GE Sensing
Telaire Products

T660x Series
UART Communications Protocol
Customer Version

(Document Revision 00)

Document Revisions:

7/31/06 – Original Document

GE Sensing
Telaire Products

T660x Series
UART Communications Protocol
Customer Version

Table of Contents

1. Communications Interface

The T660x Series Sensors communicate with an external host by means of an asynchronous, RS-232, UART serial communications port. All communications over this UART serial interface must be wrapped in the proprietary Telaire Tsunami-Lite Communications Protocol.

2. UART Communications Logic and Timing

Timing information for the T660x Series may be considered at three levels:

- Timing related to byte/command transfers
- General system timing
- Communications at Power-Up

2.1. Timing related to byte/command transfers

The UART communications interface expects a frame size of 8 bits, no parity, one stop bit, and a baud rate of 19200.

No communication reply can be initiated during data acquisition or processing. When a sensor fails to reply to a command request, simply send the request again. This phenomenon is more apparent in models with a fast cycle period.

2.2. General System Timing

The internal cycle of data acquisition and processing is dependent upon the sensor model. The duration of the cycle may be from 1 to several seconds. For a given sensor model, it is advised to keep the interval for gas concentration requests no shorter than the internal data acquisition and processing cycle. The host could interrogate the sensor more frequently for concentration readings, but it is not recommended.

The time interval between other commands is less restricted. In general, with the exception of a Status command following a Calibration command, a subsequent command can be issued as soon as the reply from the previous command has been received. For a Status command following a Calibration command, the host should wait at least 2 to 4 seconds before issuing the first Status command. This allows time for the Sensor to begin the calibration process.

2.3. Communications at Power-Up

A communications delay of several seconds occurs when the Sensor is powered up or power cycled. This communications delay time is necessary for the sensor to initialize and achieve full functionality.

After initialization, the Sensor stays in a Warm-up mode. The duration of the Warm-up period is dependent upon the Sensor model. The difference between Warm-Up mode and normal operating mode is that in the Warm-Up the

Sensor may not yet report accurate readings, and hence cannot execute any calibration commands. All other commands can be executed during Warm-up.

The Status of the sensor can be checked by using the Status command (see the commands description below). This command returns the status byte with a number of flags, including the Warm-Up status flag.

The gas ppm concentration can be read while the sensor is in Warm-Up mode; however, the data may not be accurate.

NOTE: If for any reason the sensor does not respond to a request, simply re-send the command.

3. UART Serial Communications Interface

The T660x Series Sensor communicates over an asynchronous, UART interface at 19200 baud, no parity, 8 data bits, and 1 stop bit. When a host computer or PC communicates with the Sensor, the host computer sends a request to the Sensor, and the Sensor returns a response. The host computer acts as a master, initiating all communications, and the Sensor acts as a slave, responding with a reply.

All Sensor commands and replies are wrapped in the proprietary Telaire Tsunami-Lite Communications Protocol to insure the integrity and reliability of the data exchange. The Communications Protocol for the serial interface and the Command Set for the T660x Sensor are described in detail in the sections that follow.

3.1. UART Tsunami-Lite Communications Protocol

Each command to the Sensor consists of a length byte, a command byte, and any additional data required by the command. Each response from the Sensor consists of a length byte and the response data if any. Both the command to the sensor and the response from the Sensor are wrapped in the Tsunami-Lite communications protocol layer.

Command:	<length><command><additional_data>
Response:	<length><response_data>

The communications protocol consists of a flag bytes (0xFF) and an address byte as a header. The protocol has no trailer.

Header	Message Body
<flag><address>	<Command/Response>

3.2. UART Commands from PC to Sensor

Commands sent from a host computer or PC to the Sensor have the following format:

<flag><address><length><command><additional_data>

where:

<flag> the hex value 0xFF

<address>	one byte hex value. The byte 0xFE is an address to which all sensors respond.
<length>	total length in bytes of the command and additional data
<command>	one byte hex command, values explained below
<additional_data>	may or may not be applicable, depending upon the command

For example, to request Sensor identification, the following command is used:

```
0x FF    0xFE     0x02      0x02      0x01
<flag>  <address>  |           |          |
                  <length>    |        <additional data> = SERIAL_NUMBER
                           <command> = CMD_READ
```

The length of the command is 0x02, since the command CMD_READ, SERIAL_NUMBER consists of the two bytes "0x02 0x01".

3.3. UART Response from Sensor to PC

Responses returned from the Sensor to the host computer or PC have the following format:

<flag><address><length><response_data>

where:

<flag>	the hex value 0xFF.
<address>	one byte hex value. The byte 0xFA signifies "to master" in a master/slave communication.
<length>	total length in bytes of the response data
<response_data>	may or may not be applicable, depending upon the command

In response to the above identification command CMD_READ SERIAL_NUMBER, one Sensor replied with the following byte stream:

```
0xFF   0xFA   0x0F   0x4E 0x4F  0x42 0x30 0x30 0x31 0x32 0x34  0x00 0x00 0x00 0x00 0x00 0x00 0x00
<flag> <address> |   <response_data>----------------------------------------------------------------------------------------------------|
                 <length>
```

The length of the response_data is fifteen bytes (0x0F). The first eight bytes of the response_data, "4E 4F 42 30 30 31 32 34", is the ASCII string "NOB00124", the serial number for the sensor. The remaining bytes of the fifteen byte response are filled with nulls.

3.4. UART Acknowledgement or <ACK> Reply

Some commands require that a Sensor only confirm that the command was received and the appropriate action was taken. In this case, when a Sensor does not need to return data in response to a command, it will instead reply with an Acknowledgement response, called an <ACK>. This is a response packet formatted as shown above, but with the <length> equal to 0x00, and no response data present:

0xFF	0xFA	0x00
<flag>	<address>	<length>

Examples of commands that expect an Acknowledgement response are Update Commands, Calibrate Commands, and the Skip Warmup Command. Detailed descriptions of these commands are given below.

4. Command Reference for the T660x Sensor

Every common exchange of data between a host processor (or PC) and the Sensor starts with a request data-packet sent to the Sensor, followed by a response data-packet returned from the Sensor. The request data-packet contains a command byte telling what data or sensor action is required. The command byte also determines what additional data is included in the request packet.

NOTE: Each request and response must be wrapped in the Tsunami-Lite communications protocol, as described above. The following Command Reference gives only the command syntax and response, and omits the protocol wrapping.

In the following Commands Tables, hex bytes are represented as '0x12' for clarity. However, when sending the byte string in a message, the '0x' notation must be omitted.

4.1. CMD_READ Commands

The CMD_READ Command reads a data value or parameter from the Sensor. For almost all <data ID> values, this value is read from RAM. Only COMPILE_SUBVOL and COMPILE_DATE are read directly from FLASH.

CMD_READ = 0x02 Req: 0x02 <data ID> Resp: <data> [… <data>]	Read a data value or parameter from the Sensor Request is the command byte, 0x02, followed by a byte number that identifies which data value or parameter to read. Response is one or more bytes of data. Details of useful values that can be read from the T660x Sensor follow.
CMD_READ GAS_PPM Req: 0x02 0x03	Read the gas PPM as measured by the Sensor. Response is a 2-byte binary value, least significant byte first, giving the PPM value between 0 and

Resp: <ppm_lsb>, <ppm_msb>	65,535. For some sensor models, this value must be multiplied by 16 to obtain the actual PPM.
CMD_READ SERIAL_NUMBER Req: 0x02 0x01 Resp: [ASCII string, 15 bytes, null filled]	Read the serial number from the Sensor. Response is 15 bytes, the first of which are an ASCII string of printable characters, for example "074177". The remaining bytes of the response are the null character, 0x00.
CMD_READ COMPILE_SUBVOL Req: 0x02 0x0D Resp: [3-byte ASCII string]	Read the compilation subvolume for the Sensor control software. COMPILE_DATE and COMPILE_SUBVOL together identify the software version. Response is a 3-byte ASCII string, for example "A10".
CMD_READ COMPILE_DATE Req: 0x02 0x0C Resp [6-byte ASCII string]	Read the compilation date for the sensor control software. COMPILE_DATE and COMPILE_SUBVOL together identify the software version. Response is an ASCII string representing a date, for example "060708" for July 8, 2006.
CMD_READ ELEVATION Req: 0x02 0x0F Resp <elevation_lsb> <elevation_msb>	Read the elevation in feet above sea level, a required operating parameter for the Sensor. The Sensor's elevation setting is used to estimate air pressure and is factored into the calculation of the PPM. Response is a 2-byte binary value, least significant byte first, giving the elevation value between 0 and 65,535 feet.

4.2. CMD_UPDATE Commands

The CMD_UPDATE Command writes a data value to both RAM and FLASH memories.

CMD_UPDATE ELEVATION Req: 0x03 0x0F <elevation_lsb> <elevation_msb>	Set/Write the elevation in feet above sea level, a required operating parameter for the Sensor. Elevation is expressed as a 2-byte binary value, least significant byte first. Elevation are normally expressed in increments of 500 feet.
Resp <ACK>	Response is an "acknowledgement" or <ACK> , a response data-packet with the length byte set to zero

	and no data bytes. The CMD_UPDATE command should be followed by the corresponding CMD_READ command to verify that the expected value was written.

4.3. WARMUP Command

CMD_WARM Req: 0x84 Resp: <ACK> or <no response>	Reset the sensor, which puts it in a known state, similar to power up. The Sensor initializes itself, waits a period of time in warm-up mode, and then starts to measure gas PPM. The Sensor attempts to send an <ACK>, but transmission may be aborted by the reset. The Sensor experiences the same communications delay as at power-up.

4.4. CALIBRATION Command

CMD_ZERO_CALIBRATE Req: 0x97 Resp: <ACK>	This command tells the Sensor to start zero calibration. Before sending this command, the zero gas (such as nitrogen) should be flowing to the sensor. The <ACK> response indicates that the calibration request has been received. To verify that calibration has started, wait 2 to 4 seconds and then send command CMD_STATUS to see if the calibration bit is set. When calibration is finished, the calibration bit in the status byte is cleared. A zero calibration will not start if a Sensor is in warm-up mode or in error condition. See Communication Examples, below.

4.5. STATUS and OPERATING Commands

CMD_STATUS Req: 0xB6 Resp: <status>	Read a status byte from the Sensor. The status byte indicates whether the sensor is functioning and is measuring PPM concentration. The response is a single byte, <status>, of bit flags. (Note: bit 0 is the least significant bit.) Bit 0: Error Bit 1: Warmup Mode Bit 2: Calibration Bit 3: Idle Mode Bits 4 - 7: (internal) If a given status bit is "1", the sensor is in that state or mode. If a status bit is "0", the sensor is not in that mode.
CMD_IDLE_ON Req: 0xB9 0x01 Resp: <ACK>	This command tells the Sensor to go into Idle Mode. In Idle Mode, the Lamp is turned off and no data collection takes place. Send the CMD_STATUS command to verify that the Sensor has entered Idle Mode (status bit 3 = 1).
CMD_IDLE_OFF Req: 0xB9 0x02 Resp: <ACK>	This command tells the Sensor to exit Idle Mode and resume data collection. The Sensor resumes data collection as soon as the command is received. However, several data cycles (similar to Warm-Up) are required before PPM readings are accurate. Send the CMD_STATUS command to verify that the Sensor has come out of Idle Mode (status bit 3 = 0).
CMD_ABC_LOGIC Req: 0xB7 0x00 Resp: <abc_state>	This command queries the Sensor for its ABC_LOGIC state. If ABC_LOGIC is ON, <abc_state> = 0x01. If ABC_LOGIC is OFF, <abc_state> = 0x02.
CMD_ ABC_LOGIC _ON Req: 0xB7 0x01 Resp: <0x01>	This command turns the ABC_LOGIC ON. The reply <0x01> indicates that the ABC_LOGIC has been turned on. ***NOTE: This command should only be used on CO_2 Model Sensors***.

CMD_ ABC_LOGIC _RESET Req: 0xB7 0x03 Resp: <0x01>	This command turns the ABC_LOGIC ON and resets the ABC_LOGIC to its startup state. The reply <0x01> indicates that the ABC_LOGIC has been turned on. ***NOTE: This command should only be used on CO_2 Model Sensors***.
CMD_ ABC_LOGIC _OFF Req: 0xB7 0x02 Resp: <0x02>	This command turns the ABC_LOGIC OFF. The reply <0x02> indicates that the ABC_LOGIC has been turned off.

4.6. Test Commands

CMD_HALT Req: 0x95 Resp: <ACK>	This command is used strictly for testing. It tells the Sensor to put itself into error mode and act as though a fatal error has occurred. The sensor should automatically reset itself and go into Warmup Mode. See Communication Examples, below.
CMD_LOOPBACK Req: 0x00 <data_bytes> Resp: <data_bytes>	This command is used strictly for testing. The data_bytes (up to 16 bytes) following the 0x00 command are echoed back in the response packet.

4.7. Stream Data

Upon power-up, the sensor starts streaming gas concentration data out the UART. This data stream is either two or three bytes, depending upon the sensor model. If any UART command is sent to the sensor, this data streaming is stopped. In order to resume data streaming, without power-cycling the sensor, this command should be given.

CMD_STREAM_DATA Req: 0xBD Resp: <data bytes>	Start streaming gas concentration data after each data collection cycle. The response data stream is either 2 or 3 bytes, depending upon sensor model. For models returning two bytes, the format is < msb, lsb >. For models returning three bytes, the data is the actual ppm in the format <lsb, mid, msb >. In both formats, the data is non-negative and bounded.

5. UART Communication Examples

The following examples illustrate request and response packets with the UART Tsunami-Lite Communication Protocol. Requests and responses are expressed in hexadecimal bytes. The <command> portion of a request and the <response_data> are in bold type.

5.1. UART Read Gas PPM

Req> FF FE 02 **02 03** Resp> FF FA 02 **50 02**	In the request "02 03" is CMD_READ GAS_PPM (see Command Reference, above.) In the response "50 02" is a 2-byte binary value, least significant byte first., giving the gas PPM as 592 PPM (592 = 0x0250) For models that require the result be multiplied by 16, the actual PPM would be 592 * 16 = 9472.

5.2. UART CMD_STATUS to Verify Normal Operation

Req> FF FE 01 **B6** Resp> FF FA 01 **00**	In the request, "B6" is CMD_STATUS (see Command Reference, above.) In the response, "00" is the status byte. The zero value indicates that the Sensor is in normal mode where it is measuring gas PPM. It is not in warm-up mode, it is not in calibration mode, and it is not in an error condition. Further examples of CMD_STATUS are given in the examples below.

5.3. UART Read and Update Elevation

In this set of interchanges we first read the Sensor's elevation parameter and find it is set at 1000 ft. Then we change the elevation setting to 2500 ft. Then we read back the new elevation setting and verify that it is set to 2500 ft.

Req 1> FF FE 02 **02 0F** Resp1>.FF FA 02 **E8 03** Req 2> FF FE 04 **03 0F C4 09**	In request 1, "02 0F" is CMD_READ, ELEVATION (see Command Reference, above.) In the first response, "E8 03" is the elevation, 1000 ft (1000 = 0x03E8). In request 2, "03 0F" is CMD_UPDATE, ELEVATION,

	and "C4 09" is the elevation, 2500 ft (2500 = 0x09C4).
Resp2> FF FA **00**	The second response is an <ACK>, since the length is 0x00.
Req 3> FF FE 02 **02 0F** Resp3> FF FA 02 **C4 09**	The third request and response are formatted just like the first, reading back the new elevation setting, 2500 ft.

5.4. UART Error Simulation with Recovery

In this set of interchanges we first verify that the Sensor is operating normally. Then we send a command that forces the sensor into an error state. The Sensor automatically recovers by resetting itself, and going into Warmup Mode. We then send the command to skip warm-up, thus putting the sensor back into the normal state.

Req 1> FF FE 01 **B6**	Req 1: CMD_STATUS.
Resp1> FF FA 01 **00**	Resp1: status byte is 0x00. Sensor is in normal mode, measuring gas PPM.
Req 2> FF FE 01 **95**	Req 2: CMD_HALT. Puts sensor in error mode.
Resp2> FF FA **00**	Resp2: <ACK>
Req 3> FF FE 01 **B6**	Req 3: CMD_STATUS.
Resp3> FF FA 01 **02**	Resp3: status byte is 0x02. Bit 1 high indicates Sensor is in warm-up mode. (If CMD_STATUS is sent quickly enough, the sensor may respond with 0x01, indicating the brief error state prior to reset.) Wait several seconds.
Req 4> FF FE 01 **B6**	Req 4: CMD_STATUS
Resp4> FF FA 01 **00**	Resp4: status byte is 0x00. Sensor is in normal mode, measuring gas PPM.

5.5. UART Zero Calibration

In this set of interchanges we run a zero calibration on the Sensor. Before sending any commands we start flowing a zero gas, like nitrogen, to the Sensor. Then we verify that the sensor is in normal operating mode, since calibration will not work if the sensor is not in normal operating mode. Then we send the zero calibration command to start the calibration process. We check the Sensor's status and see that it is in calibration mode. Later we check the status again and see that the Sensor has finished calibration and returned to normal operating mode.

Req 1> FF FE 01 **B6**	Req 1: CMD_STATUS.
Resp1> FF FA 01 **00**	Resp1: status byte is 0x00. Sensor is in normal mode, measuring gas PPM.
Req 2> FF FE 01 **97**	Req 2: CMD_ZERO_CALIBRATE. Starts the calibration process.
Resp2> FF FA **00**	Resp2: <ACK>
	Wait 2 – 4 seconds.
Req 3> FF FE 01 **B6**	Req 3: CMD_STATUS.
Resp3> FF FA 01 **04**	Resp3: status byte is 0x04. Sensor is in calibration mode.
	Wait 15 seconds and repeat Req 4 at intervals of 15 seconds until sensor is out of calibration (return byte is 0x00).
Req 4> FF FE 01 **B6**	Req 4: CMD_STATUS.
Resp4> FF FA 01 **00**	Resp4: status byte is 0x00. Sensor is in normal mode, measuring gas PPM.

Appendix A. Summary of Commands

CMD_READ Commands

Command	Request	Response
CMD_READ	0x02 <data ID>	<data>, [… <data>]
CMD_READ_CO2_PPM	0x02 0x03	<ppm_lsb> <ppm_msb>
CMD_READ_SERIAL_NUMBER	0x02 0x01	[ASCII, 15 bytes, null padded]
CMD_READ_COMPILE_SUBVOL	0x02 0x0D	[3-byte ASCII string]
CMD_READ_COMPILE_DATE	0x02 0x0C	[6-byte ASCII string]
CMD_READ_ELEVATION	0x02 0x0F	<elev_lsb> <elev_msb>

CMD_UPDATE Commands

Command	Request	Response
CMD_UPDATE_ELEVATION	0x03 0x0F <elev_lsb> <elev_msb>	<ACK>

WARMUP Command

Command	Request	Response
CMD_WARM	0x84	<ACK> or <no response>

CALIBRATION Command

Command	Request	Response
CMD_ZERO_CALIBRATE	0x97	<ACK>

STATUS and OPERATING Commands

Command	Request	Response
CMD_STATUS	0xB6	<status>
CMD_IDLE_ON	0xB9 0x01	<ACK>
CMD_IDLE_OFF	0xB9 0x02	<ACK>
CMD_ABC_LOGIC	0xB7 0x00	<abc_state>
CMD_ABC_LOGIC_ON	0xB7 0x01	<0x01>
CMD_ABC_LOGIC_RESET	0xB7 0x03	<0x01>
CMD_ABC_LOGIC_OFF	0xB7 0x02	<0x02>

TEST Commands

Command	Request	Response
CMD_HALT	0x95	<ACK>
CMD_LOOPBACK	0x00 <data_bytes>	<data_bytes>

CMD_STREAM_DATA

Command	Request	Response
CMD_STREAM_DATA	0xBD	<data_bytes>

Appendix B. IEEE Floating Point Format

Some Sensor commands use data formatted as 4-byte, single precision, IEEE floating point, most significant byte first (big endian.). Following is a description of that numerical format.

```
//*************************************************************************//
//                                                                         //
//            IEEE 754 4-BYTE FLOATING POINT FORMAT (BIG ENDIAN)           //
//                                                                         //
//      <-- Byte #0 --> <-- Byte #1 --> <-- Byte #2 --> <-- Byte #3 -->    //
//      ---------------------------------------------------------------    //
//     |7 6 5 4 3 2 1 0|7 6 5 4 3 2 1 0|7 6 5 4 3 2 1 0|7 6 5 4 3 2 1 0|   //
//      ---------------------------------------------------------------    //
//     |S|   Exponent    |                   Mantisa                   |   //
//      ---------------------------------------------------------------    //
//    > 1 <----- 8 -----> <------------------- 23 -------------------->    //
//                                                                         //
//      Byte 0 is the most significant byte.                               //
//      Byte 3 is the least significant byte.                              //
//                                                                         //
//      The S bit (located at byte 0 bit 7) flags the sign of the          //
//      floating point number. This format does NOT use two's              //
//      complement encoding. The S bit is defined as follows:              //
//          0 => Positive                                                  //
//          1 => Negative                                                  //
//                                                                         //
//      The exponent (located in bytes 0 and 1) is 8 bits long and         //
//      is positive biased. In the special case of a zero exponent,        //
//      the entire value of the floating point number is said to be        //
//      zero and all bits should be cleared.                               //
//                                                                         //
//      The mantissa (located in bytes 1 through 3) is 23 bits long,       //
//      but carries 24 bits of information. The implied bit is             //
//      located in the most significant (bit 23) position. If the          //
//      exponent is zero, bit 23 (and all other bits) are clear. If        //
//      the exponent is non-zero, bit 23 is set.                           //
//                                                                         //
//      When calculating the value of a floating point number that         //
//      has been stored in this format, one assigns the value 0.5          //
//      (1/2) to bit 23, 0.25 (1/4) to bit 22, 0.125 (1/8) to bit          //
//      21, and so on. If the exponent is non-zero, and so the             //
//      implied bit 23 is set, the value will fall between one half        //
//      and below one.                                                     //
//                                                                         //
```

```
//      This number (between 0.5 and 1.0) is then multiplied by two         //
//      raised to the (exponent-126) power. For example, if the             //
//      exponent contains the binary value 127 and the mantissa is          //
//      all zeros (except for the implied bit 23), the value this           //
//      number is 0.5*2^(127-126) = 1.0 (-1.0 if the sign bit is set).      //
//                                                                          //
//**************************************************************************//
```