting the RTC so it reflects the actual date and time. Otherwise, predictive calibration measurement will be invalid. The predictive calibration calculation is made automatically 60 seconds after the sensor is set to sleep mode. The test last approximately 30 seconds. If the sensor is kept in sleep mode, the test will be repeated every 24 hours. _____ **When the any counter reach 0 days, an error code will be flagged in datapack (0x30) and Aloha datapack (0xA3)** **This command is exclusive for EC sensor*
Errors during calibration GET_CAL_ ERRORS CMD 0x49	[0] Sensor Index	This command provides a detailed diagnostic of an unsuccessful calibration. [0-1] = Type of error B0: Flash error B1: Minimum sensitivity (sensor detects less than 50 % of its initial sensitivity) B2-B7: Reserved [2] = Last unsuccessful type of calibration (where the error occurred) 0x00, Zero 0x01, Span
Gas list GET_GAS_ LIST CMD 0x51	[0] Sensor Index	The sensor can be configured to detect different target gases. This command displays the list of gases at which the sensor can be configured. The list of gases is displayed with their corresponding chemical formula separated by a comma (all in ASCII format). [0...n] = Supported list of gases **Note:** The string must end with 0x00 For instance, for an LEL sensor the gas list may be: CH_4,C_4H_{10},H_2,C_5H_{12},C_3H_8 **This command is exclusive for LEL sensors*

GET COMMANDS	INSTRUMENT REQUEST (DATA)	DESCRIPTION AND SENSOR RESPONSE
Calibration and target gas request GET_GAS_ CAL_ MES CMD 0x52	[0] Sensor Index	LEL sensors can be configured to measure different target gases. They can also be calibrated with various gases. This command requests the name of the target gas and the gas that can be used to calibrate the sensor. The default target and calibration gas is methane (CH4). If the target or calibration gases are not changed, the default response will be **0x2C00** If the gases had been changed by the user, the response will be: [0...n] = Calibration gas [n+1] = comma (“,”) [n+2]= Measurement gas. **Note:** String ends with 0x00. _____ **Important:** The target gas and calibration gas will return to its default value (CH_4) when the sensor is reset or unpowered. **This command is exclusive for LEL sensors*
Sensor unit GET_GASUNIT_ LIST CMD 0x54	[0] Sensor Index	Displays the list of units that can be implemented by the sensor to read gas concentrations. [0-N]= Available unit codes 0x00: **ppm** (parts per million) 0x01: **%** (gas percentage) 0x02: **ppb** (parts per billion) 0x27: **% LEL** (lowest explosive limit) 0x28: **% VOL** (volume percentage)

GET COMMANDS	INSTRUMENT REQUEST (DATA)	DESCRIPTION AND SENSOR RESPONSE
Data log request GET_EC_ DATALOG CMD 0x60	This command is used to request the historical sensor diagnostics. **Two configurations cases can be presented:** **Case 1:** Requests the size of the field data log array. [0-1] = 0xFF FF	[0]= Length (typically 0x02) [1-2]= Number of reports (array size of the datalog) **Note:** The last element in the array (top of pile) corresponds to the element next item to be written. Consult CMD 0x60 Case 2 for reference.
	Case 2: This function requests a **single element in the field data log array** (single data field diagnostic test). The data corresponding to other elements in the field data log array can be requested by changing the data log index. [0-1] = Data log index (in hex) Data log index represents the array of the environmental and diagnostic test reports. For instance, if the sensor has been running for two days and a half, the data log index will be numbered in the following order: Day 0/Index 0: Test done the day before yesterday Day 1/Index 1: Test done yesterday Day 2/Index 2: Test done today* ***Note:** The last array element on top of the pile will have a null value and it will remain as such until the test is performed at the end of the day (overwriting the value) **Note 2:** For diagnostic test frequency times, consult section Appendix 1, Section 5 (diagnostic tests)	Electrochemical sensors can track and store the environmental conditions at which the sensor had been exposed to. Furthermore, the sensors store the results of the automatically generated diagnostics tests in order to be able to accurately calculate the outputs of the intelligent features, such as the End of Life and Predictive calibration. This command is used to request the field data stored within the sensor (environmental and diagnostic test data log) [0] = Length of the data** [1…N] = Data** corresponding to the diagnostic test log
Accuracy GET_EC_ ACCURACY CMD 0x61	[0] Sensor Index	Displays the accuracy of electrochemical sensors. The result is given in percent accuracy (%). [0-1]= Percent accuracy * 100 Where **percent accuracy** = (HEX to DEC)/100 i.e. 0x 03 E8 = 10% percent accuracy

2.2 SET COMMANDS AND WRITE-PROTECT

Set commands allows you to change the values of some parameters and sets the sensor's operational conditions such as the RTC, UF, write-protect, partner ID, dead band and gas unit. **To use any set command, it is necessary to set the write-protect off.**

2.2.1 WRITE-PROTECT

Write Protect is enabled by default, if Write Protect is not deactivated, the instrument will not be able to set the sensor's criterion, and an error code will be the response of the sensor. To enable of disable the function, the following instructions must be used:

Table 5. Write Protect Command

COMMAND	DATA FROM INSTRUMENT TO SENSOR	SENSOR RESPONSE
Write protect WRITE_ PROTECT CMD 0xA0	Activates or deactivates write-protect [0] = 0x00 for setting write-protect off 0x01 for setting write-protect on **Note: When write-protect is switched-off, the sensor keeps this status for 300 seconds before setting the write-protect on automatically.** **An example can be found in the supplementary documentation (Appendix II: Data decoding examples, i.1)*	

For instance, when the sensor is firstly powered-on, by default it will be in sleep mode. To start the sensor, it is necessary to set-the Write Protect off and then change the sensor mode to Work Mode (by using the GOTO_MODE command).

2.2.2 SET COMMAND: SENSOR RESPONSE

Once any set command has been sent, the sensor will respond with the following packet to corroborate that the operation has been successful.

TABLE 6. SENSOR RESPONSE COMMAND

ITEM	SOP	VER-SION	LENGTH	AI	CMD	DATA	CRC	EOP
Size (Bytes)	1	1	1	2	1	0	2	1
Content	0x7B	0x59	Number of bytes from CMD to EOP	Auto increment from 0 to 65535	Various: 0xA0, 0xA2, 0xA6, 0x80, 0x82, 0x89, 0x8A, 0x8B, 0x8C, 0x8D, 0x8E, 0x8F, 0x90, 0x91, 0x92, 0x93 & 0x95 Consult section 3 if the sensor response is **0x71**		CRC-16: Count from SOP to data	0x7D

Note that the sensor response (confirmation that the command has been processed) does not have DATA subsection in its typical architecture, as compared with other commands.

2.2.3 SET COMMANDS

The following table enlist the supported set commands:

Table 7. SET Commands

COMMANDS	COMMAND DATA (FROM INSTRUMENT TO SENSOR)	SENSOR RESPONSE
Go to mode GOTO_MODE CMD 0xA6	This command changes the operating mode of the sensor. The sensor can be in stand-by mode (sleep), functional mode (work). This function can also reset the sensor. [0]= 0x01 - Reset 0x02 - Sleep 0x03 - Work **Note:** When the sensor is in sleep mode, the power consumption of the sensor will be lowered considerably. During this time, the sensor will not be able to provide gas reading. Once the sensor has changed its mode to Work Mode there will be certain start-up time which would depend on the type of sensor. (Consult Appendix 1: power at sleep mode) **Important:** The sensor should be powered at all time, even if the sensor is in stand-by mode. It is necessary to repeat the start up procedure if the sensor is unpowered or reset. If the sensor has flagged errors prior to the unpowering/ resetting, the errors will be cleared (except from the End of Life indication, which is irreversible). **An example can be found in the supplementary documentation (Appendix II: Data decoding examples, i.2)*	
Sensor parameter SET_SEN_PARA CMD 0x80	This command allow the user to change the span calibration concentrations, zero calibration concentration (iO2), STEL, TWA, Low alarm and High alarm. [0] Sensor Index [1] Mask Parameter; High B15: Reserved B14: Reserved B13: Reserved B12: Reserved B11: Reserved B10: Reserved B9: Reserved B8: Zero calibration (**only for oxygen sensor**) Equivalent to oxygen percentage in clean air (default value is 20.9% O_2) [2] Mask Parameter; Low B7: Reserved B6: Time-weighted average (TWA) B5: Short term exposure limit (STEL) B4: Reserved B3: Span High B2: High Alarm B1: Low Alarm B0: Span [3-n] Parameter data (the length depends of how many parameters are going to be changed). **Important:** The mask parameters have the same format than the one the one described in GET_SEN_PARA, CMD 0x33. Where a single sensor parameter is 4 unsigned bytes. **Sensor parameter** = (HEX to DEC)/100 **A few examples can be found in the supplementary documentation (Appendix II: Data decoding examples: v.2 & v.3)*	

COMMUNICATION PROTOCOL
SDCS SMART DEVICE COMMUNICATION STANDARD

COMMANDS	COMMAND DATA (FROM INSTRUMENT TO SENSOR)	SENSOR RESPONSE
Real Time Clock SET_SEN_RTC CMD 0x82	Sets the internal Real Time Clock of the sensor. [0] Year (0*-99) *Where 0 stands for 2000 [1] Month (1-12) [2] Day (1-31) [3] Hour (0-23) [4] Minute (0-59) [5] Second (0-59) **Caution:** The RTC should be resynchronized daily. **Note:** The sensor will flag an error if the date is set to a date older than the manufacturing date. **Important:** The diagnostic tests (End of Life and Predictive Calibration) take into account the historical value of various parameters such as the temperature, electrolyte concentration and time. It is important to pay special attention into setting the RTC so it reflects the actual date and time. **Important:** Please note that if the RTC is configured incorrectly (at least 5 years after the manufacturing date) the sensor will trigger the End of Life flag, which is irreversible. **An example can be found in the supplementary documentation (Appendix II: Data decoding examples, i.4)*	
Partner code SET_SEN_ PARTNERID CMD 0x89	Sensors can have an OEM specific code programmed in, which is written during manufacture and cannot be modified. Instrument can check that the sensor has the unique code – if not then sensor has not been sourced through them and instrument can refuse the sensor. Similarly to the OEM code, this function offers a second lock, which can be configured by the user. [0-5] Abbreviated partner ID (no more than 6 characters) **Warning: Once this command has been set, it won't be possible to change it once again.**	
Dead band SET_SEN_ DEADBAND CMD 0x8A	[0] = Sensor index [1]= Enable or disable function 0x00 Disable 0x01 Enable [2-5] = Dead band outgoing limit (in gas reading format) [6-9] = Dead band incoming limit (in gas reading format) **Note:** The dead band follows the same format that the 0x33 command (4 unsigned bytes)	
Target gas and calibration gas Set_Sen_Gas_ Cal_Mes CMD 0x8B	Configures the calibration gas and target gas. The list of available gases supported by the sensor can be requested by using the GET_GAS_LIST command (CMD 0x51). [0] = Sensor index [1…n] = Calibration gas [n+1] = 0x44 (comma in HEX to ASCII. i.e. ",") [n+2…p] = Target gas **Note: String starts and ends with 0x00** _____ **Important:** If the sensor is unpowered or reset, the calibration and target gas will go back to its default setting (i.e. methane for both, calibration and target gas). **This command is exclusive for LEL sensors*	

COMMUNICATION PROTOCOL
SDCS SMART DEVICE COMMUNICATION STANDARD

COMMANDS	COMMAND DATA (FROM INSTRUMENT TO SENSOR)	SENSOR RESPONSE
Compliance Standard Set_Sen_Cmpl_Std CMD 0x8C	The compliance standard for LEL sensors can be changed from EN 50054 to EN 60079-20-1. [0]= Sensor index [1]= Compliance standard index 0x00 = EN 50054 (Default compliance standard) 0x01 = EN 60079-20-1 **Note:** When the sensor is reset or powered off the compliance standard returns to its default value, i.e. EN 50054	
User factor index SET_SEN_UF_INDEX CMD 0x8D	This function tells the sensor what instrument model it is in (by setting a predetermined index value). The user factor index will be different from one model to another because the airpath of the gas and/or used membrane used in the instrument may be different. This commands allows the sensor to identify the instrument in order to compensate for different gas dynamics. **Important:** Consult start-up chart in section 4 for additional information. [0] = Sensor index [1] = User factor index (UF) **Note:** The user factor reverts to its default value (UF0 =100) after the sensor is reset of powered off. **An example can be found in the supplementary documentation (Appendix II: Data decoding examples, i.5)*	
Gas unit SET_SEN_GASUNIT CMD 0x8E	Set the gas reading unit for sensor (it will depend on available unit list, use CMD 0x54 to obtain the gas list) [0]=Sensor index [1]=Gas reading unit 0x00: **ppm** (parts per million) 0x01: **%** (gas percentage) 0x02: **ppb** (parts per billion) 0x27: **% LEL** (lowest explosive limit) 0x28: **% VOL** (volume percentage)	
Calibration due days SET_CAL_INTERVAL_DAYS CMD 0x8F	Set the period for the countdown timer for the next calibration. [0] = Sensor index [1-2] = Due days for calibration e.g. 90 days = 0x54	
Bump test due days SET_BUMP_INTERVAL_DAYS CMD 0x90	Set the period for the countdown timer corresponding to the next bump test. [0] = Sensor index [1-2] = Due days for bump test e.g. 1 day = 0x01	
Set bump time SET_BUMP_TIME CMD 0x91	Reset the countdown timer of bump due days. **Note:** This function assumes that the bump test has been done (an operation must be implemented at the instrument level). The user should perform a bump test and confirm that the sensor has passed the test before using this command. [0]: Year (0*-99) *Where 0 stands for 2000 [1]: Month (1-12) [2]: Day (1-31) [3]: Hour (0-23) [4]: Minute (0-59)	

COMMUNICATION PROTOCOL
SDCS SMART DEVICE COMMUNICATION STANDARD

COMMANDS	COMMAND DATA (FROM INSTRUMENT TO SENSOR)	SENSOR RESPONSE
Set sensor accuracy SET_EC_ ACCURACY CMD 0x92	Set the accuracy of the electrochemical sensor. The smaller the percent accuracy, the shorter the period required to perform span and zero calibrations. [0]= Sensor index [1-2]= Percent accuracy * 100 **Accuracy (%) = (HEX to DEC)/100** Note: Command expressed in Hexadecimal. Where **Accuracy (%) = (HEX to DEC)/100** e.g. 0x03 E8 0x03 E8= 1000 1000/100= 10 % accuracy **Note:** This command is not valid for LEL sensors	
Clear STEL/TWA counter CLEAR_STEL/ TWA_COUNT CMD 0x93	Resets STEL and/or TWA internal counter. This command should be used when the instrument is passed from one individual to another after either a 15-minute period for STEL or an 8-hour working shift for TWA. [0]= Counter to be cleared B7-B3: Reserved B1: TWA B0: STEL e.g. 0x03 will clear both parameters (TWA & STEL) [1]= Request B2-B7: Reserved B1: Clear counter B0: Reserved	
User calibration USER_CAL CMD 0xA1	**This process consists of various sequential steps that needs to be completed to complete the calibration of the sensor.** **Consult calibration flow chart in Section 6 of this document for a visual reference.** **A few examples can be found in the supplementary documentation (Appendix II: Data decoding examples: vi.1 & vi.2)*	
	Step 1 [0] = Sensor Index. For example, a COSH sensor has two target gases: CO & H_2S 0x0003= 0b0011 (i.e. sensors in B0 & B1) [1]= BitMap 0x00: Disabled 0x01: Enabled [2] = Calibration type 0x00, Zero 0x01, Span 0x02, Span high (for sensors that require more than one span calibration point) 3-255, Reserved [3] = Operation = **0x80 (Prepare for calibration)** **Note:** The calibration countdown begins after this point	

COMMANDS	COMMAND DATA (FROM INSTRUMENT TO SENSOR)	SENSOR RESPONSE
	Step 2*: Aborts the calibration and finishes process **The process can be made at any point during the calibration countdown time.* [0] = Sensor Index. [1]= BitMap 0x00: Disabled 0x01: Enabled [2] = Calibration type 0x00, Zero 0x01, Span 0x02, Span high (for sensors that require more than one span calibration point) 3-255, Reserved [3] = Operation = **0x81 (abort calibration)** **Note:** After sending this command, the calibration process ends.	
	Step 2: Proceeds to calibrate and record data of calibration [0] = Sensor Index. [1]= BitMap 0x00: Disabled 0x01: Enabled [2] = Calibration type 0x00, Zero 0x01, Span 0x02, Span high (for sensors that require more than one span calibration point) 3-255, Reserved [3] = Operation = **0x00 (Start user calibration)** **Note:** During this process the date, temperature and calibration points will be recorded in sensor.	[0-1] – Process cost in milliseconds (Hexadecimal format) e.g. 0x320=800ms
	Step 3: Request calibration result [0] = Sensor Index. [1]= BitMap 0x00: Disabled 0x01: Enabled [2] = Calibration type 0x00, Zero 0x01, Span 0x02, Span high (for sensors that require more than one span calibration point) 3-255, Reserved [3] = Operation= **0x83 (Get calibration result)**	[0] Sensor Index [1] Calibration result: 0x00 = Failed cal. 0x01 = Successful cal.

2.3 ALOHA MODE

The Aloha mode can be configured to send data packs constantly after a certain period and/or when a gas threshold has been reached. Once the Aloha mode is enabled, it will transmit data packs without request of the instrument.

Important: This mode is only available when interfacing with a single sensor.

2.3.1 GET ALOHA CONFIGURATION

The following command is used to request the Aloha configuration:

Table 8. Get Aloha Commands

COMMANDS	COMMAND DATA (INSTRUMENT)	RESPONSE DATA (SENSOR)
Aloha mode GET_ALOHA_ MODE CMD 0x53	Some iseries devices can measure more than one gas type. This command will specify which gas is the one that is being measured and it will correspond to its sensor index number. [0] Sensor Index	Displays whether the sensor is in aloha mode (by period and/or gas threshold) or not. [0] Mode: B0: Period B1: Gas threshold B2-7: Reserved **Note:** Case 1: [0] = 0x00 (Aloha mode is inactive) There are no following bytes Case 2: [0] = 0x01 (Aloha mode by period) [1-2] Period (in seconds) Case 3: [0] = 0x02 (Aloha mode by threshold) [1-4] Gas threshold (Gas reading format, i.e. (HEX to DEC)/100) Case 4: [0] = 0x03 (Aloha mode by period and threshold) [1-2] = Period (in seconds) [3-6] = Gas threshold (Gas reading format, i.e. (HEX to DEC)/100)

**An example can be found in the supplementary documentation (Appendix II: Data decoding examples, iv.2)*

2.3.2 SET ALOHA MODE

Table 9. Set Aloha Commands

COMMAND	COMMAND DATA (INSTRUMENT)	RESPONSE DATA (SENSOR)
Aloha configuration ALOHA_CONFIG CMD 0xA2	**Different configurations cases can be presented:** **Case 1:** Neither period nor gas threshold [0] = Sensor index [1]: Mode = **0x00** B0: Aloha by period 0x00 = Disabled B1: Aloha by gas threshold 0x00 = Disabled B2 to B15: Reserved	
	Case 2: Aloha by period [0] = Sensor index [1]: Mode= **0x01** B0: Aloha by period 0x01 = Enabled B1: Aloha by gas threshold 0x00 = Disabled B2 to B15: Reserved [2-3]: Period between gas measurements in seconds. The value must be ≥1	
	Case 3: Aloha by gas threshold [0] = Sensor index [1]: Mode = **0x02** B0: Aloha by period 0x00 = Disabled B1: Aloha by gas threshold 0x01 = Enabled B2 to B15: Reserved [2-5]: gas threshold in gas reading format ((HEX to DEC)/100)	
	Case 4: Aloha by period and gas threshold [0] = Sensor index [1]: Mode B0: Aloha by period 0x01 = Enabled B1: Aloha by gas threshold 0x01 = Enabled B2 to B15: Reserved [2-3]: Period between gas measurements in seconds. The value must be ≥1 [4-7]: Gas threshold in gas reading format (i.e. (HEX to DEC)/100)	

**An example can be found in the supplementary documentation (Appendix II: Data decoding examples, iv.1)*

2.3.3 ALOHA RESPONSE

Once the aloha mode is set, the response of the sensor will be an aloha data pack:

Table 10. Response Aloha Commands

COMMAND	COMMAND DATA (INSTRUMENT)	RESPONSE DATA (SENSOR)
Aloha data pack ALOHA_DATA_PACK CMD 0xA3	This command is the equivalent of get data pack (CMD 0x30) and is sent automatically by the sensor when the period and/or the gas threshold is reached. The instrument doesn't need to send any command after setting up aloha mode. (Consult CMD 0xA2: ALOHA_CONFIG) **Important:** If any command other than Get_aloha_mode (0x53) is used while the sensor is already in aloha mode, the aloha configuration will be annulled and will return to its default value. i.e. Aloha by threshold and aloha by period are not enabled by default.	[0] Sensor index [1] Status: B0: Reserved B1: Warming up B2: Reserved B3: In calibration B6: In sleep mode B4-B7: Reserved [2] Alarm: B0: Over range B1: User factor hasn't been set B2: Time is not synch (RTC hasn't been set) B3: High alarm B4: Low alarm B5: STEL alarm B6: TWA alarm B7: Drift (flagged when the apparent gas concentration reading has gone too far negative) [3-n]= Error code: Byte 0: Number of error codes Byte 1-n: Error codes (1 byte each error code) Note: n only shows up if there is an error List of error codes (**NOT** in order of priority): Error code 001 = Diagnostic electrode failure Error code 101 = Sensing electrode impedance too high Error code 102 = Reference electrode failure Error code 103 = Electrolyte too dry Error code 104 = End of life Error code 105 = Counter electrode failure Error code 106 = Broken bead / short circuit (LEL) Error code 108 = LED/PD Failure (NDIR) Error code 109 = Span calibration is due Error code 110 = Bump test is due Error code 111 = User Factor not valid Error code 112 = Operational temp out of range Error code 113 = Electrolyte too wet *The order or priority is found in troubleshooting appendix.* [n+1...n+4] = Gas reading: concentration measurement Gas reading is 4 bytes **unsigned integer**. Gas measurement= Hex to Dec/100

**An example can be found in the supplementary documentation (Appendix II: Data decoding examples, iv.3)*

3. ERRORS

If the sensor receives a command packet, but cannot perform the requested command or cannot prepare the information. It will return an error packet indicating the reason of the failure.

The error packet will have the following format, where **0x71** is the command that indicates that there was an error in the request.

TABLE 11. ERROR PACKET

ITEM	SOP	VER-SION	LENGTH	AI	CMD	DATA	CRC	EOP
Size (Bytes)	1	1	1	2	1	1	2	1
Content	0x7B	0x59	Number of bytes from CMD to EOP	Auto-index	**0x71**	Error code, see Table 12	CRC-16: Count from SOP to data	0x7D

TABLE 12. LIST OF ERRORS

ERROR CODE	RETURN CODE NAME	DESCRIPTION
0x31	FAIL_UNKNOWN	Unknown error
0x32	FAIL_INVALIDCMD	The command is not supported by this sensor
0x33	FAIL_DATASIZE	The command is supported by this sensor, but the data in the command packet is not valid
0x34	FAIL_ INVALIDVALUE	The value to be set is not valid
0x39	FAIL_ WRITEPROTECT	The value to be set is protected, cannot be changed without deactivating write protect
0x3A	FAIL_SLEEP	The operation is not supported when the sensor is in sleep mode
0x3F	FAIL_OPERATION	The operation wasn't executed successfully

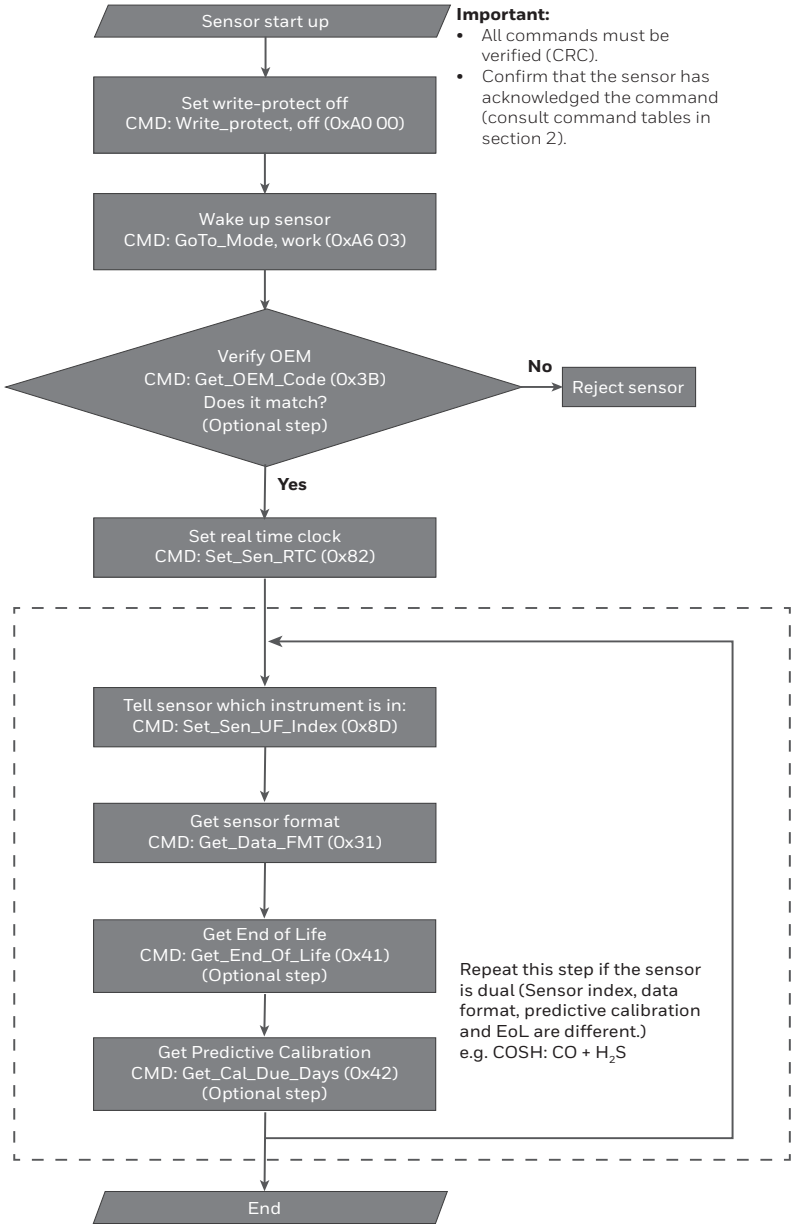
**An example can be found in the supplementary documentation (Appendix II: Data decoding examples, vi.3)*

4. START UP

To initiate the sensor and obtain gas readings, it is necessary to follow the procedure described in the following initiation flow-chart. When the sensor is firstly powered-on, it will take five seconds for the device to initialise peripheries and load configuration files; once the process is finished, it will automatically go to sleep mode. To wake up the sensor, it is necessary to set Write-Protect off and use GoTo_Mode. As described in the diagram, the OEM code must match, so the sensor can be used in the instrument (otherwise the device is rejected and will not be able to be used by the instrument).

Note: OEM lock check is optional.

Figure 6. Sensor Start-up Flow-Chart



After following all the steps, it is necessary to wait for the sensor to stabilize. The state of the sensor can be verified by asking the status of the sensor (CMD 0x30).

Note: If the OEM modifies the instrument design in a way that affects the correction factor, it is considered to be a new instrument and is given a new reference number. When a new instrument is released, new correction factors need to be generated for all sensors that the OEM uses.

5. GETTING GAS CONCENTRATION READINGS FROM SENSOR

The following flow chart exemplifies the common procedure performed by the sensor to obtain a gas measurement. **Bear in mind, the gas concentration measurement won't be accurate until the sensor is stable (the stabilisation period will depend on the type of sensor).**

Figure 7. Obtaining Gas Reading

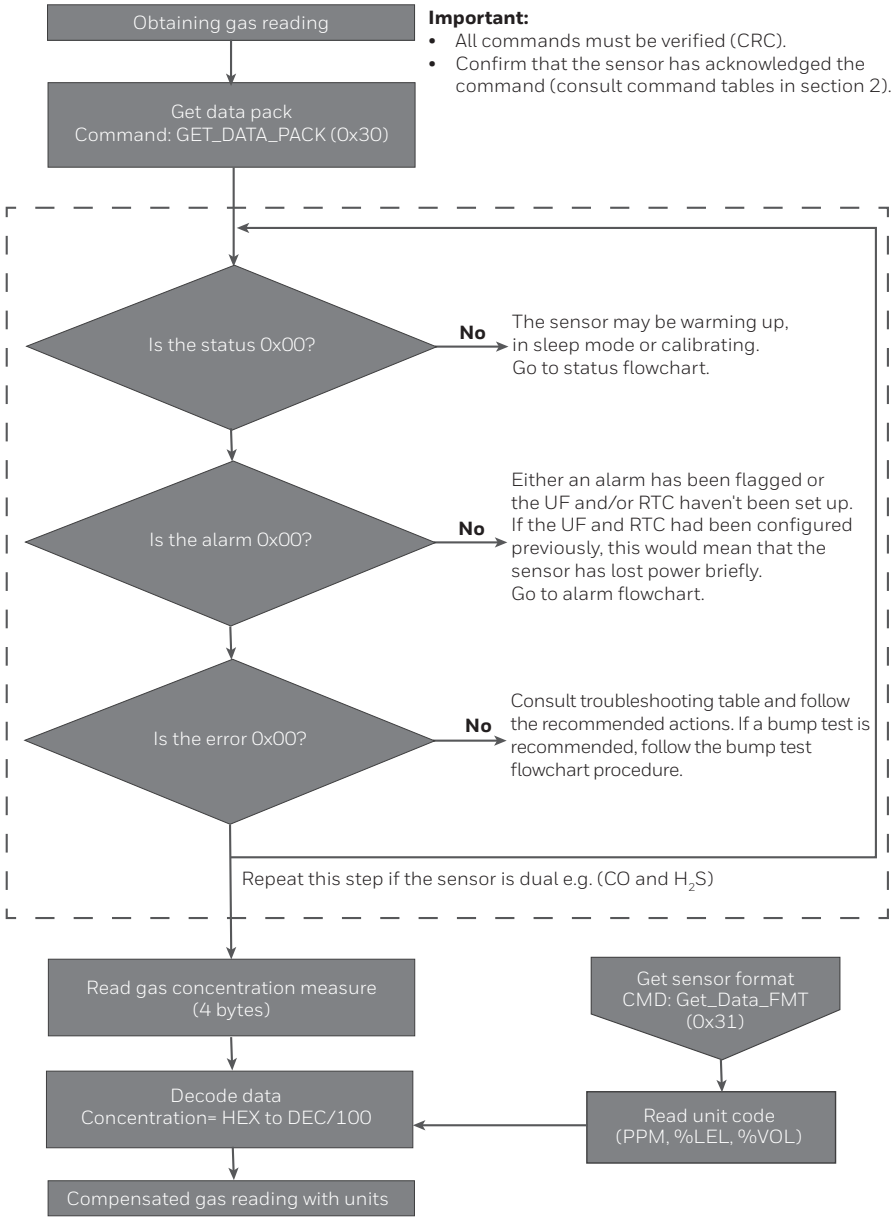


Figure 8. Status

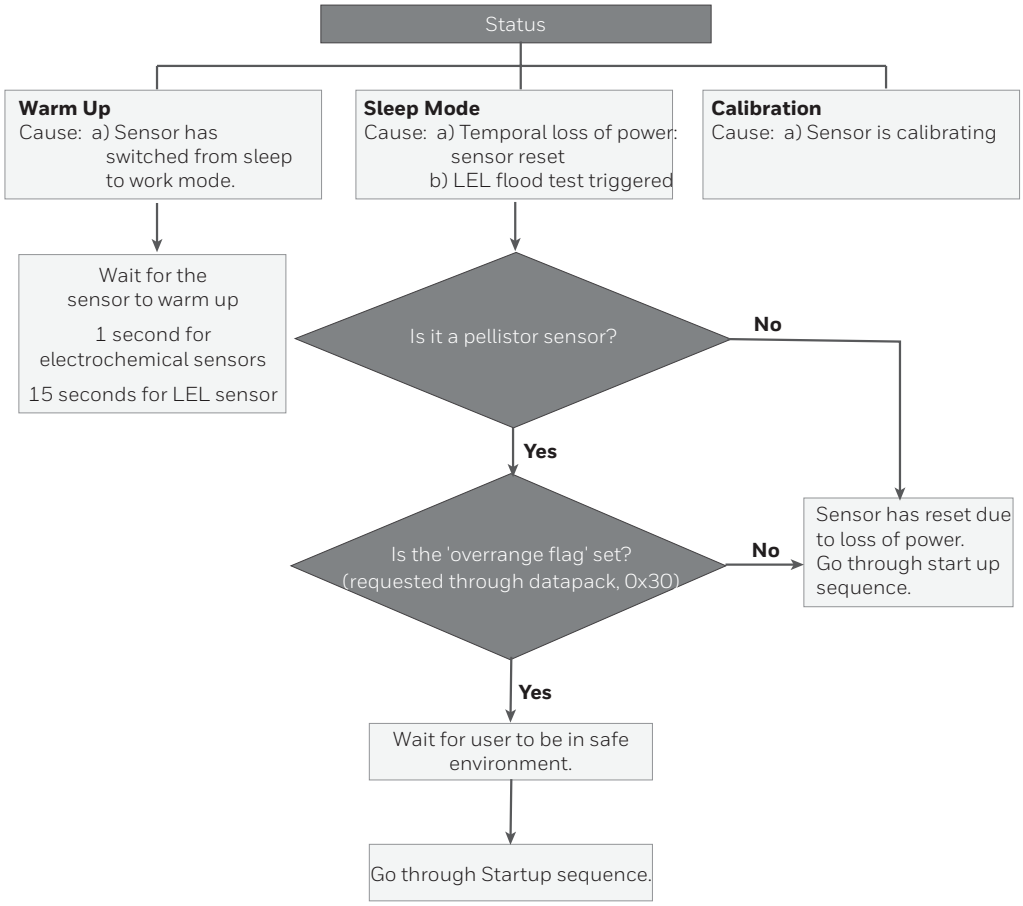


Figure 9. Alarm

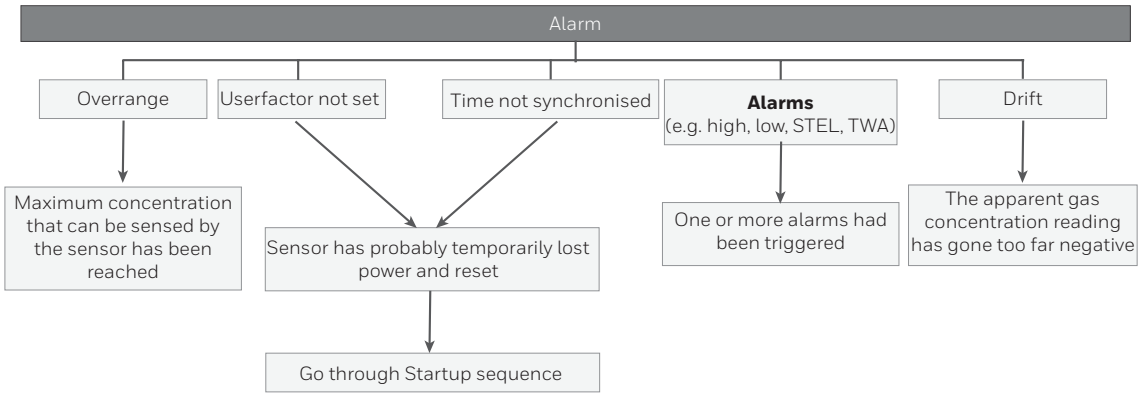
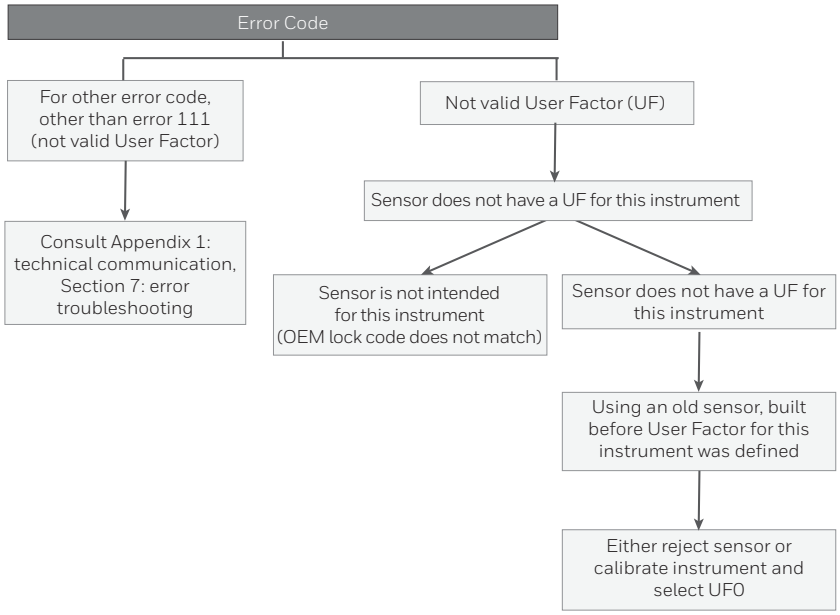
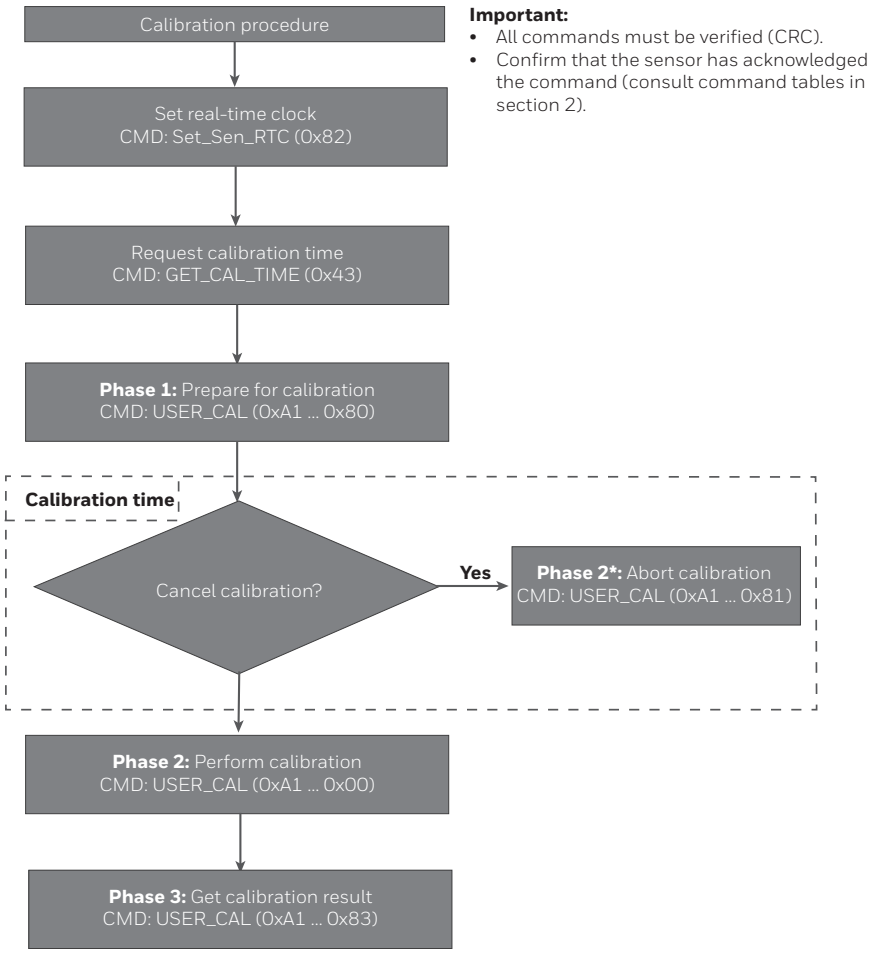


Figure 10. Error Codes



6. CALIBRATION DATA

Figure 11. Calibration Flow Chart



Important:
- All commands must be verified (CRC).
- Confirm that the sensor has acknowledged the command (consult command tables in section 2).

7. BUMP TEST

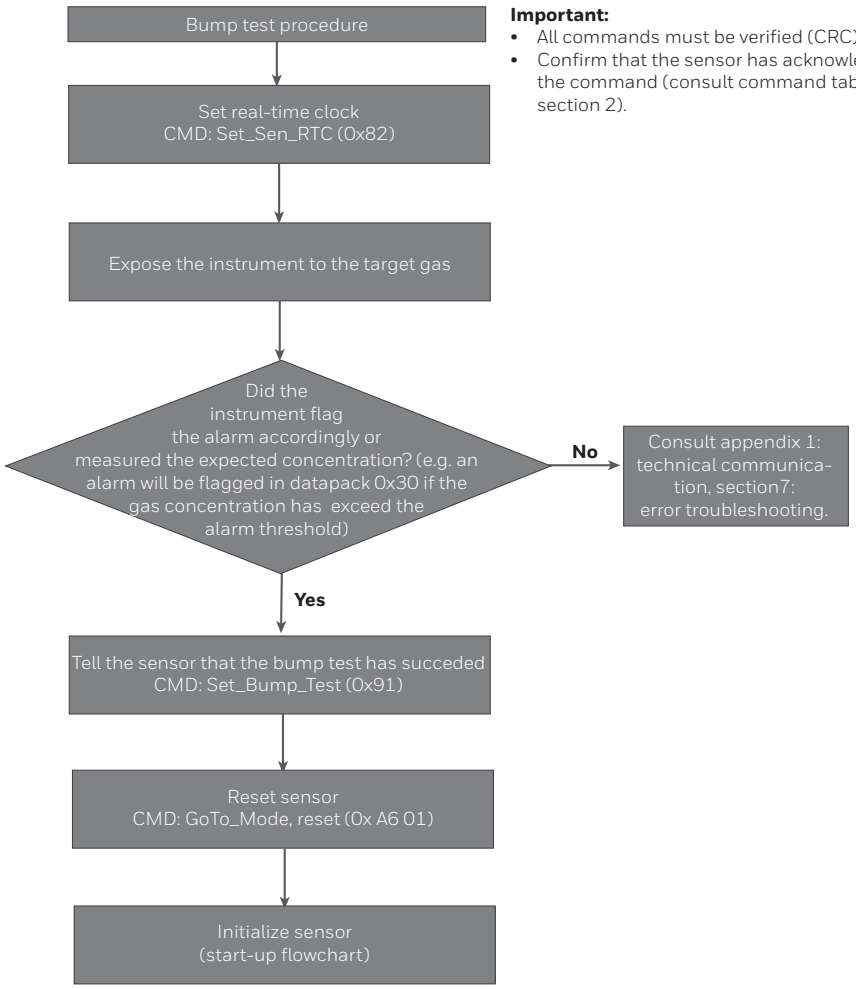
Honeywell recommends users of portable gas detectors containing electrochemical or catalytic bead sensors conduct a daily 'bump' check before relying on the unit to verify an atmosphere is free from hazard.

A 'bump' test is a means of verifying that an instrument is working within acceptable limits by briefly exposing to a known gas mixture formulated to change the output of all the sensors present. This is different from a calibration where the instrument is also exposed to a known gas mixture but is allowed to settle to a steady figure and the reading adjusted to the stated gas concentration of the test mixture.

For oxygen monitors a level of confidence that the unit is working adequately may be gained by exhaling over the sensor inlet and viewing the reduction in reading obtained. In many cases the reduction in Oxygen concentration obtained is sufficient to trigger the low oxygen alarm.

Please bear in mind that the bump test does not account to measure the gas sensor accuracy. The main intent is to verify the functionality of the gas sensing system. Then, it is recommended to use a gas concentration high enough to trigger the sensor alarm. To configure and read the low and high concentration alarms, refer to: GET_SEN_PARA (0x33) & SET_SEN_PARA (0x80).

Figure 12. Bump Test Flow Chart



Communication Protocol

APPENDIX 1: TECHNICAL COMMUNICATION

1.0 POWER CONSUMPTION: SLEEP MODE

When the instrument is turned off, it should send a command that enables the sensor to sleep, wake up or reset the sensor. By using the "go to" mode (GOTO_MODE: 0xA6), you can command the sensor to go to these different operating modes.

Note that **the iseries sensors should be powered continuously while they are in an instrument.** Failing to do so will automatically reset parameters such as the real-time clock, User factor, et cetera. Additionally, cutting the power of sensor will increase stabilization times.

During the sleep mode, the sensor **cannot provide gas concentrations or temperature readings;** however, it will **considerably save power** and keep the communication between the instrument and sensor.

In sleep mode, the iseries sensors will take different actions, depending on the type of sensor that is being used:

- For some toxic electrochemical sensors (such as CO, H_2S and SO_2), the sensor will remain powered but it will not be taking any measurements; during this period, there will be a residual current of approximately
  8.5 uA.
- For LEL sensors, the pellistor will turn off power to bead and wait for communication from instrument to wake the sensor up.
- For the oxygen sensor, the pump stays powered to have a quick start-up. After powering the sensor for approximately 60 minutes, the sensor will start taking accurate measurements and no additional time will be required (even after the sensors goes from sleep to work mode).
  - The oxygen pump consumer a continuous current of 100 uA.

2. WAKE UP TIME

On power-up, the iseries sensor can receive the first command after 5000 ms from when the V_{DD} supply is within operating specifications. The iseries sensor can begin to communicate after 5000 ms from when the V_{DD} supply is operational.

However, the sensor will not be able to accurately measure gas concentrations until the sensor is stable (this will depend on the type of sensor). GET_DATA_PACK command can consult the sensor status and validate measurements.

3. TIMEOUT

Once the instrument has sent a command, the sensor should respond within a 250 ms timeframe. If the sensor is unable to answer back after this period, a time-out will be accounted.

If the sensor does not answer within this timeframe on three consecutive attempts, the sensor will be accounted as if it was offline.

4. DEAD BAND

Dead band is a region where a change in gas concentration produces no variation in the sensor measurement output. When the function is enabled, the outgoing and incoming parameters can be configured based on the user's requirements.

4.1 DEADBAND FOR TOXIC SENSORS

For example, assume the dead band has been configured with the following parameters in a toxic sensor:

- Outgoing concentration = 3 ppm
- Incoming concentration = 2 ppm

When the gas concentration is below the outgoing limit, the sensor's gas reading is 0 ppm; once this outgoing limit is exceeded, the actual gas readings start to be displayed. As the gas concentration falls, it will only read zero once it has fallen below the incoming limit.

Note: Incoming limit < outgoing limit

Figure 13. Gas Concentration in Sensing System

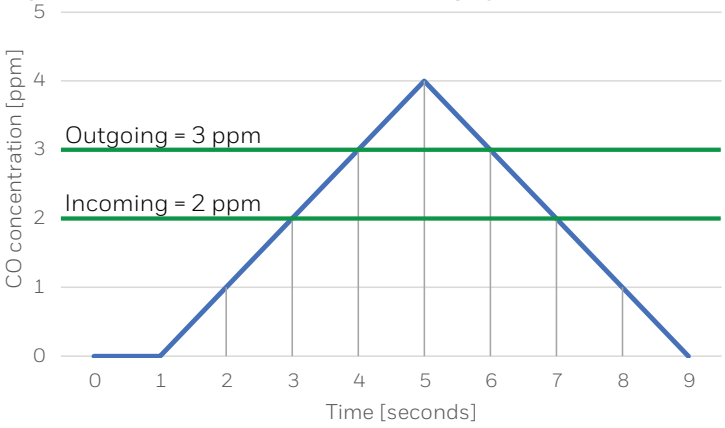
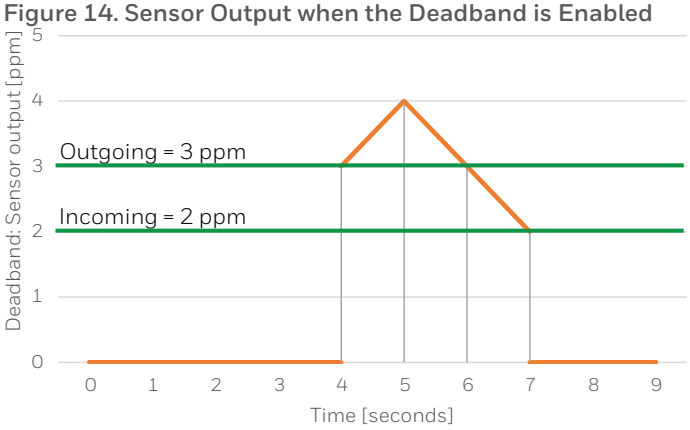


Figure 14. Sensor Output when the Deadband is Enabled



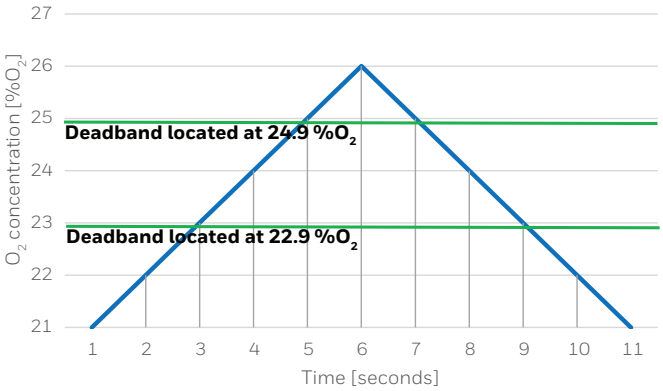
4.2 DEADBAND FOR OXYGEN SENSORS

For the case of oxygen (where the baseline is 20.9 $\%O_2$ when the sensor is in clean ambient air), the dead band works in the same way but in **positive and negative directions simultaneously**. For instance, in the following case consider an incoming and outgoing values of 2 and 4 correspondingly.

In the first case, we will suppose that the the sensor is initially in clean air, then the oxygen concentration is increased, and afterwards it measures ambient air again. In the second case, the sensor is exposed to clean air, then the oxygen is depleted and finally it goes back to its initial state (clean air).

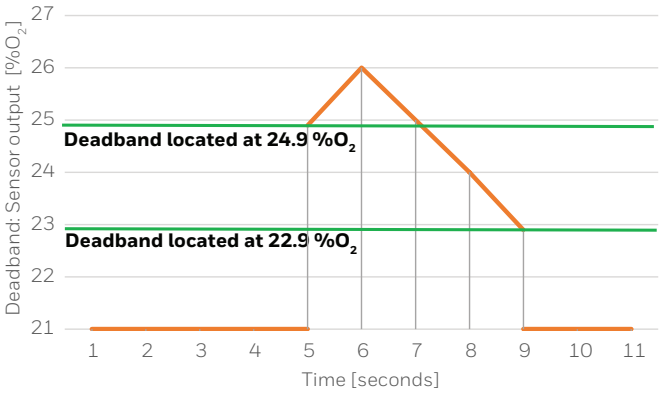
Note: Please note that the oxygen sensor will activate all the deadband limits when this function is enabled (positive and negative). However for demonstrative purposes, the deadband limits are represented as if they were two different cases.

Figure 15: Case 1: Oxygen Concentration in Sensing System is Increasing



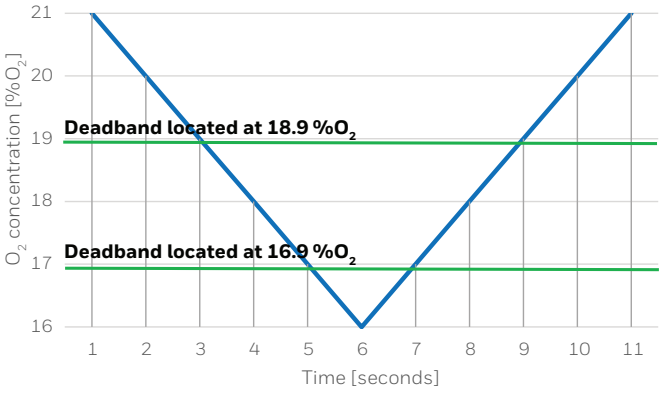
Ambient air = 20.9% O_2; Incoming=2; Outgoing=4;
Positive outgoing deadband located at 24.9 $\%O_2$ (ambient air + outgoing value)
Positive incoming deadband located at 22.9 $\%O_2$ (ambient air + incoming value)

Figure 16: Sensor Output When the Deadband is Enabled



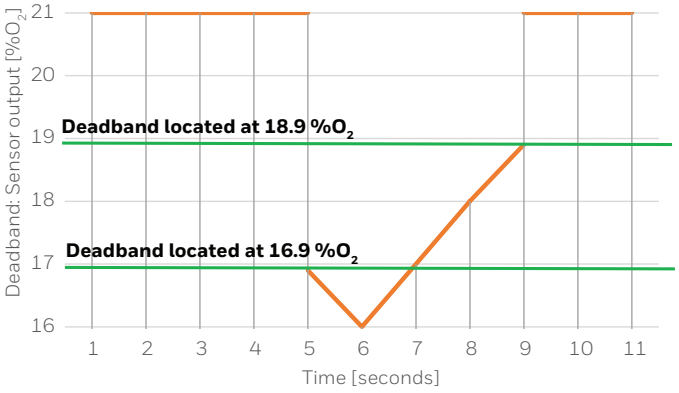
Ambient air = 20.9% O_2; Incoming=2; Outgoing=4;
Positive outgoing deadband located at 24.9 $\%O_2$ (ambient air + outgoing value)
Positive incoming deadband located at 22.9 $\%O_2$ (ambient air + incoming value)

Figure 17. Case 2: Oxygen Concentration in Sensing System is Decreasing



Ambient air = 20.9%O_2; Incoming=2; Outgoing=4;
Negative incoming deadband located at 18.9 %O_2 (ambient air - incoming value)
Negative outgoing deadband located at 16.9 %O_2 (ambient air - outgoing value)

Figure 18. Sensor Output When the Deadband is Enabled



Ambient air = 20.9%O_2; Incoming=2; Outgoing=4;
Negative incoming deadband located at 18.9 %O_2 (ambient air - incoming value)
Negative outgoing deadband located at 16.9 %O_2 (ambient air - outgoing value)

5. INTERNAL TESTS

The iseries sensors can detect several internal faults using on board diagnostics:

- Sensor lost PCBa contact (measured by Reflex Text)
- Sensing electrode impedance too high (measured by diagnostic electrode)
- Reference electrode failure (measured by diagnostic electrode)
- Electrolyte too dry or too wet (measured by diagnostic electrode)
- Counter electrode failure (from counter polarisation measurement)
- Broken bead or short circuit resistance measurement in Pellistor
- Diagnostic electrode failure (diagnostic electrode fails to produce valid data)
- LED/PD failure (signal measurement in NDIR)
- The predictive calibration algorithm flags an error when when its accuracy is becoming too poor to give a reliable accurate reading or when the calibration countdown reaches 0. The output of the predictive calibration algorithm is calculated using using time, temperature, accuracy and electrolyte concentration as inputs (taken from the diagnostic test).
- End of life flags an error when its sensitivity is falling too low to give a reliable accurate reading or when the sensor has taken/lost a considerable amount of water. The output of the EoL algorithm is calculated using using time, temperature, sensitivity and electrolyte concentration as inputs (taken from the diagnostic test).

The following sections describe the logic and times at which the sensors perform such tests.

5.1 COUNTER ELECTRODE AND REFLEX TEST (ELECTROCHEMICAL SENSORS)

The counter electrode (CE) and and reflex test are done periodically regardless of the mode at which the sensor is operating (sleep/work mode). The CE and reflex text run **automatically** once an hour. It important to emphasize that since these functions are trigged after this period has elapsed it is essential to configure the RTC right after powering on the sensor (as stated in the start-up flowchart in section 4 of SDCS). The next flowchart and timeline shows the times that are required for the sensor in order to perform the counter electrode and reflex test.

The Counter Electrode (CE) and reflex test are done periodically regardless of the mode at which the sensor is operating (sleep/work mode).

The CE and reflex text run **automatically** once an hour. It important to emphasize that since these functions are trigged after this period has elapsed it is essential to configure the RTC right after powering on the sensor (as stated in the start-up flowchart in section 4 of SDCS).

The next flowchart and timeline shows the times that are required for the sensor in order to perform the counter electrode and reflex test.

Figure 19. Counter Electrode and Reflex Test Flowchart

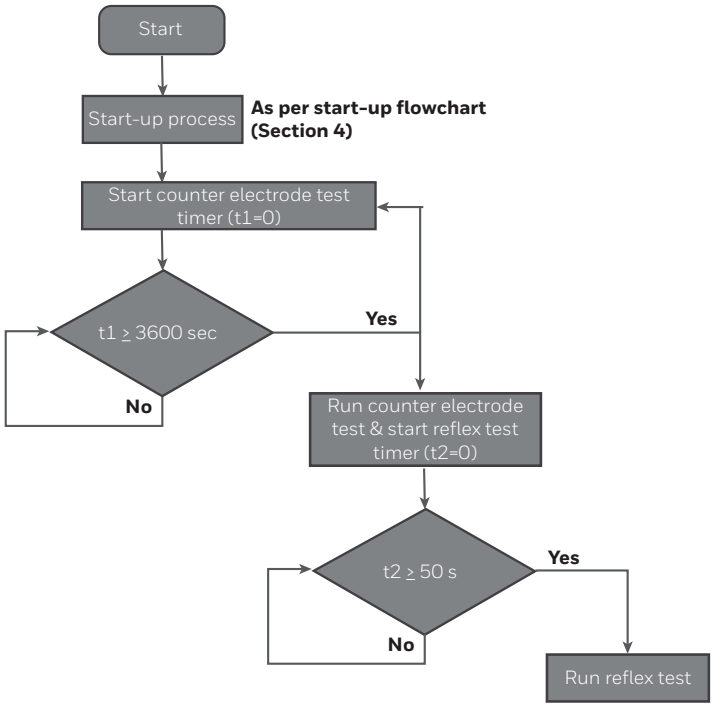
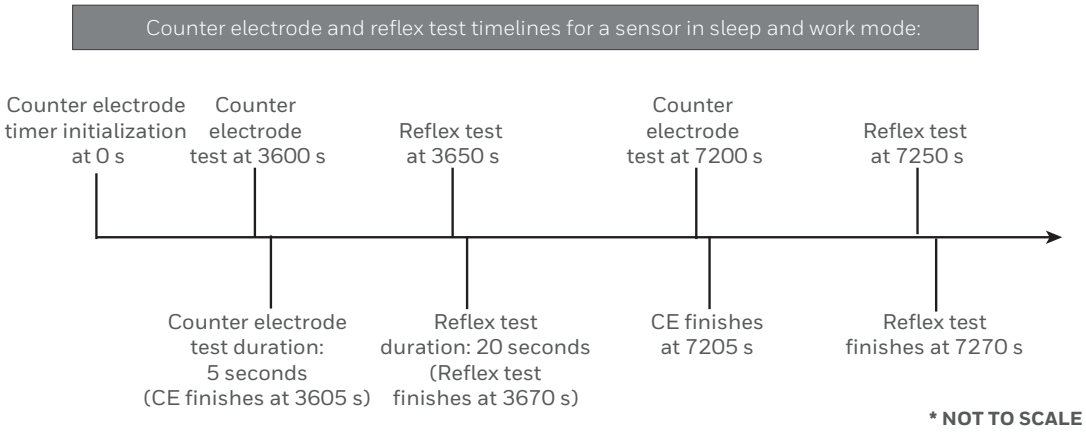


Figure 20. Counter Electrode and Reflex Test Timeline



5.2 DIAGNOSTIC TEST (ELECTROCHEMICAL)

The electrochemical iseries sensor can predict in advance when the accuracy is becoming too poor to give a reliable and accurate reading; in addition, it can also predict when its sensitivity is falling too low to give reliable and accurate readings.

The non-recoverable drift and the poor accuracy can be estimated by the End-of-Life and Predictive Calibration functions respectively. To calculate the estimated times of both functions, it is necessary to run a diagnostic test. It is important to mention that this test is **only performed when the sensor is in sleep mode**, so it is highly recommended to change the sensor to sleep mode whenever the sensor is not in use (otherwise the End-of-Life and Predictive Calibration estimations will not be updated/recalculated, leading to non-accurate results).

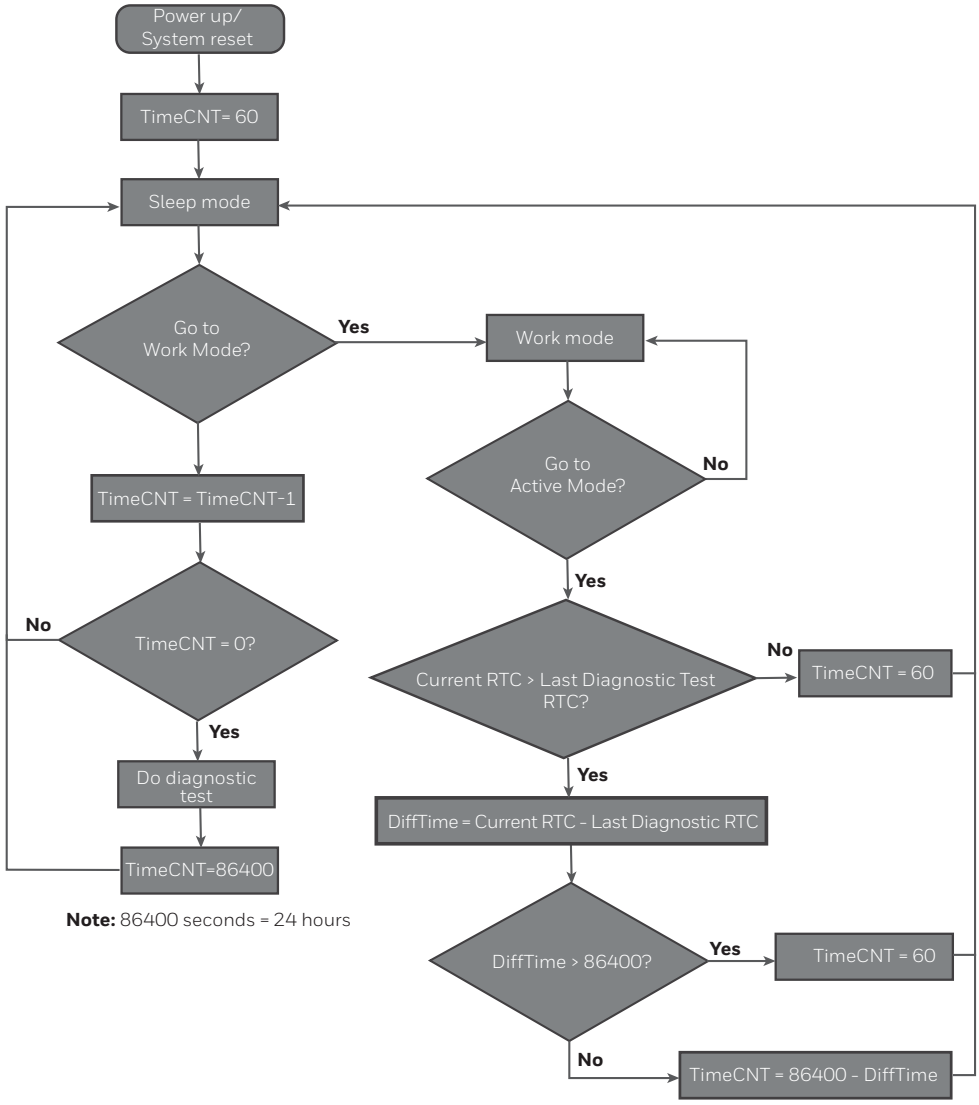
COMMUNICATION PROTOCOL
APPENDIX 1: TECHNICAL COMMUNICATION

The next flowchart and timeline shows the times that are required for the sensor in order to perform the diagnostic.

Note that no action is required from the user in order to run the diagnostic test. The test will run automatically after the sensor is switched to sleep mode.

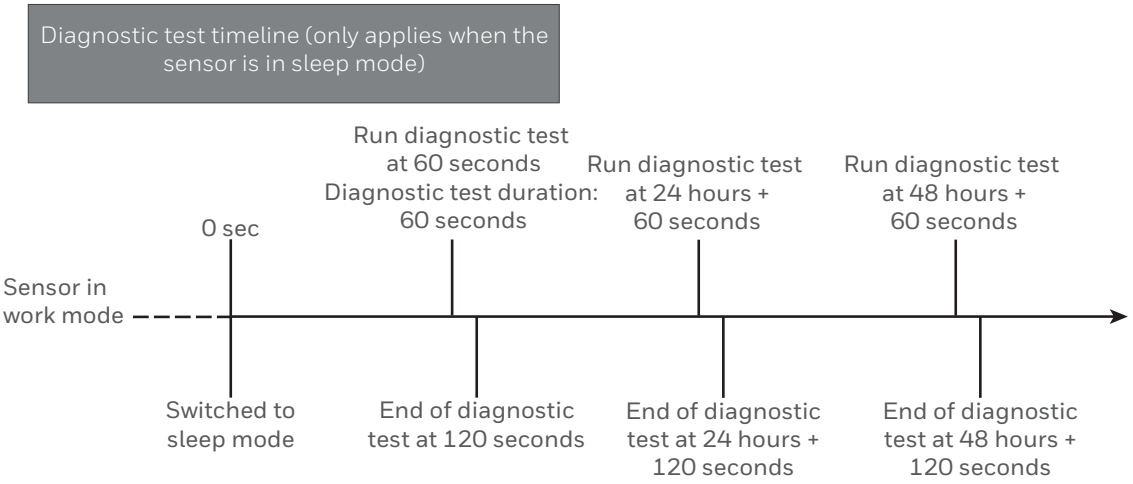
The next flowchart shows the timings required for the sensor to perform a diagnostic test.

Figure 21. Diagnostic Test Flowchart



Note: 86400 seconds = 24 hours

Figure 22: Diagnostic Test Flowchart Timeline



***NOT TO SCALE.**
Note: The diagnostic test is performed ONLY when the sensor is in sleep mode.

6. TYPICAL CURRENT DURING START-UP AND AFTER SENDING COMMANDS:

The next figures show the typical currents of electrochemical sensors after powering the sensors and performing an automatic diagnostic test.

For all cases, the sensors haven't run a diagnostic test in the last 24 hours. See section 5.2 to consult the frequency of the diagnostic test.

Figure 23. Oxygen Start-Up Currents

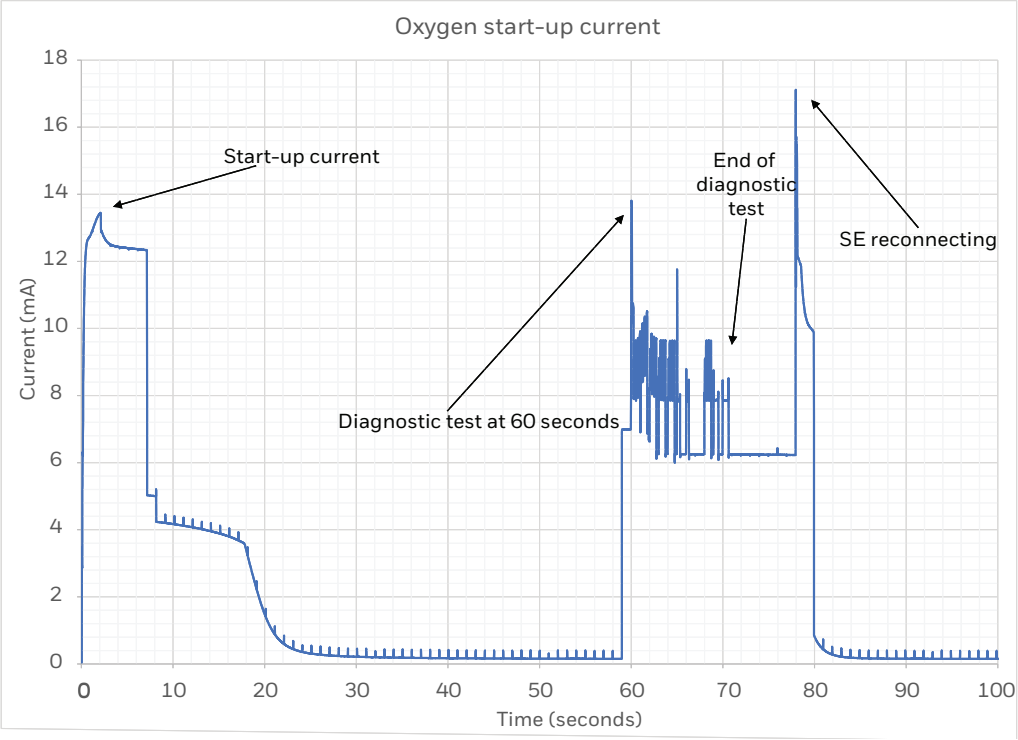


Figure 24. Oxygen Current in Work Mode

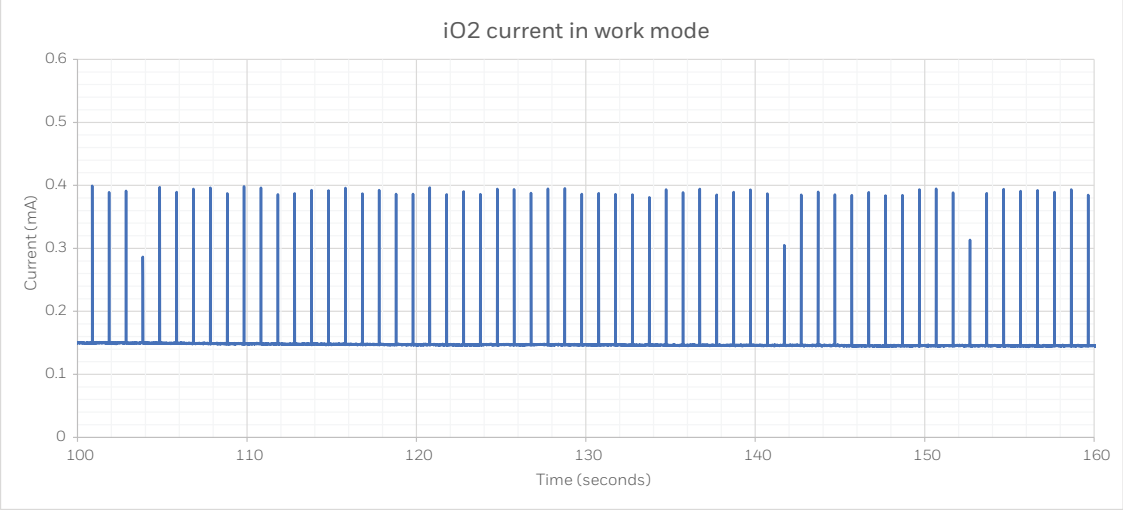


Figure 25. Oxygen Current in Work Mode

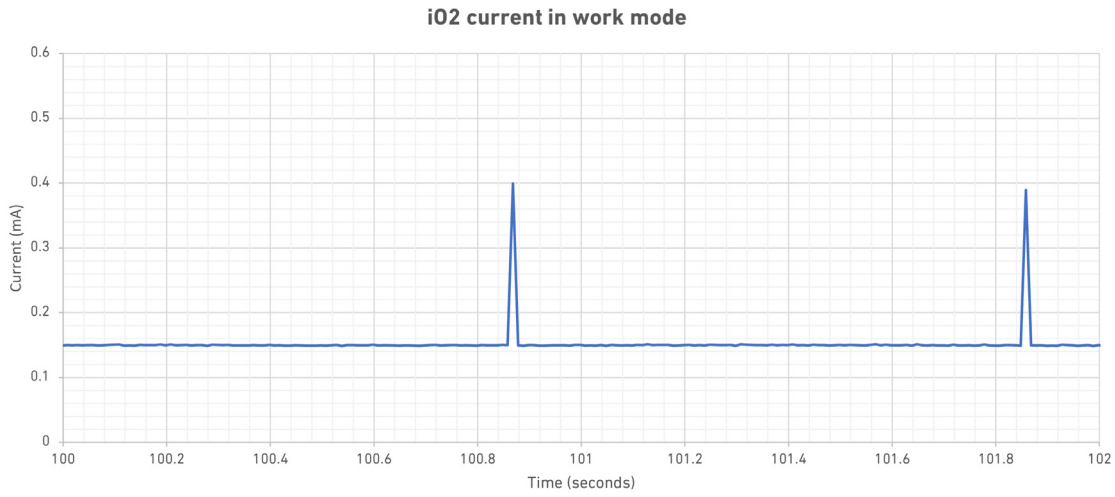


Figure 26. Hydrogen Sulfide Start-Up Currents

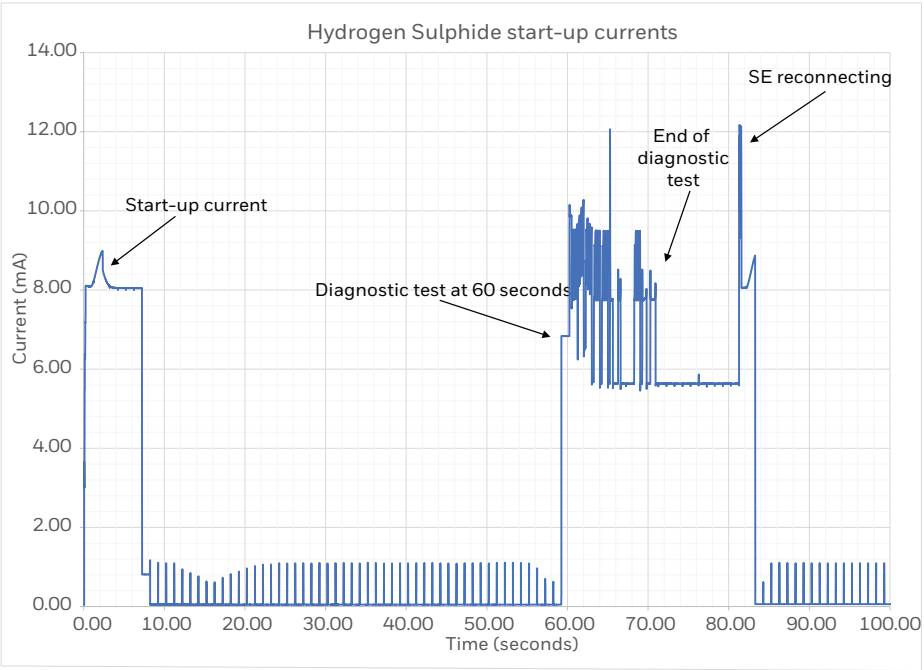
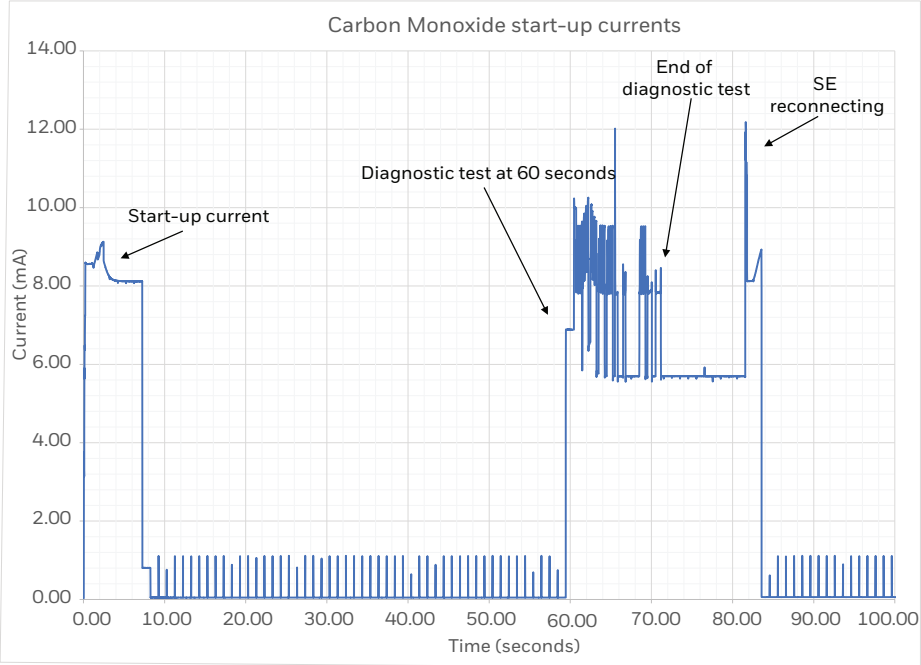
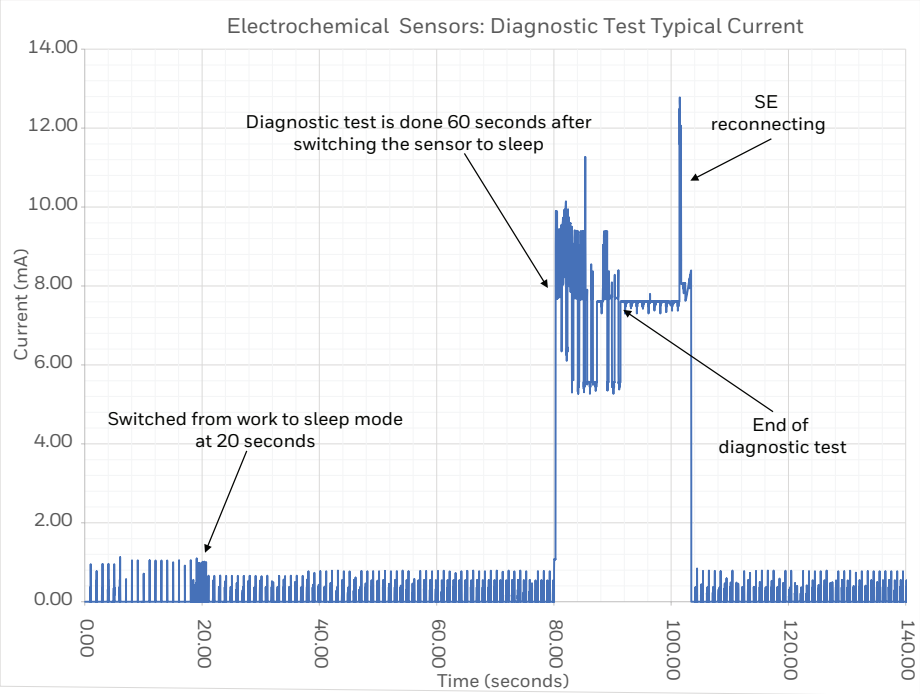


Figure 27. Carbon Monoxide Start-Up Currents



The next figure shows the typical current of an electrochemical sensor when performing a diagnostic test.

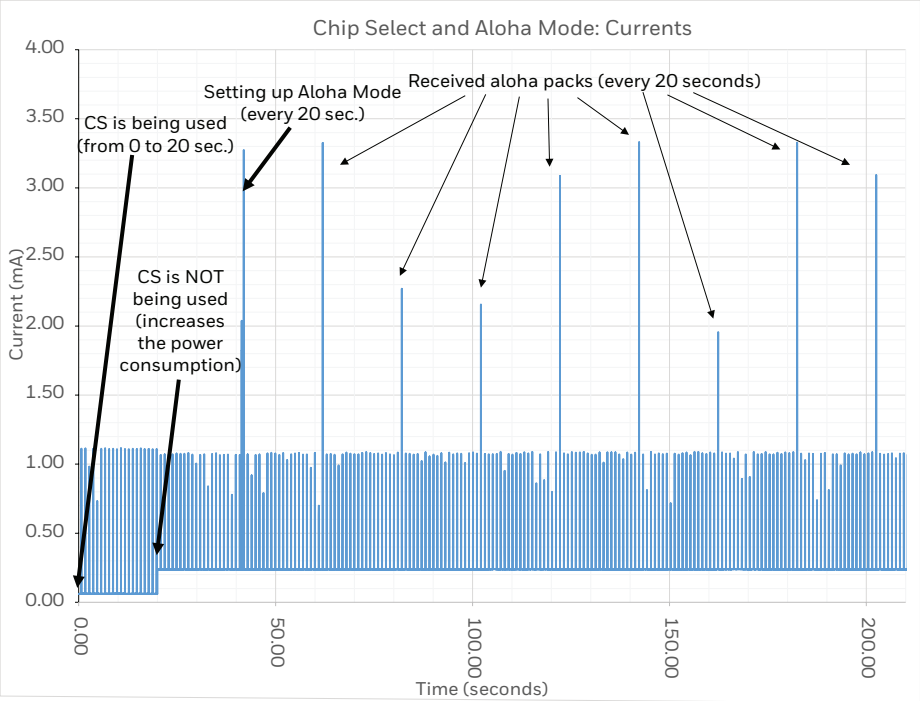
Figure 28. Electrochemical Sensors: Diagnostic Test, Typical Current



The next figure shows that there are different drawing currents when the sensor is being driven with and without Chip Select. **When Chip Select is being used the drawing current (and hence overall power consumption) is lower.**

The figure also shows the typical current values given by the sensor whenever it is sending a gas concentration measurement (by sending aloha data packs to the instrument).

Figure 29. Chip Select and Aloha Mode: Currents



7. ERROR TROUBLESHOOTING

This following table shows the list of errors in **order of priority** (how critical is the error):

TABLE 13. ERRORS

PRIORITY	ERROR CODE	SENSOR TYPE	ERROR DESCRIPTION	DETECTION METHOD	DESCRIPTION	ACTION
1	001	EC	Diagnostic electrode failure	Diagnostic electrode	Internal diagnostic electrode has failed or is unable to take a reading. This does not necessarily mean the sensor itself is not working, but diagnostics capability is compromised	Do a bump test* to verify that sensor is working properly. Replace sensor if bump test fails
2	101	EC	Sensing electrode impedance too high	Reflex test	A small electrical pulse (reflex test) is used to check if the impedance of the sensing electrode circuit is outside the expected range. May be due to mechanical loss of contact or electrode damage/degradation	Replace sensor
3	102	EC	Reference electrode failure	Diagnostic electrode	The electrochemical voltage of the reference electrode is compared with an internally generated standard. If the reference voltage has drifted out of range, the sensor performance may be compromised (slow response, low sensitivity, baseline offset, etc.)	Do a bump test* to verify that sensor is working properly. Replace sensor if bump test fails. Replace sensor if error code continues to reappear after resetting
4	103	EC	Electrolyte too dry	Diagnostic electrode	Sensor has lost too much water from its electrolyte by evaporation due to operation for too long in dry conditions. Sensor performance is likely to be degraded	Bump test* sensor to confirm if it is still working. Sensor can be put into a humid environment to help it recover. Replace if it does not recover. See characterization note for recovery times
5	104	EC	End of life	Diagnostic electrode	The sensitivity has decayed below its acceptable lower limit and the sensor is no longer usable, or it has exceeded its five-year life. Replace sensor	Replace sensor
6	105	EC	Counter electrode failure	Counter polarisation measurement	Counter electrode in sensor is overloaded or not working correctly. This may result in the sensor being unable to maintain correct operating conditions, causing performance degradation	Do a bump test* to verify that sensor is working properly. Replace sensor if bump test fails. Replace sensor if error code continues to reappear after resetting
7	106	LEL	Broken bead or short circuit	Resistance measurment	Resistance of the bead circuit(s) is too high or low, most likely due to mechanical damage	Replace sensor
8	108	NDIR	LED/PD failure	Signal measurement	Optical components in sensor are not working	Replace sensor
9	109	All	Span calibration is due	Prediction algorithm and countdown timer	Calibration is due, based on either fixed countdown timer or predicted accuracy out of range, whichever occurs first	Recalibrate sensor, this will automatically reset the countdown and prediciton if succesful

TABLE 13. ERRORS

PRIORITY	ERROR CODE	SENSOR TYPE	ERROR DESCRIPTION	DETECTION METHOD	DESCRIPTION	ACTION
10	110	All	Bump test is due	Countdown timer	Predefined time since last bump test has been exceeded. A bump test is a brief exposure of the instrument to the target gas in order to verify that the sensor responds and the alarms function accordingly	Do a bump test*[2] to verify that sensor is working properly. Replace the sensor if bump test fails. If bump test is successful, send command 0x91 to sensor to reset timer
11	111	All	User factor not valid		User factor has not been set or instrument has selected a user factor that is not defined. This is likely to be caused by using a sensor that has not been characterised for use with this instrument. The user factor is used to correct the sensor reading for effects of the instrument on the gas concentration	Check that sensor is intended for use with this instrument. Replace with correct sensor if necessary. Instrument should select the appropriate user factor on startup
12	112	EC, NDIR	Temperature out of range	Temperature out of range (-40 °C, 60°C)	Sensor has been exposed to temperatures outside of the rated operating range, which may have damaged it	Do a bump test* to verify that the sensor is working properly. Replace sensor if bump test fails
13	113	EC	Electrolyte too wet	Diagnostic electrode	The sensor electroyte has absorbed too much water. Due to operation in high humidity for too long. Performance may be degraded, and there is risk of the sensor bursting or leaking due to the volume increase of the electrolyte	Bump test* sensor to confirm if it is still working. Sensor can be put into a dry environment to help it recover. Replace sensor if does not recover. See characterisation note for recovery times
14	118	All	ROM check failed	Non-volatile memory check	The program memory stores the firmware code. After system start-up, the LRC is calculated (32 bytes per second) for the whole app firmware code, then the result is compared with the initial value stored in the ROM memory. If these parameters do not match, an error is flagged.	Power off sensor and wait 30 seconds. Then reconnect and initalise sensor. If the same error is flagged, replace sensor.
15	119	All	RAM check failed	Volatile memory check	Volatile memory is the microcontroller internal RAM used as the runtime data memory. In the firmware, the RAM is tested with checkboard algorithms: during the test, a 4 bytes words are written and the readback is checked.	Power off sensor and wait 30 seconds. Then reconnect and initalise sensor. If the same error is flagged, replace sensor.

Notes:

**If performing a bump test due to any of the other error codes, send reset command to sensor after successful result to clear the error flag.*

**[2] If performing a bump test due to error code 110 (Bump test due), send 0x91 to sensor after successful result to reset the timer, as the sensor itself does not know if has been bump tested.*

APPENDIX 2: DATA DECODING EXAMPLES

I. STARTING UP SENSOR

1. SET WRITE-PROTECT OFF

Sent data packet (from instrument to sensor)

0x7B	Start of packet (Equivalent ASCII character “{“)
0x59	Version
0x07	Length: 7 bytes (From auto-index to EOP: 0x 00 00 A0 00 85 8E 7D)
0x00 0x00	Instrument index
0xA0	Command: Write-protect
0x00	Data: 0 for setting write-protect off
0x85 0x8E	CRC check (0x 7B 59 07 00 00 A0 00 = 0x 85 8E)
0x7D	End of packet (Equivalent ASCII character “}“)

Received data packet (from sensor to instrument)

0x7B	Start of packet (Equivalent ASCII character “{“)
0x59	Version
0x06	Length: 6 bytes (From auto-index to EOP: 0x 00 00 A0 29 85 7D)
0x00 0x00	Auto increment index (sensor)
0xA0	Command: Write-protect
0x29 0x85	CRC check (0x 7B 59 06 00 00 A0 = 0x 29 85)
0x7D	End of packet

2. GO TO WORK MODE

Sent data packet (from instrument to sensor)

0x7B	Start of packet (Equivalent ASCII character “{“)
0x59	Version
0x07	Length: 7 bytes (From auto-index to EOP)
0x00 0x01	Instrument index
0xA6	Command: GoTo_Mode
0x03	Work mode
0x11 0x93	CRC check
0x7D	End of packet

Received data packet (from sensor to instrument)

0x7B	Start of packet (Equivalent ASCII character “{“)
0x59	Version
0x06	Length
0x00 0x01	Auto increment index (Sensor)
0xA6	GoTo_Mode
0xAF 0x92	CRC check
0x7D	EOP

APPENDIX 2: DATA DECODING EXAMPLES

3. GET OEM CODE (OPTIONAL)

Sent data packet (from instrument to sensor)

0x7B	Start of packet (Equivalent ASCII character “{“)
0x59	Version
0x06	Length: 6 bytes (From auto-index to EOP)
0x00 0x02	Instrument index
0x3B	Command: Get_EOM_Code
0x26 0xDF	CRC check
0x7D	End of packet

Received data packet (from sensor to instrument)

0x7B	Start of packet (Equivalent ASCII character “{“)
0x59	Version
0x0C	Length: 12 bytes
0x00 0x02	Auto increment index (Sensor)
0x3B	Command: Get_EOM_Code
0x4E	ASCII code of 0x4E= “N” (Uppercase N)
0x6F	ASCII code of 0x6F= “o” (Lowercase o)
0x4C	ASCII code of 0x4C= “L” (Uppercase L)
0x6F	ASCII code of 0x6F= “o” (Lowercase o)
0x63	ASCII code of 0x63= “c” (Lowercase c)
0x6B	ASCII code of 0x6B= “k” (Lowercase k)
0x08 0x43	CRC check
0x7D	EOP

4. SET RTC

>>Sent data packet (from instrument to sensor)

0x7B	Start of packet (Equivalent ASCII character “{“)
0x59	Version
0x0C	Length: 12 bytes
0x00 0x03	Instrument index
0x82	Command: Set_RTC (real time clock)
0x15	21 (Year=2021)
0x02	Month= 2 (February)
0x12	18 (Day)
0x11	17(Hour)
0x33	51 (Minute)
0x0D	31 (Second)
0x8E 0x80	CRC check
0x7D	EOP

Received data packet (from sensor to instrument)

0x7B	Start of packet (Equivalent ASCII character “{“)
0x59	Version
0x06	Length: 6 bytes
0x00 0x03	Auto increment index (Sensor)
0x82	Command: Set_RTC (real time clock)
0x23 0x49	CRC check
0x7D	EOP

5. SET UF

>>Sent data packet (from instrument to sensor

0x7B	Start of packet (Equivalent ASCII character "{")
0x59	Version
0x08	Length: 8 bytes
0x00 0x04	Instrument index
0x8D	Command: SET_SEN_UF_ INDEX (User Factor Index)
0x00	Sensor index (some sensors can detect more than one target gas and it can be identified by its sensor index)
0x00	User factor number
0xF7 0x75	CRC check
0x7D	EOP

<<Received data packet (from sensor to instrument)

0x7B	Start of packet (Equivalent ASCII character "{")
0x59	Version
0x06	Length: 6 bytes
0x00 0x04	Auto increment index (Sensor)
0x8D	Command: SET_SEN_UF_ INDEX (User Factor Index)
0xB1 0x68	CRC check
0x7D	EOP

6. GET DATA FORMAT

>>Sent data packet (from instrument to sensor

0x7B	Start of packet (Equivalent ASCII character "{")
0x59	Version
0x07	Length
0x00 0x05	Instrument index
0x31	Command: Get_Data_Fmt
0x00	Sensor index (some sensors can detect more than one target gas and it can be identified by its sensor index)
0x63 0xC3	CRC check
0x7D	EOP

<<Received data packet (from sensor to instrument)

0x7B	Start of packet (Equivalent ASCII character "{")
0x59	Version
0x07	Length
0x00 0x05	Auto increment index (Sensor)
0x31	Command: Get_Data_Fmt
0x00	Unit code: 0x00=ppm (parts per million)
0x01	Reading resolution integer=1
0x00	Reading resolution exponent=0
0x08 0x77	Mask parameter: High and low mask parameter respectively b0000 1000 0111 0111 Enabled: B0=span; B1=low alarm; B2=high alarm; B4=overrange; B5=STEL; B6=TWA; B11=Drift.
0x3C 0x9F	CRC check
0x7D	EOP

7. GET END OF LIFE PREDICTION (OPTIONAL)

>>Sent data packet (from instrument to sensor

0x7B	Start of packet (Equivalent ASCII character "{")
0x59	Version
0x07	Length: 7 bytes
0x00 0x06	Instrument index
0x41	Command: Get_End_Of_Life
0x00	Sensor index (some sensors can detect more than one target gas and it can be identified by its sensor index)
0x43 0xF9	CRC check
0x7D	EOP

<<Received data packet (from sensor to instrument)

0x7B	Start of packet (Equivalent ASCII character "{")
0x59	Version
0x08	Length: 8 bytes
0x00 0x06	Auto increment index (Sensor)
0x41	Command: Get_End_Of_Life
0x07 0x21	1825 days to End of Life
0xC2 0x43	CRC check
0x7D	EOP

8. GET PREDICTIVE CALIBRATION (OPTIONAL)

>>Sent data packet (from instrument to sensor

0x7B	Start of packet (Equivalent ASCII character "{")
0x59	Version
0x07	Length: 7 bytes
0x00 0x07	Instrument index
0x42	Command: Get_PredCal_Due_Days
0x00	Sensor index (some sensors can detect more than one target gas and it can be identified by its sensor index)
0xC9 0xEE	CRC check
0x7D	EOP

<<Received data packet (from sensor to instrument)

0x7B	Start of packet (Equivalent ASCII character "{")
0x59	Version
0x08	Length: 8 bytes
0x00 0x07	Auto increment index (Sensor)
0x42	Command: Get_PredCal_Due_Days
0x00 0xB4	180 days to calibrate sensor
0xC7 0x01	CRC check
0x7D	EOP

COMMUNICATION PROTOCOL
APPENDIX 2: DATA DECODING EXAMPLES

II. OBTAINING GAS READINGS

1. GET DATA PACK BEFORE THE SENSOR IS READY TO TAKE MEASUREMENTS (SENSOR IN WARM UP)

>>Sent data packet (from instrument to sensor)

0x7B	Start of packet
0x59	Version
0x09	Length
0x00 0x06	Instrument index
0x30	Command: Get_data_pack
0x00	Sensor index (some sensors can detect more than one target gas and it can be identified by its sensor index)
0x00 0x2F	Bitmap: High and low respectively. b0010 1111 Flags: B0=Status; B1=Alarm; B2=Error; B3=Gas reading; B5=Temperature.
0x52 0x06	CRC check
0x7D	End of packet

<<Received data packet (from sensor to instrument)

0x7B	Start of packet
0x59	Version
0x0E	Length: 14 bytes
0x00 0x06	Auto increment index (sensor)
0x30	Command: Get_data_pack
0x02	Status: b0000 0010. B1= In warm up
0x04	Alarm: b0000 0100. B2= Time is not synchronised (RTC is not set up)
0x00	Error: b0000 0000 = No error
0xFF 0xFF 0xFF 0xFF	Gas reading: when sensor is in warm-up or in sleep mode the gas measurements are not valid
0xFF	Temperature reading is not available when the sensor is in warm up or in sleep mode
0x04 0x6C	CRC check
0x7D	End of packet

2. GET DATA PACK AFTER WARMING UP. GAS MEASUREMENT READING WITH 1 ERROR AND 1 ALARM.

>>Sent data packet (from instrument to sensor):

0x7B	Start of packet
0x59	Version
0x09	Length: 9 bytes
0x00 0x08	Instrument index
0x30	Command: Get_data_pack
0x00	Sensor index (some sensors can detect more than one target gas and it can be identified by its sensor index)
0x00 0x2F	Bitmap: High and low respectively. b0010 1111 Flags: B0=Status; B1=Alarm; B2=Error; B3=Gas reading; B5=Temperature.
0xD0 0xD5	CRC check
0x7D	End of packet

<<Received data packet (from sensor to instrument)

0x7B	Start of packet
0x59	Version
0x0F	Length: 15 bytes
0x00 0x08	Auto increment index (sensor)
0x30	Command: Get_data_pack
0x00	Status: b0000 0000. Sensor in work mode
0x10	Alarm: b0001 0000. B4=Low alarm flagged
0x01	Error: 1 error flagged
0x6D	Error code: 109 (Span calibration is due)
0x00 0x00 0x10 0x68	Gas reading= (HEX to DEC/100)= (4200/100)=42 Note: Units are requested with CMD 0x31: Get_Sen_Data_Fmt
0x09B	Temperature reading in sensor: (HEX to DEC) - 127 = 155-127= 28 C
0x23 0x33	CRC check
0x7D	End of packet

3. GET DATA PACK: GAS READING, 1 ALARM AND 2 ERRORS

>>Sent data packet (from instrument to sensor):

Same than above (Example II.2)

<<Received data packet (from sensor to instrument)

0x7B	Start of packet
0x59	Version
0x10	Length: 10 bytes
0x00 0x08	Auto increment index (sensor)
0x30	Command: Get_data_pack
0x00	Status: b0000 0000. Sensor in work mode
0x40	Alarm: b0100 0000. B6=TWA alarm flagged (Time-weighted average)
0x02	Errors: 2 errors flagged
0x6E	Error code: 110 (Bump test is due)
0x6F	Error code: 111 (User factor not valid)
0x00 0x00 0x02 0xBC	Gas reading= (HEX to DEC/100)= (700/100)=7 Note: Units are requested with CMD 0x31: Get_Sen_Data_Fmt
0x81	Temperature reading in sensor: (HEX to DEC) - 127 = 129-127= 2 C
0xDF 0x8B	CRC check
0x7D	End of packet

III. INFORMATION COMMAND: REQUEST TARGET GAS

Sent data packet (from instrument to sensor)

0x7B	Start of packet
0x59	Version
0x07	Length: 7 bytes
0x00 0x08	Index (From instrument)
0x35	Command: Get_Sen_GasName
0x00	Sensor index (some sensors can detect two gases)
0x7B 0x27	CRC check
0x7D	End of packet

<<Received data packet (from sensor to instrument)

0x7B	Start of packet
0x59	Version
0x09	Length
0x00 0x08	Auto increment index (sensor)
0x35	Command: Get_Sen_GasName
0x43	ASCII code of 0x43= “C” (Uppercase C)
0x4F	ASCII code of 0x4F= “O” (Uppercase O)
0x00	Data: 0x00 (null chart). Indicates the end of ASCII string.
0x33 0x0D	CRC check
0x7D	EOP

IV. ALOHA MODE

1. SET ALOHA MODE: 300 SECONDS BY PERIOD

Sent data packet (from instrument to sensor)

0x7B	Start of packet
0x59	Version
0x0A	Length: 10 bytes
0x00 0x0A	Index (instrument)
0xA2	Command: Aloha_Configuration
0x00	Sensor index (some sensors can detect more than one target gas and it can be identified by its sensor index)
0x01	B0 = Aloha by period (case 2 in SDCS document)
0x01 0x2C	300 seconds
0x48 0x65	CRC check
0x7D	End of packet

<<Received data packet (from sensor to instrument)

0x7B	Start of packet
0x59	Version
0x06	Length: 6 bytes
0x00 0x0A	Auto increment index (sensor)
0xA2	Command: Aloha_Configuration
0x95 0x8A	CRC Check
0x7D	End of packet

APPENDIX 2: DATA DECODING EXAMPLES

2. GET ALOHA MODE:

>>Sent data packet (from instrument to sensor)

0x7B	Start of packet
0x59	Version
0x07	Length
0x00 0x0B	Index (Instrument)
0x53	Command: Get aloha mode
0x00	Sensor index (some sensors can detect more than one target gas and it can be identified by its sensor index)
0xAF 0x1E	CRC check
0x7D	End of packet

<<Received data packet (from sensor to instrument)

0x7B	Start of packet
0x59	Version
0x09	Length
0x00 0x0B	Auto increment index (sensor)
0x53	Command: Get aloha mode
0x01	Gas threshold (case 2 in SDCS document)
0x01 0x2C	Period in seconds: 300 seconds
0xEA 0x52	CRC check
0x7D	End of packet

3. ALOHA DATA PACK (RECEIVED EVERY 300 SECONDS):

<<Received data packet (from sensor to instrument)

0x7B	Start of packet
0x59	Version
0x0E	Length = 14 bytes
0x00 0x0C	Auto increment index (sensor)
0xA3	Command: Aloha_Data_Pack
0x00	Sensor index (some sensors can detect more than one target gas and it can be identified by its sensor index)
0x00	Status
0x04	Alarm: 0100 = B2 (RTC hasn't been configured)
0x00	Error code (No errors)
0x00 0x00 0x00 0x00 0x00	Gas reading = 0
0x74 0x0C	CRC check
0x7D	End of packet

APPENDIX 2: DATA DECODING EXAMPLES

IV. SENSOR PARAMETERS

1. GET SENSOR PARAMETERS

>>Sent data packet (from instrument to sensor)

0x7B	Start of packet
0x59	Version
0x09	Length
0x02 0x12	Index (instrument)
0x33	Command: Get Sensor Parameters
0x00	Sensor index (some sensors can detect more than one target gas and it can be identified by its sensor index)
0x00	Mask parameter, high: b0000 0000 (No parameter requested)
0x43	Mask parameter, low: b0100 0011 Parameters requested: B0=Span; B1=Low alarm; B6=TWA
0x69 0x0E	CRC check
0x7D	End of packet

<<Received data packet (from sensor to instrument)

0x7B	Start of packet
0x59	Version
0x12	Length: 18 bytes
0x00 0x15	Auto increment index (sensor) The auto increment index is independent from the sensor instrument index, so the values can be different
0x33	Command: Get sensor parameters
0x00 0x00 0x27 0x10:	First received parameter is Span 0x 2710= 10 000 Parameter= HEX to DEC/100= 100
0x00 0x00 0x0B 0xB8:	Second received parameter is Low alarm 0x 0BB8= 3 000 Parameter = HEX to DEC/100= 30
0x00 0x00 0x0D 0xAC	TWA parameter 0x 0DAC= 3 500 Parameter = HEX to DEC/100= 35
0xEF 0x10	CRC check
0x7D	End of packet

COMMUNICATION PROTOCOL
APPENDIX 2: DATA DECODING EXAMPLES

2. SET SENSOR PARAMETERS

Sent data packet (from instrument to sensor)

0x7B	Start of packet
0x59	Version
0x11	Length: 17 bytes
0x00 0x14	Index (instrument)
0x80	Command: Set Sensor parameter
0x00	Sensor index (some sensors can detect more than one target gas and it can be identified by its sensor index)
0x00	Mask Parameter, high (No parameter selected)
0x24	Mask Parameter, low: 0010 0100
	Parameters to be configured: B2=High alarm; B5=STEL
0x00 0x00 0x2A 0xF8	First parameter to be configured is High Alarm
	0x2AF8= 11 000
	Parameter= HEX to DEC/100= 110
0x00 0x00 0x4E 0x20	Second parameter to be configured is STEL
	0x4E20= 20 000
	Parameter= HEX to DEC/100= 200
0x0B 0x15	CRC check
0x7D	End of packet

<<Received data packet (from sensor to instrument)

0x7B	Start of packet
0x59	Version
0x06	Length: 6 bytes
0x00 0x17	Auto increment index (sensor)
	The auto increment index is independent from the sensor instrument index, so the values can be different
0x80	Command: Set sensor parameters
0xDB 0x46	CRC check
0x7D	End of packet

3. SET SENSOR PARAMETERS (FAILED REQUEST: WRITE PROTECT IS ENABLED)

>>Sent data packet (from instrument to sensor)

Same than above V.2

<<Received data packet (from sensor to instrument)

0x7B	Start of packet
0x59	Version
0x07	Length
0x00 0x20	Auto increasing index (sensor)
	The auto increment index is independent from the sensor instrument index, so the values can be different
0x71	Error
0x39	Error code: Fail_WriteProtect. To use this command, it is necessary to set write-protect off
0x61 0x94	CRC check
0x7D	End of packet

COMMUNICATION PROTOCOL
APPENDIX 2: DATA DECODING EXAMPLES

VI. CALIBRATION

1. ZERO CALIBRATION
STEP 0: RTC & GET_CAL_TIME REQUEST

Resynchronise RTC:

>>Sent data packet (from instrument to sensor):

0x7B 0x59 0x0C 0x00 0x19 **0x82** 0x15 0x02 0x15 0x14 0x33 0x0A 0xC2 0x8F 0X7D

0x82 Command: Set_RTC (real time clock)
0x15 21 (Year=2021)
0x02 Month= 2 (February)
0x15 21 (Day)
0x14 20 (Hour)
0x33 51 (Minute)
0x0A 10 (Second)

<<Received data packet (from sensor to instrument):

0x7B 0x59 0x06 0x00 0x19 **0x82** 0xFF 0x4A 0x7D

0x82 Command: Set_RTC (real time clock)

Request Calibration Time:

>>Sent data packet (from instrument to sensor):

0x7B 0x59 0x07 0x00 0x1A **0x43** 0x00 0x4C 0xD1 0x7D

0x43 Command: Get_Sen_Cal_Time
0x00 Sensor index

<<Received data packet (from sensor to instrument):

0x7B 0x59 0x08 0x00 0x1A **0x43** 0x00 0x3C 0x6A 0x94 0x7D

0x43 Command: Get_Sen_Cal_Time
0x00 0x3C Calibration time: 60 seconds

CALIBRATION STEP 1: USER_CAL PREPARE REQUEST

>>Sent data packet (from instrument to sensor):

0x7B 0x59 0x0A 0x00 0x1B **0xA1** 0x00 0x01 0x00 0x80 0x84 0x29 0x7D

0xA1 Command: Calibration
0x00 Sensor Index
0x01 Bit Map enabled
0x00 Calibration type: zero calibration
0x80 Operation: Prepare for calibration

<<Received data packet (from sensor to instrument):

0x7B 0x59 0x06 0x00 0x1B **0xA1** 0x66 0xAB 0x7D

0xA1 Command: Calibration

CALIBRATION STEP 2: USER_CAL START REQUEST

>>Sent data packet (from instrument to sensor):

0x7B 0x59 0x0A 0x00 0x1C **0xA1** 0x00 0x01 0x00 0x00 0x5E 0x12 0x7D

0xA1 Command: Calibration
0x00 Sensor Index
0x01 Bit Map enabled
0x00 Calibration: Zero calibration
0x00 Operation: Start user calibration

COMMUNICATION PROTOCOL
APPENDIX 2: DATA DECODING EXAMPLES

<<Received data packet (from sensor to instrument):

0x7B 0x59 0x08 0x00 0x1C **0xA1** 0x03 0x20 0x9F 0xC0 0x7D

0xA1 Command: Calibration

0x03 0x20 = 800 milliseconds (time cost)

CALIBRATION STEP 3: USER_CAL GET CALIB RESULT REQUEST

>>Sent data packet (from instrument to sensor):

0x7B 0x59 0x0A 0x00 0x1D **0xA1** 0x00 0x01 0x00 0x83 0x25 0x18 0x7D

0xA1 Command: Calibration

0x00 Sensor Index

0x01 Bit Map enabled

0x00 Calibration: Zero calibration

0x83 Operation: Get calibration result

<<Received data packet (from sensor to instrument):

0x7B 0x59 0x08 0x00 0x1D **0xA1** 0x00 0x01 0x01 0x05 0x7D

0xA1 Command: Calibration

0x00 Sensor index

0x01 Calibration result= 1 (Successful calibration)

2. SPAN CALIBRATION

A. REQUESTING AND CHANGING SPAN CALIBRATION

>>Sent data packet (from instrument to sensor):

0x7B 0x59 0x09 0x00 0x09 **0x33** 0x00 0x00 0x01 0x6C 0x4A 7D

0x33 Command: Get sensor parameter

0x00 Sensor index

0x00 0x01 Mask parameter = b0000 0000 0000 0001. B0=Span. Span calibration requested.

<<Received data packet (from sensor to instrument):

0x7B 0x59 0x0A 0x00 0x09 **0x33** 0x00 0x00 0x05 0xDC 0x61 0xAA 0x7D

0x33 Command: Get sensor parameter

0x00 0x00 0x07 0xD0 Span calibration value = (HEX TO DEC)/100= 20

>>Sent data packet (from instrument to sensor):

0x7B 0x59 0x0D 0x00 0x10 **0x80** 0x00 0x00 0x01 0x00 0x00 0x09 0xC4 0x74 0xF2 0x7D

0x80 Command: Set sensor parameter

0x00 Sensor index

0x00 0x01 Mask parameter = b0000 0001 = B1(Span). Span calibration to be changed.

0x00 0x00 0x07 0xD0 New span calibration value = (HEX TO DEC)/100= 25.

<<Received data packet (from sensor to instrument):

0x7B 0x59 0x06 0x00 0x10 **0x80** 0x49 0x45 0x7D

0x80 Command: Set sensor parameter

B. SPAN CALIBRATION: ABORTED SPAN CALIBRATION

1. ZERO CALIBRATION

Step 0: RCT & GET_CAL_TIME REQUEST

Resynchronise RTC:

>>Sent data packet (from instrument to sensor):

0x7B 0x59 0x0C 0x00 0x11 **0x82** 0x15 0x02 0x15 0x15 0x33 0x0A 0xC2 0xA8 0x7D

0x82 Command: Set_RTC (real time clock)
0x15 21 (Year=2021)
0x02 Month= 2 (February)
0x15 21 (Day)
0x15 21 (Hour)
0x33 51 (Minute)
0x0A 10 (Second)

<<Received data packet (from sensor to instrument):

0x7B 0x59 0x06 0x00 0x11 **0x82** 0xAF 0x49 0x7D

0x82 Command: Set_RTC (real time clock)

Request Calibration Time:

>>Sent data packet (from instrument to sensor):

0x7B 0x59 0x07 0x00 0x12 **0x43** 0x00 0xE5 0x70 0x7D

0x43 Command: Get_Sen_Cal_Time
0x00 Sensor index

<<Received data packet (from sensor to instrument):

0x7B 0x59 0x08 0x00 0x12 **0x43** 0x00 0x3C 0xEF 0x57 0x7D

0x43 Command: Get_Sen_Cal_Time
0x03 0x20 = 800 milliseconds (time cost)

Calibration step 1: USER_CAL PREPARE REQUEST

>>Sent data packet (from instrument to sensor):

0x7B 0x59 0x0A 0x00 0x13 **0xA1** 0x00 0x01 0x01 0x80 0xBA 0x5A 0x7D

0xA1 Command: Calibration
0x00 Sensor Index
0x01 Bit Map enabled
0x01 Calibration type: Span calibration
0x80 Operation: Prepare for calibration

<<Received data packet (from sensor to instrument):

0x7B 0x59 0x06 0x00 0x13 **0xA1** 0xEF 0x02 0x7D

0xA1 Command: Calibration

Calibration step 2*: USER_CAL ABORT CALIB REQUEST

>>Sent data packet (from instrument to sensor):

0x7B 0x59 0x0A 0x00 0x14 **0xA1** 0x00 0x01 0x01 0x81 0x9B 0x1E 0x7D

0xA1 0x00 0x01 0x01 0x81
0xA1 Command: Calibration
0x00 Sensor Index
0x01 Bit Map enabled
0x81 Abort calibration

<<Received data packet (from sensor to instrument):

0x7B 0x59 0x06 0x00 0x14 **0xA1** 0xD1 0x80 0x7D

0xA1 Command: Calibration

APPENDIX 3: CRC-16 CYCLIC REDUNDANCY

CRC-16 IS BASED ON POLYNOMIAL $X^{16} + X^{15} + X^2 + 1$, 0x8005

CRC detailed parameters:

CRC width:	**CRC-16**
Polynomial:	**0x8005**
Initial value:	**0x0000**
Final Xor value:	**0x0000**

```
#define CRC16_POLY 0x8005
#define CRC16_INIT_REM 0x0000
#define CRC16_FINAL_XOR 0x0000
#define CRC_TABLE_SIZE 256
const unsigned short crc16Table[CRC_TABLE_SIZE] = {
0x0000, 0x8005, 0x800F, 0x000A,
0x801B, 0x001E, 0x0014, 0x8011,
0x8033, 0x0036, 0x003C, 0x8039,
0x0028, 0x802D, 0x8027, 0x0022,
0x8063, 0x0066, 0x006C, 0x8069,
0x0078, 0x807D, 0x8077, 0x0072,
0x0050, 0x8055, 0x805F, 0x005A,
0x804B, 0x004E, 0x0044, 0x8041,
0x80C3, 0x00C6, 0x00CC, 0x80C9,
0x00D8, 0x80DD, 0x80D7, 0x00D2,
0x00F0, 0x80F5, 0x80FF, 0x00FA,
0x80EB, 0x00EE, 0x00E4, 0x80E1,
0x00A0, 0x80A5, 0x80AF, 0x00AA,
0x80BB, 0x00BE, 0x00B4, 0x80B1,
0x8093, 0x0096, 0x009C, 0x8099,
0x0088, 0x808D, 0x8087, 0x0082,
0x8183, 0x0186, 0x018C, 0x8189,
0x0198, 0x819D, 0x8197, 0x0192,
0x01B0, 0x81B5, 0x81BF, 0x01BA,
0x81AB, 0x01AE, 0x01A4, 0x81A1,
0x01E0, 0x81E5, 0x81EF, 0x01EA,
0x81FB, 0x01FE, 0x01F4, 0x81F1,
0x81D3, 0x01D6, 0x01DC, 0x81D9,
0x01C8, 0x81CD, 0x81C7, 0x01C2,
0x0140, 0x8145, 0x814F, 0x014A,
0x815B, 0x015E, 0x0154, 0x8151,
0x8173, 0x0176, 0x017C, 0x8179,
0x0168, 0x816D, 0x8167, 0x0162,
0x8123, 0x0126, 0x012C, 0x8129,
0x0138, 0x813D, 0x8137, 0x0132,
0x0110, 0x8115, 0x811F, 0x011A,
0x810B, 0x010E, 0x0104, 0x8101,
0x8303, 0x0306, 0x030C, 0x8309,
```

APPENDIX 3: CRC-16 CYCLIC REDUNDANCY CHECK

```
0x0318, 0x831D, 0x8317, 0x0312,
0x0330, 0x8335, 0x833F, 0x033A,
0x832B, 0x032E, 0x0324, 0x8321,
0x0360, 0x8365, 0x836F, 0x036A,
0x837B, 0x037E, 0x0374, 0x8371,
0x8353, 0x0356, 0x035C, 0x8359,
0x0348, 0x834D, 0x8347, 0x0342,
0x03C0, 0x83C5, 0x83CF, 0x03CA,
0x83DB, 0x03DE, 0x03D4, 0x83D1,
0x83F3, 0x03F6, 0x03FC, 0x83F9,
0x03E8, 0x83ED, 0x83E7, 0x03E2,
0x83A3, 0x03A6, 0x03AC, 0x83A9,
0x03B8, 0x83BD, 0x83B7, 0x03B2,
0x0390, 0x8395, 0x839F, 0x039A,
0x838B, 0x038E, 0x0384, 0x8381,
0x0280, 0x8285, 0x828F, 0x028A,
0x829B, 0x029E, 0x0294, 0x8291,
0x82B3, 0x02B6, 0x02BC, 0x82B9,
0x02A8, 0x82AD, 0x82A7, 0x02A2,
0x82E3, 0x02E6, 0x02EC, 0x82E9,
0x02F8, 0x82FD, 0x82F7, 0x02F2,
0x02D0, 0x82D5, 0x82DF, 0x02DA,
0x82CB, 0x02CE, 0x02C4, 0x82C1,
0x8243, 0x0246, 0x024C, 0x8249,
0x0258, 0x825D, 0x8257, 0x0252,
0x0270, 0x8275, 0x827F, 0x027A,
0x826B, 0x026E, 0x0264, 0x8261,
0x0220, 0x8225, 0x822F, 0x022A,
0x823B, 0x023E, 0x0234, 0x8231,
0x8213, 0x0216, 0x021C, 0x8219,
0x0208, 0x820D, 0x8207, 0x0202
};
unsigned short crc16MakeTableMethod(unsigned short crc, const unsigned short *table,
unsigned char *pbuffer, unsigned int length)
{
while(length--)
crc = table[((crc >> 8) ^ *pbuffer++)] ^ (crc << 8); // normal
return(crc ^ CRC16_FINAL_XOR);
}
unsigned char msg[10] = “123456789”;
unsigned short length = 9;
crc16 = crc16MakeTableMethod(CRC16_INIT_REM, crc16Table, msg, length);
// the expected CRC: crc16=0xFEE8
```

WARRANTY/REMEDY

Honeywell warrants goods of its manufacture as being free of defective materials and faulty workmanship during the applicable warranty period. Honeywell's standard product warranty applies unless agreed to otherwise by Honeywell in writing; please refer to your order acknowledgment or consult your local sales office for specific warranty details. If warranted goods are returned to Honeywell during the period of coverage, Honeywell will repair or replace, at its option, without charge those items that Honeywell, in its sole discretion, finds defective. **The foregoing is buyer's sole remedy and is in lieu of all other warranties, expressed or implied, including those of merchantability and fitness for a particular purpose. In no event shall Honeywell be liable for consequential, special, or indirect damages.**

While Honeywell may provide application assistance personally, through our literature and the Honeywell web site, it is buyer's sole responsibility to determine the suitability of the product in the application.

Specifications may change without notice. The information we supply is believed to be accurate and reliable as of this writing. However, Honeywell assumes no responsibility for its use.

FOR MORE INFORMATION

Honeywell Advanced Sensing Technologies services its customers through a worldwide network of sales offices and distributors. For application assistance, current specifications, pricing, or the nearest Authorized Distributor, visit sps.honeywell.com/ast or call:

USA/Canada	+302 613 4491
Latin America	+1 305 805 8188
Europe	+44 1344 238258
Japan	+81 (0) 3-6730-7152
Singapore	+65 6355 2828
Greater China	+86 4006396841

Honeywell
Advanced Sensing Technologies
830 East Arapaho Road
Richardson, TX 75081
sps.honeywell.com/ast

002712-2-EN (External) | 2 | 08/21

Honeywell