# Introduction to Using Calex Infrared Temperature Sensors with Third Party Modbus Software

## Introduction to Modbus

Modbus is an open communication protocol commonly used in industry. It enables the transmission of data over serial lines between electronic devices.

Calex sensors use RS485, RS232 or USB hardware to transmit Modbus data.

A standard RS485 Modbus network consists of 1 Modbus Master, such as a Modbus PLC or software such as a SCADA system, and up to 247 Slave devices (such as Calex infrared temperature sensors and output modules). If USB or RS232 is used, typically only one Slave device is connected.

## Data Format

The data is sent as binary digits (bits), with a standard data rate of 9600 baud (bits per second). We can also provide other baud rates to suit special requirements – please contact Calex for more information.

Bits are usually interpreted by software in hexadecimal (base 16), with a block of 4 bits being represented by one of 16 characters from 0 to F. A pair of hexadecimal characters represents 8 bits (one byte) of data. Some Modbus software also allows values to be entered in decimal – be sure to check whether this is the case for your software.

Hexadecimal numbers are denoted by the prefix "0x" to distinguish them from decimal numbers. For example, "0x0010" is the hexadecimal number 10 (decimal 16).

The standard data format for Calex sensors is 8 data bits, no parity bits, and 1 stop bit. Modbus software can be configured to use this data format, and we can provide special alternative data formats if required.

## Storing and Accessing Data in Modbus

Information is stored in the Modbus Slave device in a series of Registers, each with its own address in the device's memory. The size of each Register is 2 bytes (1 Word, 4 hex characters) or more.

Because the Registers are sequential, it is possible to read more than one Register at the same time using a single Modbus command, if required.

To read from or write to a sensor, the Modbus Master (e.g. the SCADA software or the Modbus PLC) sends a command made up of a series of parts, and the Slave device will respond with a message of a corresponding format.

## Modbus Commands

The first part of the command is the Modbus Slave address of the sensor (called the Slave ID or Device ID in some software). Each device has its own address from 1 to 247, which must be unique on the network to prevent communication conflicts. Groups of sensors are supplied with sequential Modbus addresses, and they can be changed by the user via the sensor's configuration interface.

The format of the rest of the message depends on the type of command, which is described by a Function Code.

Calex sensors utilise some or all of the following Modbus Function Codes:

| Function Code | Action |
|---|---|
| 04 (04 hex) | Read register |
| 03 (03 hex) | Read register |
| 06 (06 hex) | Write single register |
| 16 (10 hex) | Write multiple register |
| 22 (16 hex) | Mask write register |
| 23 (17 hex) | Read/Write register |

The Master then tells the Slave which Register address to read from or write to, how much data there is, and (if writing) the value to be written.

At the end of every Modbus command, there are two bytes used for error detection, these are known as the Cyclic Redundancy Check (CRC). The Modbus Slave also calculates the CRC and compares it to the CRC from the Master. If the CRCs are different, an error will result. Modbus software handles CRC calculation automatically.

## List of Modbus Registers (Modbus Map)

A Modbus Map is a list of Register addresses that describes what the data is (e.g. the filtered temperature); where the data is stored in the device's memory (the register address), the length of the register, and how the data is stored (for example the emissivity setting 0.95 is stored in Calex sensors as 9500, and the measured temperature 23.5°C is stored as 235).

This list of registers can be found in the instruction manual of each Calex Modbus infrared temperature sensor.

# Examples of Using Modbus with Calex Infrared Temperature Sensors

## Example 1 - To Read the Filtered Object Temperature

In this example, a PyroMiniBus sensor with address 17 is the Modbus Slave.

The Modbus Master sends the command 11 03 000E 0001 59E7

| | |
|---|---|
| 11 | Slave Address<br>(11 hex is the Modbus address of the sensor i.e. 17 in decimal) |
| 03 | Function Code 3<br>(Read Register) |
| 000E | Data Address of the first register requested<br>(From the PyroMiniBus manual, address 0x000E = Filtered Object Temperature) |
| 0001 | Total number of registers requested.<br>(This register has a length of 1 Word, as shown in the PyroMiniBus manual) |
| 59E7 | CRC (If possible, let the software automatically calculate this) |

The Modbus Slave responds with the requested data. The response is 11 03 02 00E7 CD39

| | |
|---|---|
| 11 | Slave Address<br>(11 hex = address 17) |
| 03 | Function Code 3<br>(Read Register) |
| 02 | The number of data bytes to follow<br>(1 register x 2 bytes each = 2 bytes total = 4 hex characters) |
| 00E7 | The contents of register 000E; the measured temperature<br>(0x00E7 = decimal 231 = 23.1°C) *Note, as stated in the PyroMiniBus manual, the temperature is in tenths of a degree.* |
| CD39 | CRC |

## Example 2 - To Write the Emissivity Setting

In this example, a PyroMini sensor with address 176 is the Modbus Slave.

The emissivity setting for the sensor is to be set to 0.95.

The Modbus Master sends the command B0 06 0014 251C 76C9

| | |
|---|---|
| B0 | Slave Address<br>(0xB0 is the Modbus address of the sensor i.e. 176 in decimal) |
| 06 | Function Code 6<br>(Write Register) |
| 0014 | Data Address of the first register requested<br>(from the Modbus table in the PyroMini instruction manual, address 0x0014 = Emissivity Setting) |
| 251C | Value to write (emissivity setting 0.95)<br>The PyroMini manual states that 1 LSB (Least Significant Bit) = 0.0001. Emissivity setting 0.95 = decimal 9500 = hex 251C |
| 76C9 | CRC |

The Modbus Slave then sends a response to confirm the data has been written – the response is B0 06 02 251C ECA4

| | |
|---|---|
| B0 | Slave Address<br>(0xB0 is address 176 in decimal) |
| 06 | Function Code 6<br>(Write Register) |
| 02 | The number of data bytes to follow<br>(1 register x 2 bytes each = 2 bytes total) |
| 251C | The contents of register 0014; the emissivity value<br>(0x251C = decimal 9500 = emissivity 0.95) *Note: the units of the emissivity value are 0.0001* |
| ECA4 | CRC |

## More information

For more information on how Modbus works, please use the following links:

http://www.simplymodbus.ca

http://www.modbus.org/specs.php