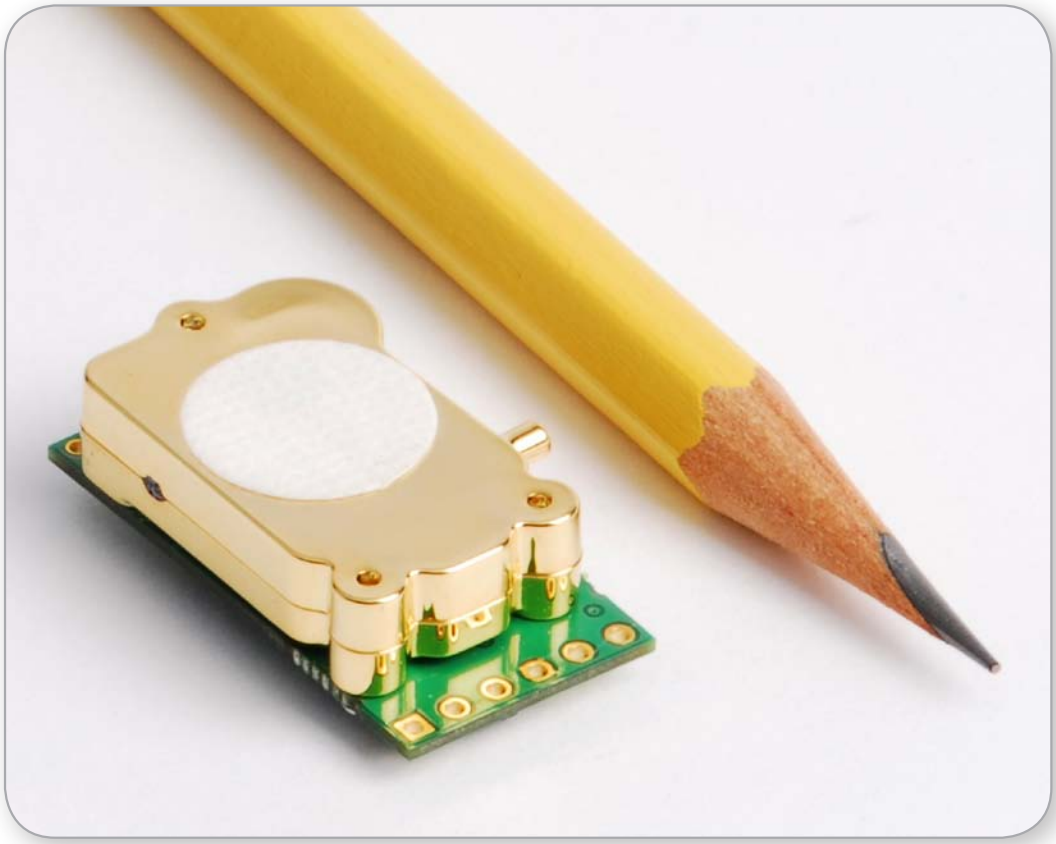
TELAIRE®

User Instructions
T67xx Rev-003

T67XX CO_2 Sensor Module



Amphenol
Advanced Sensors

REVISION RECORD

REVISION 1	
ORIGINATOR - Norman Hannotte	RELEASED - Feb.19.2014
Description of changes:	
- Initial Document release	
REVISION 2	
ORIGINATOR - Norman Hannotte	RELEASED - July.1.2014
Description of changes:	
- Correction to I^2C pin-out	
- Added example code, customer support sections	
- Add details of ABC Logic™ on/off	
- Added detail to installation / mounting section	
- Addition of Operation details, safety, conversion of PWM to analog signal sections	
- Remove watermark and preliminary markings	
REVISION 3	
ORIGINATOR - David Cooper	RELEASED - May.13.2016
Description of changes:	
- Added clarification for I^2C communications	
- Added examples for I^2C communications	

T67XX User Instructions Table of Contents

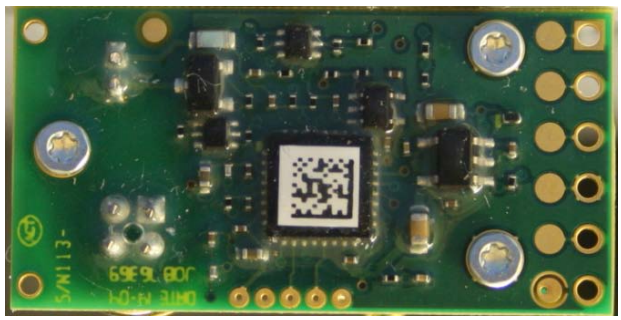
T67XX CO_2 Sensor Module

OVERVIEW

The purpose of this document is to outline the required interface design and communication protocol for the T67xx CO_2 Sensor. The intended audience is any developer who wishes to query the sensor for information utilizing either the I^2C, PWM or UART interfaces.

INTERFACE CONNECTOR

The six pin through hole connector on the PCB is were power and IO for the sensor are located, see Figure 1 below. A six pin, 0.1" header must be installed in order to connect the sensor to a controller. There are several different I/O configurations that are supported by the sensor, and are determined by the voltage that is measured on pin 6 of the connector at startup.



1. TX / SDA
2. RX / SCL
3. V++
4. GND
5. PWM
6. CTRL / TEST

Figure 1 - T67XX Interface Connector

Note: Precautions should be taken to observe specified limits and prevent damage from electrostatic discharge or rough handling. Please refer to ANSI/ESD S20. 20-1999 for more information on preventing ESD damage and IPC 610 Rev D for more information on proper electronic assembly practices. In addition to this, the sensor does not have internal reverse polarity protection. Care should be taken to connect the sensor to the controller in the correct wiring configuration to avoid damage.

Pin 6 is left unconnected by the user and will therefore be left floating. It will be pulled up by an internal 1MΩ resistor. In this condition;

T67XX CO_2 Sensor Module - Configuration #1

Table 1 – I/O Pin Configuration #1

Pin	Description
1	UART TX (output from sensor)
2	UART Rx (input to sensor)
5	PWM output at approximately 1Hz

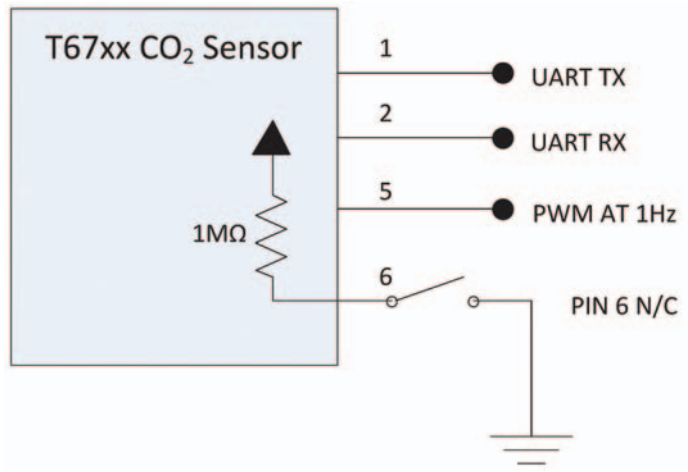


Figure 2 - Simplified Schematic I/O configuration #1

In this condition the sensor implements an I²C interface. The default conditions are;

- The sensor only acts as a slave device
- I²C 7-bit addressing is used. The default slave address is 21 (0x15)
- I²C 100kbit/s Standard Mode

Note: There is an internal pull up resistor on pin 1 of the I²C interface. Customer will need to provide an external pull up resistor on pin 2 with a recommended value of 4.7k. I²C interface can operate at both 3.3V and 5V logic levels.

Pin 6 is pulled to ground by a resistor between 10kΩ and 100kΩ

T67XX CO_2 Sensor Module - Configuration #2

Table 2 - I/O Pin Configuration #2

Pin	Description
1	I²C SDA (Serial Data Line)
2	I²C SCL (Serial Clock Line)
5	PWM output at approximately 25kHz

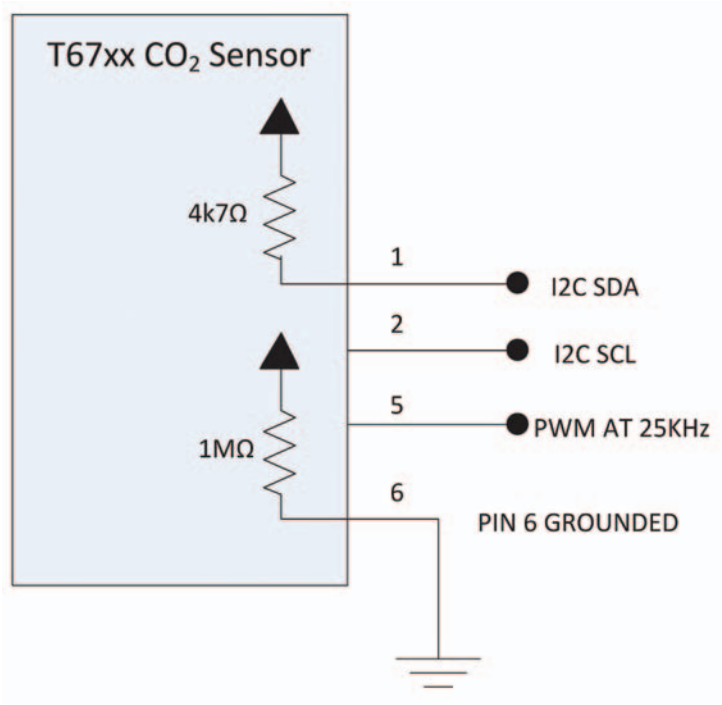


Figure 3 - Simplified Schematic I/O configuration #2

In this condition the sensor implements an I²C interface. The default conditions are;

- The sensor only acts as a slave device
- I²C 7-bit addressing is used. The default slave address is 21 (0x15)
- I²C 100kbit/s Standard Mode

Note: There is an internal pull up resistor on pin 1 of the I²C interface. Customer will need to provide an external pull up resistor on pin 2 with a recommended value of 4.7k. I²C interface can operate at both 3.3V and 5V logic levels.

Pin 6 is pulled to ground by a resistor between 10kΩ and 100kΩ

T67XX CO_2 Sensor Module - Configuration #3

Table 3 - I/O Pin Configuration #3

Pin	Description
1	UART TX (output from sensor)
2	UART Rx (input to sensor)
5	Becomes a test input for Telaire and should be left unconnected by the user
6	Becomes an output pin used to drive an RS485 transceiver

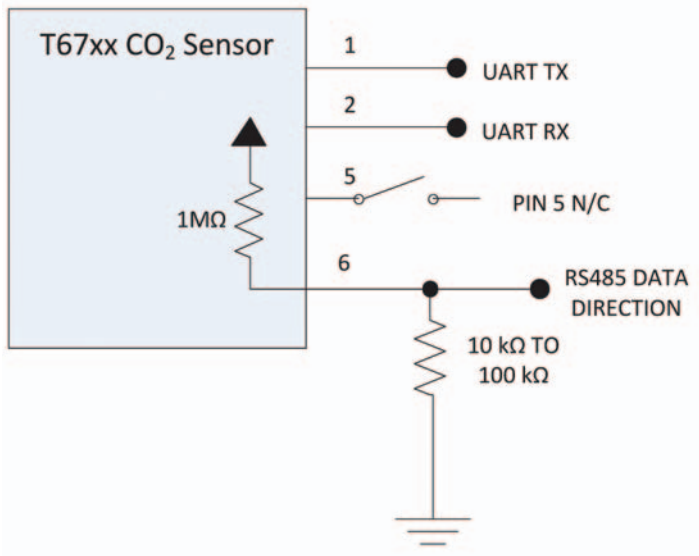


Figure 3 - Simplified Schematic I/O configuration #2

The condition described by (3) above is useful if the sensor is used in an RS485 dropped node network configuration. This network would be supplied by the user. In this condition pin 6 becomes the RS-485 transceiver data-direction logic.

T67XX COMMUNICATION – MODBUS PROTOCOL

The T67xx sensor uses the Modbus protocol for all communications. The documents are freely available on the Modbus WEB site at http://www.modbus.org/specs.php.

UART (RS232/RS485)

For UART communications, reference the recommendations found in the document 'Modbus Serial Line Protocol and Implementation Guide V1.02'. This document includes detailed information on how to calculate the required CRC (Cyclical Redundancy Checking) bytes.

It is important to note that for Modbus over serial lines (i.e., RS-232 and RS-485) the user MUST INCLUDE THE CYCLICAL REDUNDANCY CHECK (CRC) fields at the end of the Modbus request. The CRC calculation is not necessary for communications over the I^2C interface.

I^2C

The I^2C implementation does not use the Serial Line protocol. The sensor does use the Modbus protocol, and wraps the message in I^2C format. Please reference to the "I^2C specification and user's manual" at the following URL for details on I^2C communication www.nxp.com/documents/user_manual/UM10204.pdf.

The sensor always operates as a slave. In the examples below the implementer using the I^2C interface will need communicate with the sensor as either a master-transmitter (for Modbus requests) or a master-receiver (for Modbus responses). The sensor will not initiate communications, i.e., it will not become a master and respond to the request. It is up to the master to read the response from the sensor.

I^2C TIMING CONSIDERATIONS

Unlike the UART interface, where the sensor's serial port controls the response back to the master device, the I^2C master can ask for the response immediately after sending the request because the I^2C master controls the clock. If the master asks for a response immediately, without giving the sensor time to formulate a response, the master will receive a string of zeros.

It is suggested that the master send the request, wait 5 to 10 milliseconds and then ask for the response. This time does depend on bus loading and board layout but carefully constructed test setups have demonstrated that the sensor can respond within 1 millisecond in controlled conditions with a data rate of 100kbps. The suggested delay of 10 milliseconds should be adequate for almost all conceivable cases.

T67XX SPECIFICATION

All information about the performance of the sensor can be found on the website. Please refer to the spec sheets found at www.telaire.com for the latest specifications.

T67XX INSTALLATION / MOUNTING

DESIGN CONSIDERATIONS

To maximize the performance of T67xx module, it is important to plan an appropriate location for the sensor at the design stage. Airflow and proper exposure to ambient air must be secured for T67xx module to ensure optimal performance. Inadequate airflow will deteriorate the response time of the sensor. Also, avoid excess heat.

The sensor is designed for benign environments. The performance and reliability may be negatively affected in environments that contain corrosive or caustic gases including but not limited to Ammonia, Chlorine, NOx and Ozone. Care must be taken to ensure that the sensor is not exposed to these compounds under any operating condition

INSTALLATION

The T67xx module is a sensitive electronics assembly, so effort should be made to minimize exposure to excess heat from any type of soldering operation. Excess exposure to heat from installation is known to create small shift in calibration of the sensor. Although the ABC Logic™ algorithm will correct these minor fluctuations within the first 24 hours of operation, the user should minimize overexposure to heat when soldering to negate unexpected operation after installation. Typically, an expose of not more than 10 sec. using a soldering iron set to 750°F (400°C) will minimize the influence of heat on the measurement of the T67xx module. The use of low temperature solder is also recommended.

It is recommended to handle the sensor by the edges of the PCBA. Any stress applied to the PCB or the gold OBA assembly can cause mechanical stress that will create temporary shift in the calibration of the sensor. Although he ABC Logic™ algorithm will correct for these shifts in the first 24 hours of operation, extra care in handling will minimize the risk of variations that can be seen out of the box.

ESD PRECAUTIONS

Precautions should be taken to observe specified limits and prevent damage from electrostatic discharge or rough handling. Please refer to ANSI/ESD S20. 20-1999 for more information on preventing ESD damage for more information on proper electronic assembly practices.

T67XX OPERATION DETAILS

ABC LOGIC™

Automatic Background Logic, or ABC Logic™, is a patented self-calibration technique that is designed to be used in applications where concentrations will drop to outside ambient conditions (400 ppm) at least three times in a 7 days, typically during unoccupied periods. Full accuracy to be achieved utilizing ABC Logic™. With ABC Logic™ enabled, the sensor will typically reach its operational accuracy after 24 hours of continuous operation at a condition that it was exposed to ambient reference levels of air at 400 ppm CO_2. Sensor will maintain accuracy specifications with ABC Logic™ enabled, given that it is at least four times in 21 days exposed to the reference value and this reference value is the lowest concentration to which the sensor is exposed. ABC Logic™ requires continuous operation of the sensor for periods of at least 24 hours.

Note: Applies when used in typical residential ambient air. Consult Telaire if other gases or corrosive agents are part of the application environment.

ENVIRONMENTAL

Operating Temperature: 50°F to 122°F (10°C to 50°C), 0 to 95% RH, non-condensing

Storage Temperature: -22°F to 158°F (-30°C to 70°C)

ABSOLUTE MAXIMUM RATINGS

In order to avoid damage to the sensor, the following parameters should never be exceeded at any time during operation:

Parameter	Min (V)	Max (V)
Supply Voltage (V+, pin 3)	-0.3	7
Ground (V-, pin 4)	-0.3	0.3
TX / SDA, RX / SCL (pin 1,2)	-0.3	7
MD DIR / PWM (pin 5)	-0.3	7
UART/I^2C Select (pin 6)	-0.3	3

POWER-ON SEQUENCE

The sensor is capable of responding to commands after power on, but operational accuracy of sensor won't happen until 120 sec have elapsed. The sensor will reach full accuracy / warm up after 10 min. of operation.

POWER SUPPLY REQUIREMENTS

Supply Voltage: +4.5 – 5.5VDC

Average Current*: 20mA

Peak Current*: 155mA

*Based on typical values at a nominal 5VDC input

EVALUATION / DEMONSTRATION KITS

Evaluation kits are available in order to help customers develop their application. The kit includes a sample of the sensor, cable and software that will enable the user to better understand the performance of the sensor in their intended design. Please contact Telaire for further details or visit the website at www.telaire.com.

T67XX PWM OUTPUTS

POWER-ON SEQUENCE

The T67xx sensor has a selectable Pulse Width Modulation (PWM) output feature on Pin5, based on the state of Pin 6 during power on. The two types of supported PWM operate at ~1Hz PWM and 25k Hz and are proportional to the range of the sensor as determined by the model. For example, if the model is a T6713-5k, the units PWM output will be proportional to a 0 to 5000 ppm output range. For PN T6713, the unit will have an output proportional to a 0 – 2000 ppm measurement. A 4.7K pull up resistor to a voltage reference can be added if desired.

The slow ~1Hz PWM output option allows the user to measure the duration of the pulse, and correlate this to a CO_2 measurements.

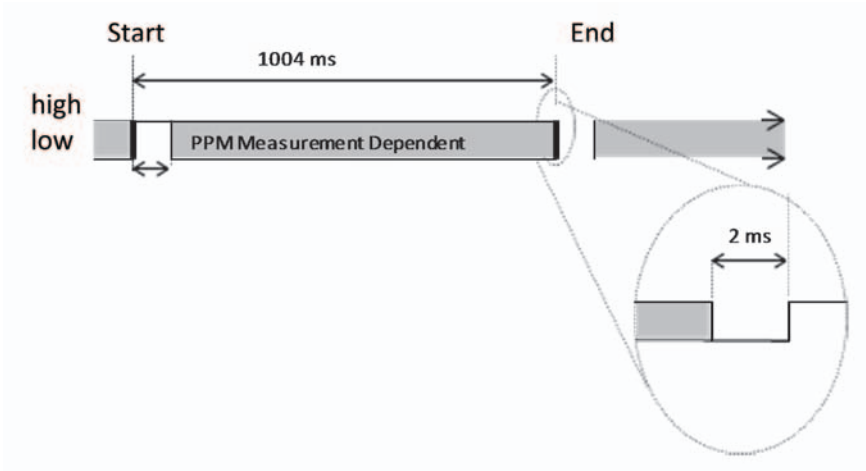


Figure 5 - Details of ~1Hz PWM

In order to convert the pulse to a reading in PPM, the user should use the following equation:

$$PPM = (t_{pulse} - 2) * 2 \text{ for } 0 - 2000 \text{ ppm models}$$
$$PPM = (t_{pulse} - 2) * 5 \text{ for } 0 - 5000 \text{ ppm models}$$

Where:

$$t_{pulse} = Measured\ value\ of\ pulse$$
$$PPM = Measured\ CO_2\ Value$$

T67XX PWM OUTPUTS

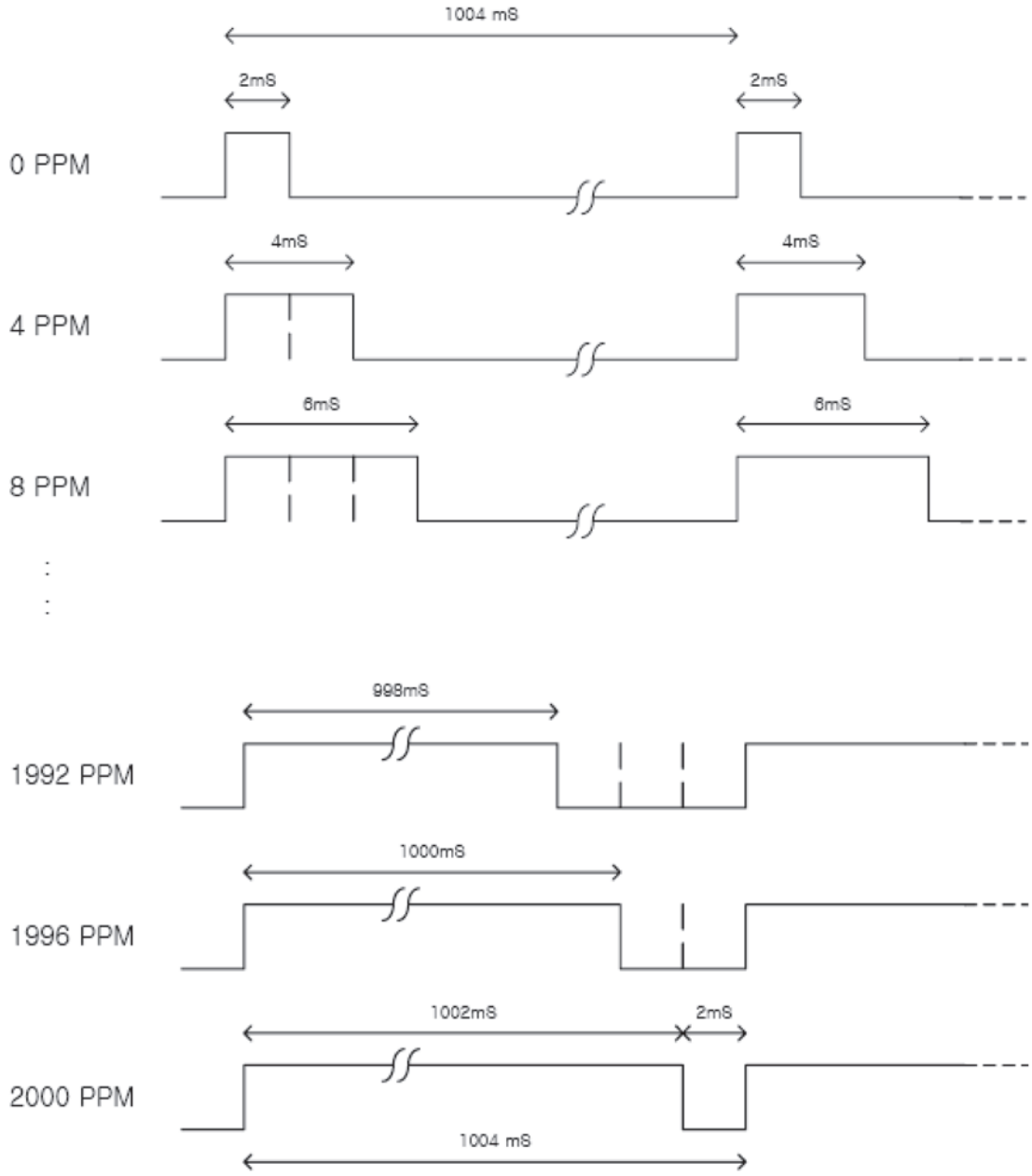


Figure 6 - Example of the PWM for -2K model

T67XX PWM OUTPUTS

Potential circuit design to convert 25 kHz PWM to an analog output for I/O configuration # 2 (I^2C, PWM at 25 kHz):

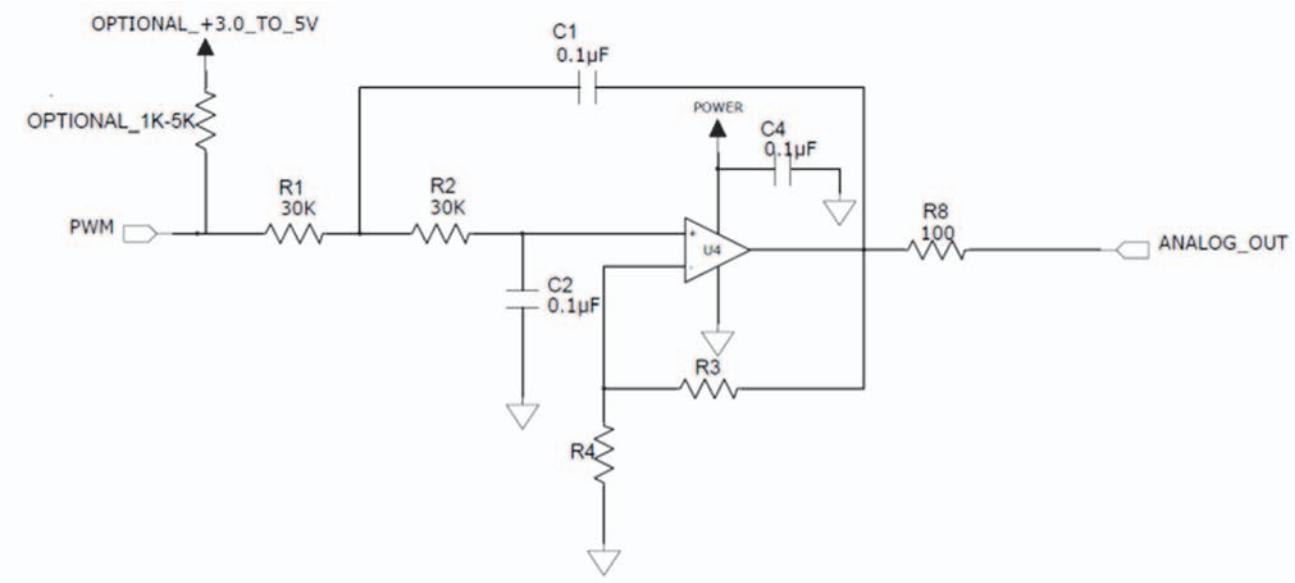


Figure 7 - Potential Circuit Design for 25 kHz PWM output to Analog Signal Conversion

The 25 kHz PWM can be filtered to create an analog output. Above is one filtering circuit where R3 and R4 will set the gain of the amplifier.

T67XX SAFETY

DISCLAIMER

Telaire makes no warranty, representation or guarantee regarding the suitability of this product for any particular application, including safety critical applications. Nor does Telaire assume any liability arising out of the application or use in any product or circuit. Telaire specifically disclaims all liability without limitation consequential or incidental damages. No statuary or fitness for particular purpose shall be implied.

WARNING! Before installing the product, review the product data sheet and this application guide. The product shall be used only within power supply and electrical input and output limits as specified by the datasheet and application guide. Improper use of the product may result in product damage and property loss and/or personal injury.

SAFETY WHILE IN USE

Before installing, handling, using, or servicing this product, please consult the data sheet and application notes. The product shall be used only within power supply and electrical input and output limits as specified by the datasheet and application guide. Improper use of the product may result in product damage and property loss and/or personal injury. In use of the product, the customer has sole responsibility for designing and implementing a solution which will ensure safe operation (including review of appropriate reliability or required redundancy, mitigation of failure modes, and/or meeting appropriate standards). The customer is

responsible for review of any special conditions for use including, but not limited to, environmental conditions, electrical supply, residual risk, etc.). The sensor is designed for benign environments. The performance and reliability may be negatively affected in environments that contain corrosive or caustic gases including but not limited to Ammonia, Chlorine, NOx and Ozone.

MATERIAL CONTENTS

ROHS and REACH declaration of conformity are available upon request. Please contact customer service for more details.

T67XX COMMAND SUMMARY

The following commands can be sent to the sensor.

Table 4 - Modbus Command Summary

Name	Modbus Register	Register Address	Data type	Description
FIRMWARE_REVISION	Input (RO)	‘5001’D ‘1389’H	uint16_t	Returns the firmware revision from the sensor
STATUS	Input (RO)	‘5002’D ‘138A’H	uint16_t	Returns a status register from the sensor. Additional details are below
GAS PPM	Input (RO)	‘5003’D ‘138B’H	uint16_t	The current gas ppm calculation
RESET DEVICE	Coils (WO)	‘1000’D ‘03E8’H	uint16_t	Reset the sensor over the Modbus network
START SINGLE POINT CAL	Coils (WO)	‘1004’D ‘03EC’H	uint16_t	Start a single-point calibration
SLAVE ADDRESS	Holding (RW)	‘4005’D ‘0FA5’H	uin16_t	Change of sensor address (default address is ‘21’D (‘15’H)
ABC LOGIC™ ENABLE / DISABLE	Coils (RW)	‘1006’D ‘03EE’H	uin16_t	Enable or disable ABC Logic™

Where RO is Read Only, WO is Write Only and RW is Read or Write.

There are examples given in the sections following sections.

T67XX COMMAND SUMMARY (cont.)

FIRMWARE REVISION

This command will return the current firmware revision of the sensor. The user is required to append the CRC for each of the command over UART.

Use the Modbus Read Input Registers function (4) and read one (1) register at address ‘5001’D (‘1389’H).

Example:

UART (all bytes in hexadecimal)

Modbus Request (UART)

‘15’H	Slave address
‘04’H	Function Code
‘13’H	Starting Address (MSB)
‘89’H	Starting Address (LSB)
‘00’H	Number of registers to read (MSB)
‘01’H	Number of registers to read (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Modbus Response (UART)

‘15’H	Slave address
‘04’H	Function code
‘02’H	Byte count
xx	Status (MSB)
xx	Status (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Example 1 - Modbus request/response to read the firmware revision (UART)

T67XX COMMAND SUMMARY (cont.)

I²C (all bytes in hexadecimal)

The default I²C slave address is ‘21’D (‘15’H) and not shown.

Modbus Request (Master-Transmitter/Slave-Receiver)

‘04’H	Function Code
‘13’H	Starting Address (MSB)
‘89’H	Starting Address (LSB)
‘00’H	Number of registers to read (MSB)
‘01’H	Number of registers to read (LSB)

Modbus Response (Master-Receiver/Slave-Transmitter)

‘04’H	Function code
‘02’H	Byte count
xx	status (MSB)
xx	status (LSB)

Example 2 - Modbus request/response to read the firmware revision (I²C)

It is important to note that the sensor is a slave device only. The user must implement a master I²C device to facilitate reading and writing to the sensor.

T67XX COMMAND SUMMARY (cont.)

STATUS

This command returns a register with the status of various functions on the sensor. The user must verify that no error condition exists in the sensor. Also, user must verify that the sensor has completed the warm-up stage.

Use the Modbus Read Input Registers function (4) and read one (1) register at address ‘5002’D (‘138A’H).

Modbus Request (UART)

‘15’H	Slave Address (default is 21)
‘04’H	Function code
‘13’H	Starting address (MSB)
‘8A’H	Starting Address (LSB)
‘00’H	Input registers to read (MSB)
‘01’H	Input registers to read (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Modbus Response (UART)

‘15’H	Slave address
‘04’H	Function code
‘02’H	Byte count
xx	Status (MSB)
xx	Status (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Example 3 – Modbus request/response to read the STATUS register (UART)

Modbus Request (I^2C) (Master-Transmitter/Slave-Receiver)

‘04’H	Function code
‘13’H	Starting address (MSB)
‘8A’H	Starting Address (LSB)
‘00’H	Input registers to read (MSB)

T67XX COMMAND SUMMARY (cont.)

Modbus Response (I^2C) (Master-Receiver/Slave-Transmitter)

'04'H	Function code
'02'H	Byte count
xx	Status (MSB)
xx	Status (LSB)

Example 4 – Modbus request/response to read the STATUS register (I^2C)

The STATUS register is a bit-vector where each bit represents the status of some function within the sensor. Not all bits are assigned.

Bit position	HEX	Comments
xxxxxxxx, xxxxxxx1	'0001'H	Error condition
xxxxxxxx, xxxxxx1x	'0002'H	Flash error
xxxxxxxx, xxxxx1xx	'0004'H	Calibration error
xxxxxxxx, xxxx1xxx	'0008'H	NA
xxxxxxxx, xxx1xxxx	'0010'H	NA
xxxxxxxx, xx1xxxxx	'0020'H	NA
xxxxxxxx, x1xxxxxx	'0040'H	NA
xxxxxxxx, 1xxxxxxx	'0080'H	NA
xxxxxxx1, xxxxxxxx	'0100'H	RS-232
xxxxxx1x, xxxxxxxx	'0200'H	RS-485
xxxxx1xx, xxxxxxxx	'0400'H	I^2C
xxxx1xxx, xxxxxxxx	'0800'H	Warm-up mode
xxx1xxxx, xxxxxxxx	'1000'H	NA
xx1xxxxx, xxxxxxxx	'2000'H	NA
x1xxxxxx, xxxxxxxx	'4000'H	NA
1xxxxxxx, xxxxxxxx	'8000'H	Single point calibration

For error conditions, a '1' indicates an error; a '0' indicates no error. Flash error is fatal (i.e., there is no recovery). Calibration errors can be cleared by running the calibration procedure again with successful results.

For calibration conditions, a '1' indicates that the calibration cycle is currently in progress. No other calibration cycle can start while one is currently in progress and will result in an error being reported by the Modbus response to the new calibration request.

If the Warm-up bit is set the sensor is in a mode where internal registers are being initialized and gas (ppm) data is not necessarily correct.

T67XX COMMAND SUMMARY (cont.)

GAS PPM

This command reports the current gas measurement in ppm.

Use the Modbus Read Input Registers function (4) and read one (1) register at address ‘5003’D (‘138B’H).

Modbus Request (UART)

‘15’H	Slave Address (default is 21)
‘04’H	Function code
‘13’H	Starting address (MSB)
‘8B’H	Starting Address (LSB)
‘00’H	Input registers to read (MSB)
‘01’H	Input registers to read (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Modbus Response (UART)

‘15’H	Slave Address (default is 21)
‘04’H	Function code
‘02’H	Byte count
xx	MSB of the 16-bit data
xx	LSB of the 16-bit data
xx	CRC (LSB)
xx	CRC (MSB)

Example 5 – Modbus request/response to read the GAS PPM register (UART)

Modbus Request (I^2C) (Master-Transmitter/Slave-Receiver)

‘04’H	Function code
‘13’H	Starting address (MSB)
‘8B’H	Starting Address (LSB)
‘00’H	Input registers to read (MSB)
‘01’H	Input registers to read (LSB)

Modbus Response (I^2C) (Master-Receiver/Slave-Transmitter)

‘04’H	Function code
‘02’H	Byte count
xx	MSB of the 16-bit data
xx	LSB of the 16-bit data

Example 6 – Modbus request/response to read the GAS PPM register (I^2C)

T67XX COMMAND SUMMARY (cont.)

To calculate the gas ppm do the following;

ppm = MSB * 256 + LSB

For example, if the Modbus (UART) response was

‘15’H	Slave Address (default is 21)
‘04’H	Function code
‘02’H	Byte count
‘01’H	MSB of the 16-bit data
‘9F’H	LSB of the 16-bit data
xx	CRC (LSB)
xx	CRC (MSB)

Example 7 – Calculating GAS ppm

Then the gas could be calculated as

1*256 + 159 = 415

Where

‘01’H = ‘1’D and ‘9F’H = ‘159’D

T67XX COMMAND SUMMARY (cont.)

RESET DEVICE

The sensor can be reset (though this is not normally recommended) over the Modbus interface. The reset is immediate and there is no response. The sensor will act as if the power was cycled.

Use the Modbus Write Single Coil function (5) and write '00FF'H to the register at address '1000'D (03E8'H) to reset the sensor.

Modbus Request (UART)

'15'H	Slave Address (default is 21)
'05'H	Function code
'03'H	Output address (MSB)
'E8'H	Output Address (LSB)
'FF'H	Output value (MSB)
'00'H	Output value (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Example 8 - Modbus request to reset the sensor (UART)

Modbus Request (I^2C) (Master-Transmitter/Slave-Receiver)

'05'H	Function code
'03'H	Output address (MSB)
'E8'H	Output Address (LSB)
'FF'H	Output value (MSB)
'00'H	Output value (LSB)

Example 9 – Modbus request to reset the sensor (I^2C)

Note: There is no response. The sensor resets immediately.

T67XX COMMAND SUMMARY (cont.)

START SINGLE POINT CALIBRATION

This command starts the single point calibration routine.

The single-point calibration routine is usually done at ambient conditions (~500ppm, 25 °C) and takes several minutes to complete after being started (~6 minutes). The sensor can be queried during this time for status and current gas ppm readings. The user can check on the status of the calibration by reading the status register and noting if the single-point calibration bit is set. The calibration can be stopped before completing. See examples.

Use the Modbus Write Single Coil function (5) and write '00FF'H to the register at address '1004'D (03EC'H) to start the calibration. Write '0000'H during calibration to stop the calibration function.

Modbus Request (UART)

'15'H	Slave Address (default is 21)
'05'H	Function code
'03'H	Output address (MSB)
'EC'H	Output Address (LSB)
'FF'H	Output value (MSB)
'00'H	Output value (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Modbus Response (UART)

'15'H	Slave Address (default is 21)
'05'H	Function code
'03'H	Output address (MSB)
'EC'H	Output Address (LSB)
'FF'H	Output value (MSB)
'00'H	Output value (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Example 10 - Modbus request/response to start Single Point Calibration (UART)

Modbus Request (I^2C)

'05'H	Function code
'03'H	Output address (MSB)
'EC'H	Output Address (LSB)
'FF'H	Output value (MSB)
'00'H	Output value (LSB)

T67XX COMMAND SUMMARY (cont.)

Modbus Response (I^2C)

‘05’H	Function code
‘03’H	Output address (MSB)
‘EC’H	Output Address (LSB)
‘FF’H	Output value (MSB)
‘00’H	Output value (LSB)

Example 11 – Modbus request/response to start Single Point Calibration (I^2C)

Calibration takes several minutes. The status of the single point can be determined by reading the status register.

The calibration command cannot be restarted but can be stopped as detailed in the following example.

Modbus Request (UART)

‘15’H	Slave Address (default is 21)
‘05’H	Function code
‘03’H	Output address (MSB)
‘EC’H	Output Address (LSB)
‘00’H	Output value (MSB)
‘00’H	Output value (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Modbus Response (UART)

‘15’H	Slave Address (default is 21)
‘05’H	Function code
‘03’H	Output address (MSB)
‘EC’H	Output Address (LSB)
‘00’H	Output value (MSB)
‘00’H	Output value (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Example 12 – Modbus request/response to stop Single Point Calibration (UART)

T67XX COMMAND SUMMARY (cont.)

Modbus Request (I^2C)

‘05’H	Function code
‘03’H	Output address (MSB)
‘EC’H	Output Address (LSB)
‘00’H	Output value (MSB)
‘00’H	Output value (LSB)

Modbus Response (I^2C)

‘05’H	Function code
‘03’H	Output address (MSB)
‘EC’H	Output Address (LSB)
‘00’H	Output value (MSB)
‘00’H	Output value (LSB)

Example 13 – Modbus request/response to stop Single Point Calibration (I^2C)

Note: The UART Modbus request/response is not shown.

T67XX COMMAND SUMMARY (cont.)

CHANGE SLAVE ADDRESS

It is possible to change the Modbus slave address. The change will only take effect after the sensor has been reset (e.g., over the Modbus interface) or power cycled. The change does not take effect immediately.

User the Modbus Write Single Register function (6) and write the new address to the register at address ‘4005’D (‘0FA5’H).

This example changes the current slave address to ‘16’D (‘10’H).

Modbus Request (UART)

‘15’H	Slave Address (default is 21)
‘06’H	Function code
‘0F’H	Register address (MSB)
‘A5’H	Register Address (LSB)
‘00’H	Register value (MSB) MSB will always be zero
xx	Register value (LSB) LSB should be between 1-247
xx	CRC (LSB)
xx	CRC (MSB)

Modbus Response (UART)

‘15’H	Slave Address (default is 21)
‘06’H	Function code
‘0F’H	Register address (MSB)
‘A5’H	Register Address (LSB)
‘00’H	Register value (MSB)
xx	Register value (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Example 14 – Changing the default Slave Address (UART)

Modbus Request (I^2C)

‘06’H	Function code
‘0F’H	Register address (MSB)
‘A5’H	Register Address (LSB)
‘00’H	Register value (MSB) MSB will always be zero
xx	Register value (LSB) LSB should be between 1-247

T67XX COMMAND SUMMARY (cont.)

Modbus Response (I^2C)

‘06’H	Function code
‘0F’H	Register address (MSB)
‘A5’H	Register Address (LSB)
‘00’H	Register value (MSB)
xx	Register value (LSB)

Note: It is necessary to restart the sensor, either by power cycling or writing ‘FF’H to the RESET register, before this change will take effect.

ABC LOGIC™ ENABLE / DISABLE

ABC Logic™ is manipulated through the Modbus interface by using the Write Single Coil function (5). Coils are viewed as basically switches, relays or discrete (i.e., single bit) inputs and outputs.

The ABC LOGIC CONTROL coil register is ‘1006’D (‘03EE’H).

A write of ‘FF00’H (i.e., ON) will enable the ABC Logic™ and a write of ‘0000’H (i.e., OFF) will disable the ABC Logic™ function.

For example, to enable the ABC Logic™ one would send the Modbus request;

‘05’H, ‘03’H, ‘EE’H, ‘FF’H, ‘00’H

Note that the example is shown as a simple Modbus PDU. For communications over a serial port (i.e., Modbus Serial Line PDU) the slave address would need to be prepended and the CRC appended.

Modbus Request (UART)

‘15’H	Slave Address (default is 21)
‘05’H	Function code
‘03’H	Register address (MSB)
‘EE’H	Register Address (LSB)
‘FF’H	Register value (MSB) Enable ABC Logic™
‘00’H	Register value (LSB) LSB will always be zero
xx	CRC (LSB)
xx	CRC (MSB)

T67XX COMMAND SUMMARY (cont.)

Modbus Response (UART)

‘15’H	Slave Address (default is 21)
‘05’H	Function code
‘03’H	Register address (MSB)
‘EE’H	Register Address (LSB)
‘FF’H	Register value (MSB)
‘00’H	Register value (LSB)
xx	CRC (LSB)
xx	CRC (MSB)

Example 16 – Enable ABC Logic™ (UART)

Modbus Request (I^2C)

‘05’H	Function code
‘03’H	Register address (MSB)
‘EE’H	Register Address (LSB)
‘00’H	Register value (MSB) Disable ABC Logic™
‘00’H	Register value (LSB) LSB will always be zero

Example 17 – Disable ABC Logic™ (I^2C)

T67XX COMMAND SUMMARY (cont.)

EXAMPLE CODE

READING VALUE FOR CO_2

Reading the T67xx sensor in an embedded design is fairly easy. This example assumes that the controlling micro-processor has already initialized a UART and the sensor has the default settings from the factory (e.g., the slave address is 0x15).

Because the slave address and CRC will never change the programmer can just define the entire Modbus request as constants in as array.

For example;

```
static const uint8_t Read_CO2_Cmd[8] = {
  0x15,          // assumed slave address
  0x04,          // read input register function code
  0x13,          // register address 5003 (MSB)
  0x8B,          // register address 5003 (LSB)
  0x00,          // number of registers (MSB)
  0x01,          // number of registers (LSB)
  0x46,          // CRC (LSB)
  0x70           // CRC (MSB)
};
static uint16_t Read_CO2_Cmd_Length =  (uint16_t)(sizeof(Read_CO2_Cmd)/sizeof(uint8_t));
```

The following example sends the above character array out the serial port and then delays 50ms before looking for a response. The response should be 7 bytes long and in this example will fill a data structure (i.e., char array) named input_buf[] with the result.

```
input_buf[0] = 0x15   /* slave address */
input_buf[1] = 0x04   /* function code */
input_buf[2] = 0x02   /* byte count */
input_buf[3] = 0x??   /* CO2 reading, MSB */
input_buf[4] = 0x??   /* CO2 reading, LSB */
input_buf[5] = 0x??   /* CRC, LSB */
input_buf[6] = 0x??   /* CRC, MSB */
```

The CO_2 can then be calculated by

```
input_buf[3] * 256 + input_buf[4]
```

T67XX COMMAND SUMMARY (cont.)

THE CODE SNIPPET FOLLOWS.

```
uart1_write((uint8_t*)Read_CO2_Cmd, Read_CO2_Cmd_Length); delay(50);

 uart1_read((uint8_t*)input_buf, 8);
 if ((input_buf[0]==0x15) && (input_buf[1]==0x04) && (input_buf[2]==0x02))
 {
   raw_co2 = (uint16_t)((input_buf[3]<<8) | input_buf[4]);
 }
```

Arduino I²C Example

As discussed in Section 3.2.1 the I²C needs some delay after the request for a correct response. The following is a snippet of Arduino code that can be used to test the I²C interface.

```
#define ADDR_6713  0x15 // default I²C slave address

byte data[6];
int CO2ppmValue;

void GetCO2PPM()
{
  // start I²C
  Wire.beginTransmission(ADDR_6713);
  Wire.write(0x04);
  Wire.write(0x13);
  Wire.write(0x8B);
  Wire.write(0x00);
  Wire.write(0x01);

  // end transmission
  Wire.endTransmission();

  // read report of current gas measurement in ppm after delay!
  delay(10);

  Wire.requestFrom(ADDR_6713, 4);    // request 4 bytes from slave device
  data[0] = Wire.read();
  data[1] = Wire.read();
  data[2] = Wire.read();
  data[3] = Wire.read();

  CO2ppmValue = ((data[2] & 0x3F ) << 8) | data[3];
}
```

Notes:

Customer Support Centers

U.S.A.

Sales and Services
(Repair/Calibration):
Amphenol Thermometrics, Inc.
St Marys Center
967 Windfall Road
St Marys, Pennsylvania 15857
U.S.A.
T: +1 814-834-9140
F: +1 814-781-7969

U.K.

Sales and Service:
Amphenol Thermometrics (U.K.) Limited
Crown Industrial Estate Priorswood Road
Taunton, TA2 8QY, UK
T: +44 1823-335-200

Brazil

Sales and Service
Amphenol TFC DO Brazil LTDA
Rodovia Governador Adhemar
Pereira de Barros KM 121,5 S/N
13098-396 Campinas
Sao Paulo, Brazil

U.S.A.

Technical Support:
Amphenol Thermometrics, Inc.
St Marys Center
967 Windfall Road
St Marys, Pennsylvania 15857
U.S.A.
T: +1 814-834-9140
F: +1 814-781-7969

Or

Amphenol Advanced Sensors
6860 Cortona Dr., Suite B
Goleta, CA 93117

China:
Amphenol (Changzhou)
Connector Systems
305 Room, 5D
Jintong Industrial Park
Wujin, Changzhou, Jiangsu, China
T:+86 519 8831 8080 ext. 50087
F:+86 519 8831 2601

Amphenol
Advanced Sensors

www.telaire.com

www.amphenol-sensors.com