Users Guide

OneSix™ Server

Version 2.2

Point Six, Inc
2333 Alumni Parkway
Suite 305
Lexington, KY 40517
USA

(859) 266-3606

http://www.pointsix.com

Contents

Overview

Quick Start

When OneSix is run for the first time it displays this help screen. You may view this help screen again by selecting "Help" from OneSix's main window.

OneSix is designed to gather data from sensors. There are two ways OneSix gathers sensor data, through a Wireless network or a 1-Wire network. The setup process is a little different for each type of network. OneSix makes available the data it receives from each sensor through a DDE link. The DDE interface is the same whether the network is Wireless or 1-Wire.

The main OneSix™ Server window will show the total number of devices in the startup list and how many of those that are online. The offline devices will be shown in the list box at the bottom of the window.

OneSix™ can be a stand-alone data logger. When data logging is enabled, OneSix™ will store collected data in an ASCII file at a programmable interval. See section "OneSix Server Data Logging".

This Help and all other Help topics may be accessed from the main window by selecting Help or pressing F1.

Wireless Networks

- Plug the receiver into your serial port
- Start OneSix.
- If necessary, tell OneSix what kind of Receiver you are using and what COM port it is attached to.
- When the "Add Wireless Device" screen appears, press the service mode button on your transmitter to transmit a packet that OneSix can see.
- When a device appears in the window, you may click to select it and then press the 'Edit' button to change its label name and transmit time.
- Press OK to add these sensors to the OneSix list and for OneSix to start acquiring data.

1-Wire Networks

OneSix™ Server does the following when run for the first time:

- Searches for a MicroLan™ network that is attached to a free communication port with devices connected.
- If necessary, tell OneSix what kind of Host Adaptor you are using and what COM port it is attached to.
- Searches the network for existing devices and adds them to its device list in the **onesix.ini** file. At this time OneSix™ assigns unique labels (that will be used for the DDE topic names) and configures each device.
- Creates DDE variables for each device and starts polling and gathering data from each device.

If there is not an **onesix.ini** file or if [Server]NumberDevices=0 and when OneSix™ starts, it will automatically search for all devices on the network and add them to its list of devices to poll.

Description

OneSix™ is a data acquisition Dynamic Data Exchange (DDE) server that acquires data from devices and passes this data using DDE to a client application. If the devices are wireless, then OneSix waits for the device to transmit and then processes the transmission. If the devices are on a 1-Wire network, then OneSix gathers the data by polling the network. For wireless networks, OneSix™ gathers data from the Point Sensor family. For 1-Wire™ networks, OneSix gathers data from T8 series product, T1SS, and T2SS for generalized I/O data acquisition and control, DS1820 based temperature sensors, DS2423 counters, DS2450 Analog inputs and other DS2406 based devices such as Pressure and Humidity as well as the 3000 series of sensors.

Any Windows® application that can use DDE can obtain this data from the OneSix™ Server. Examples of such applications include Microsoft® Excel, Wonderware® Intouch, National Instruments® LookOut, Capital Equipment Corp.® TestPoint, and general development applications like Microsoft® Visual Basic and Borland® Delphi.

OneSix™ can be a stand-alone data logger. When data logging is enabled, OneSix™ will also store collected data in an ASCII file at a programmable interval. Another program such as a spreadsheet or a database manager can import the data. See section "OneSix Server Data Logging".

OneSix Server and Wireless

OneSix Server can receive transmissions from wireless devices such as the wireless temperature transmitter. Other devices include the wireless humidity sensor and wireless analog input. OneSix uses a wireless receiver such as the Point Repeater, the Wireless HA9 or the PointView Receiver to receive transmissions from wireless sensors. OneSix can also use multiple TCP/IP Receivers (Point Servers in Pass-Thru mode) to receive data through TCP/IP connection.

On a wired network, OneSix repeatedly polls the devices on the network. On a wireless network, OneSix waits for a transmission from the devices on the network.

When setting up a wireless network, OneSix passively waits for transmissions instead of actively polling for devices. This can make the acquisition of sensors take a longer time than for a wired network as OneSix must wait for the device to transmit. You can speed up this process by pressing the service button on the device so that it transmits while you are in the OneSix Wireless Setup screen.

Each device has an entry in the device's INI file section that tells OneSix what media the device will be using to transmit. This entry is called **"Media"** and it can have one of two values, "1-Wire" or "Wireless." Each device also has a **"repeateraddress"** attribute. When running OneSix with a Point Repeater, the Repeater ID is stored in the "repeateraddress" entry. The value in the "repeateraddress" indicates which Point Repeater sent the sensor information With all other wireless receivers, the "repeateraddress" is set to 0. Finally, the INI entry and DDE item **"Polltime"** for each device has a different meaning when working with wireless transmitters. OneSix cannot poll the wireless sensors. It must wait for a transmission. Therefore, OneSix uses polltime to estimate how frequently a transmission should come from a device. It also uses a multiple of this time to determine if a device is offline. This multiple is configurable by the user.

If you are using a TCP/IP Receiver, please see the section TCP/IP Receivers for more information on how to use this product.

Wireless Setup

If you have OneSix running with a wireless receiver such as an HA9 or a PointView, OneSix will display the Setup screen when it starts for the first time. The setup screen is the method by which OneSix acquires wireless devices. The default on the setup screen is to accept only packets that are sent in service mode from the device (the packet is sent by pressing the service button on the device.) To change this default, uncheck the "Service Mode Only" checkbox. OneSix will then display all the packets it receives. When OneSix receives a packet from a device new to it, it names the device following the naming conventions detailed below, and it displays the device's name and serial number in the window.

You have several options after the device has been displayed. The "Clear New" button will clear all the devices that have just been received. The "Delete" button will delete a single device. Click on a device to select it and click on "Delete" to delete it. To edit a device click on the device to select it and click on "Edit" to edit the device. There is also a "Stop" button. If you click that button, OneSix will stop listing devices in the window. When you click on the "Stop" button it becomes a "Receive" button. If you click on that, OneSix will start receiving packets and listing devices again.

When you are ready, hit "OK", or "Cancel" to exit.

Edit Device

The Edit screen for a new RF device shows the device's serial number, which the user cannot change, the device's current Label, which can be changed by the user, and the Transmit Rate for the device. The Label cannot be the same as the label used by any current device. The Transmit Rate is in seconds. The default value is 60. OneSix uses the Transmit Rate to determine if a device has gone offline.

OneSix Server and 1-Wire Networks

When OneSix is started with a Host Adaptor that uses 1-Wire sensors, it automatically polls the 1-Wire network to see what sensors are there. It adds the sensors it finds to a device list.

Host Adaptors / Receivers

Wireless

- Point View Receiver
- Point Host – select "HA9 Wireless" from the "Select Communication Port" screen
- Point Integrator Logger –select "HA9 Wireless" from the "Select Communication Port" screen
- Point Repeater and Repeater III – select "HA9 Wireless" and set the TZR baud rate to 19,2000.
- TCP/IP Receiver (Point Server in “Pass-Thru” mode)

1-Wire

- HA4
- HA5
- Multiple HA5s using an RS485 network
- DS9097-U
- HA6 using a 900 MHz Wireless network

Device List

OneSix™ Server interfaces to a large number of devices. New devices are continually being created for the 1-Wire™ and Wireless network. Check with your distributor for the latest.

The following is a list of devices supported by OneSix™ Server:

Wireless

Point Sensor Temperature

Point Sensor Temp/Humidity

Point Sensor Analog 5V

Point Sensor Analog 10V

Point Sensor Analog 20mA

Point Probe

Point Sensor Pressure

Point Sensor DSCI

Point Sensor IR Counter

Point Sensor Thermistor

Point Sensor Counter Temperature

Point Sensor Fast Counter Temperature

Point Sensor Alarm Temperature

Point Access/Control Reader

Point Directional Counter

Point Dual Discrete Output

Point Analog Output

1-Wire

DS2406	Dual Addressable Switch Plus 1K-Bit Memory
D2P	DS2406 with excitation on inputs
D2PHOTO	DS2406 with light sensor
T1SS	1 Channel Digital SM I/O Module to 1-Wire™ Expansion Card
T2SS	2 Channel Digital SM I/O Module to 1-Wire™ Expansion Card
T8ADH	8 Channel 12 Bit Analog, Digital I/O, RS232 to 1-Wire™ Card
T8AH	8 Channel 12 Bit Analog, RS232 to 1-Wire™ Card
T8DH	8 Channel Digital I/O, RS232 to 1-Wire™ Card
T8AD	8 Channel 12 Bit Analog, Digital I/O 1-Wire™ Expansion Card
T8A	8 Channel 12 Bit Analog, 1-Wire™ Expansion Card
T8D	8 Channel Digital I/O 1-Wire™ Expansion Card
T8DP	8 Channel Digital I/O 1-Wire™ Expansion Card with excitation
Z2TEN	0-10 Volt 12 bit Analog Output Controller
DS1820/1920	Temperature Sensor
DS18S20/18B20/1822	Temperature Sensor
DS2423	4K-Bit 1-Wire™ RAM with Counters
DS2450	4 channel 16 bit analog input with digital outputs
DS2890	1 Channel Digital Potentiometer
Temperature Sensors	family of packaged DS1820s

Pressure Probe

Force Probe

Humidity Probe

Force Probe

Conductivity Probe

3000 Series Sensors

Weather Vane	(Texas Weather Instruments)
Rain Gauge	(Texas Weather Instruments)
Lightning Counter	(Texas Weather Instruments)

Wireless Weather System (Texas Weather Instruments)

Polling

Wireless

For a wireless device the polltime has a slightly different meaning. OneSix does not actively poll wireless devices. Rather, it waits to receive a packet from the devices. Therefore, the polltime is how often OneSix expects to receive a packet from the wireless devices.

The polltime for a wireless device is configurable during device setup. OneSix will use this time to determine when to mark a device offline. OneSix has a INI file parameter called "Tries" that sets how long a device has to transmit before it is considered offline. For example, if the device has a polltime of 1 minute, and OneSix has "tries" set to 3, then OneSix will consider the device offline if it does not receive a packet from that device in 3 minutes ("tries" times "polltime").

1-Wire

OneSix™ polls each device on the network based on a polltime. This time is user configurable. If a device is determined to be off-line (cannot communicate with it), OneSix™ will poll the device at a different rate. Each off-line device is polled one at a time at this rate until proper communication is established.

Some devices have dwell times to allow a device to do internal processing or to obtain power. For example: DS18S20 requires a dwell time of 850ms. No communication occurs during this time. For specific information about dwell time for a device, see that device's documentation.

This dwell time is automatically handled by the server and has an impact on how often devices on the network are polled.

A polltime of zero stops the poll for that device.

Identification and Associations

OneSix™ identifies each device with a unique label name. A client application uses this label (a DDE topic) to obtain data from the server.

With some products such as the T8 series, OneSix™ treats a collection of devices as individual entities but identifies them with an associated label name. An Association is a collection of devices with the same label name but with unique postfix identifiers.

Filtering

For both analog and temperature devices, OneSix™ allows the user to define a deadband to control the amount of change that must occur before the server updates a Client with new data. Deadband is useful to control the amount of information that is passed through DDE to the client application. For analog devices, the deadband is entered as the percentage of full scale of Engineering units. For temperature devices, the deadband is entered in degrees C.

Also for both analog and temperature devices, One Six™ provides the means to filter the data with either an Average, Median or combination Median Average filter before either logging data or delivering data through DDE. In the One Six INI file, you specify the filter type and the number of samples to filter.

What is DDE?

Dynamic Data Exchange (DDE) is a standard inter-application communication protocol built into the Microsoft® Windows® operating system. It allows Windows® programs that support DDE to exchange data between themselves. By simply specifying an application, topic, and item, a client application can exchange data with a server application. A DDE server is a program that has access to data and can provide that data to other Windows® programs. A DDE client is a program that can obtain data from a server.

To establish a link with a DDE server, there three pieces of information required:

Application: When you use a Microsoft® Windows® application to obtain data from another Windows® application, you must provide the name of the application you wish to respond to your data requests. Application names are sometimes called service names.

Topic: Available Topics are determined by the Application. The application asking for the data must choose an available topic or data exchange cannot take place. Topics are general classifications with multiple data items.

Item: After the Application and Topic, the application must provide the specific Item. The Item determines the data information related to an application topic.

Using DDE with Microsoft® Excel

Digital Input Example:

Application Name is: ONESIX

Topic is: digital1.1

Item is: inputA

Example of what you would enter into an Excel® spreadsheet cell to read the state of the first digital input from the T8ADH via OneSix™ Server.

=ONESIX|DIGITAL1.1!INPUTA

Analog Input Example:

Application Name is: ONESIX

Topic is: analog1

Item is: input1

Example of what you would enter into an Excel® spreadsheet cell to read the value of the first analog input from the T8ADH via OneSix™ Server.

=ONESIX|ANALOG1!INPUT1

The easiest way to make a DDE Link in an Excel® cell is to use the Clipboard. Click on the DDE Variables menu option in the main menu of OneSix DDE Server. Select the information you want, press the Copy Link button and at Excel select the cell you want the information to be displayed and then click on the Paste button. Excel® will now show the linked data from the OneSix™ Server. See the section "View DDE Variables" for more information.

Using DDE with Microsoft® Visual Basic

To create a link in Visual Basic to bring real time data from OneSix DDE Server into a Visual Basic object, do the following: (The following example is used 'onesix|temp1!input'.)

1) Start the OneSix DDE Server and have it polling a device.

2) Place a Label, PictureBox, or TextBox on a Form.

3) Assign the LinkItem Property with the item name (ex: obj.LinkItem = input)

4) Assign the LinkTopic Property with the application and topic names: LinkTopic=application|topic (ex: obj.LinkTopic = onesix|temp1)

5) Assign the LinkMode Property to 1-Automatic (ex: obj.LinkMode = 1)

The DDE link will be established and real time data should be displaying in the display object.

Visual Basic will save the property settings thus the DDE link settings are saved with executable. Therefore when the form is created, the executable (or Visual Basic) will try to reestablish the DDE Link.

Use the method LinkPoke (with Label, PictureBox or TextBox) to poke DDE information to OneSix Server.

Consult the Visual Basic Help for more information.

Data Logging

OneSix™ Server can log data to an ASCII file at a programmable interval. The default name of the file is ONESIX.LOG.

A client application can control logging through DDE variables. The client can start/stop logging or just log a single record.

For diagnostic purposes, OneSix™ can log errors that have occurred on the 1-Wire network while OneSix™ is polling devices.

For more information see the section OneSix Server Data Logging.

Diagnostics

With OneSix™, you can monitor the health of your 1-Wire™ or Wireless network and the devices on it. OneSix™ reports through DDE errors that it encounters while polling devices or waiting for transmissions. These errors can be noted in an event error log. See the section called "Server Item Names" for more information.

OneSix Main Window

Main Window

OneSix™ displays the number of devices online and offline. It also lists the devices that are currently offline displaying the name and serial number of the device. If OneSix™ is running with multiple HA5s, HA6s or TCP/IP receivers, then for each offline device it will display the node address, name and serial number of the device.

Setup

OneSix™ simplifies setup by automatically finding, identifying and beginning to poll devices in a network. OneSix™ assumes default initial values. You can easily change these values by editing the ONESIX.INI file with an ASCII text editor. (The installation program for OneSix™ Server created a short cut to the ONESIX.INI file. Double clicking will automatically start WordPad or Notepad.) See the section "OneSix Server .INI File Format".

Change Port

The Change Port submenu allows you to switch to another wireless receiver or 1-Wire™ port. When the Change Port submenu is selected, OneSix™ displays the Select Communication Port window. Choose one of the following:

AUTO - instructs OneSix™ to search Port Numbers 1 to 4 for the PointView Receiver.

PointView Receiver – receives RF (wireless) packets.

HA9/TZR Wireless –uses the HA9 or TZR receiver to receive sensor packet data. (TZR must be set to 19200 baud.)

TCP/IP Receiver – uses a receiver that uses the TCP/IP protocol to receive sensor packet data. (Point Server in “pass-thru” mode).

When OK is clicked, OneSix™ tries to initialize the port. For the PointView and TCP/IP Receivers, OneSix confirms that the receiver is present. For the HA9/TZR Wireless, assumes the receiver is present. If OneSix™ fails, OneSix™ displays the OneSix Initializing Error Window allowing you to retry, select another port or exit OneSix™. If OneSix™ succeeds, OneSix™ starts listening for sensor packets or polls devices according to the device list in the INI file.

For HA6, OneSix uses a baud rate of 38,400. For HA5, MultiHA5, OneSix uses a default of 115,200 baud and expects checksum mode to be enabled. For HA5s, OneSix expects the first node addresss to be ‘a’. Set the DIP switches for the HA5 to all switches in the ‘on’ state in order to work with the default baud rate of OneSix.

For more information see the HA5 manual and the [Server] section of the OneSix.ini file.

Add/Delete IP Connection

(TCP/IP Receiver only)

OneSix will display the "Add/Delete IP Connection" screen. IP addresses already setup will be displayed. The following is a list of the columns:

Node – Node number for the IP address. OneSix uses this number to identify this IP connection. OneSix uses this number to update the DDE item "nodaladdress".

IP Address – the IP address that is used to address the receiver.

Port – the port number that is used along with the IP address to address the receiver.

Conn – identifies who initiated the connection. Outgoing: OneSix initiated the connection with the receiver. Incoming: the receiver initiated the connection with OneSix.

Password – initial password used to gain access to the receiver.

Description – a general description field associated with the IP connection.

The following is list of buttons in the "Add/Delete IP Connection" screen:

Test – OneSix will try to make a connection to this IP address and report back success or failure.

Add New – OneSix displays the "New IP Address" screen. The following is list of the fields to enter to add a new IP connection:

IP Address – the IP address that is used to address the receiver

Port – the port number that is used along with the IP address to address the receiver.

Description– a general description field associated with the IP connection.

Password – initial password used to gain access to the receiver. If the receiver does not have a password set or does not require a password then leave this field blank.

Connection – identifies whether OneSix will initiate the connection or whether the receiver will initiate the connection. Outgoing: OneSix initiates the connection. Incoming: OneSix waits for the receiver to initiate the connection.

Edit – OneSix displays the "Edit" screen showing the selected IP connection. The "Edit" screen has the same fields as the "New IP Address" screen.

Delete – delete the selected IP connection.

View IP Connection Status

(TCP/IP Receiver only)

OneSix will display the "IP Status" screen. OneSix shows the current state of the all the connections. The following describes the columns:

Node – Node number for the IP address. OneSix uses this number to identify this IP connection. OneSix uses this number to update the DDE item "nodaladdress".

IP Address – the IP address that is used to address the receiver.

Port – the port number that is used along with the IP address to address the receiver.

Conn – identifies who initiated the connection. Outgoing: OneSix initiated the connection with the receiver. Incoming: the receiver initiated the connection with OneSix.

Description – a general description field associated with the IP connection.

Status – the current status of the connection. The following is a list of the possible status messages:

Never Connected – OneSix never attempted or received a connection to the receiver

IP Connected – OneSix is currently connected to the receiver but has not logged in.

Connected – OneSix is currently connected to the receiver is waiting to receive sensor packets.

Disconnected – OneSix was connected but is now disconnected from the receiver. If the connection was initiated by OneSix, OneSix will continually try to reestablish the connection.

TCP/IP Error – a TCP/IP error was encounter and OneSix is not connected to the receiver.

No IP Connection – OneSix tried to establish a connection but failed.

Wrong Password – The receiver requires a login password and the incorrect password was used by OneSix.

Outputs

Outputs are only send packets. The Output devices do not send packets. Therefore the setup for an Output sensor must be done manually. Some Outputs need to be "trained" to the packets that it will receive for from the Server.

New and Edit

Create or edit an Output object to manage an Output sensor. The following describe the parameters necessary to set up an Output sensor.

Type – "Dual Discrete Outputs" or "12 bit Analog Output". Select the type of Output sensor. Note changeable only for the Edit Output window.

Serial Number – Some Outputs sensors can "learn" a serial number. Others have the serial number preassigned and must be entered in the field. These sensors have the serial number labeled. For those sensors that must learn their serial number, OneSix creates a default serial number. Press the "Generate New Serial Number" button to generate a new serial number.

Label – name assigned for this Output sensor. The label name is used for the DDE topic name. This name must be unique. For the Edit Window, the Label field contains a list box. Select the Output to modify from the list. You can also change the label for the current Output.

Node – This field applies only when using the TCP/IP Receiver. The Node List box contains a list of the TCP/IP Receivers to send the output packet to. For the"Send All" selection, OneSix sends the packet to all the TCP/IP Receivers.

Dual Discrete Outputs

Send when output B is written – If checked, OneSix will only send the output packet only when the "outputB" item is written to (DDE poke). Output A is first written to and the output B. If unchecked, OneSix will send the output packet when either the "outputA" or "outputB" items are written to (DDE poke). The state of both outputs is sent in the same packet.

12 bit Analog Output

Scale – the scale to apply to the raw reading (12 bits: 0 to 4095).

Offset – the offset to apply to the reading after the scale is applied. Engeering Value = Scale * raw + Offset.

Units – units label for the analog output.

Train

The Train dialog is used in the processing of training the Output device to output packets (as produced by OneSix). Follow these steps to train an Output device:

1. Select the desired Output from the list of Outputs.
2. Press and hold the button on the Output device (label as "train").
3. Click the button on the Train window. Repeat until the LED on the Output device is on.

The Output device is now ready to receive Output packets from OneSix.

Search and Add Devices

Wireless

OneSix will display the "Add Wireless Device" screen. Devices already in the onesix.ini file are displayed in the top portion of the screen. Follow the setup procedure to add more devices.

The setup screen is the method by which OneSix acquires wireless devices. The default on the setup screen is to accept only packets that are sent in service mode from the device (the packet is sent by pressing the service button on the device.) To change this default, uncheck the "Service Mode Only" checkbox. OneSix will then display all the packets it receives. When OneSix receives a packet from a new device, it names the device following the naming conventions detailed below, and it displays the device's name and serial number in the window.

You have several options after the device has been displayed. The "Clear New" button will clear all the devices that have just been received. The "Delete" button will delete a single device. Click on a device to select it and click on "Delete" to delete it. To edit a device, click on the device to select it and click on "Edit" to edit the device. There is also a "Stop" button. If you click that button, OneSix will stop listing devices in the window. When you click on the "Stop" button, it becomes a "Receive" button. If you click on that, OneSix will start receiving packets and listing devices again.

When you are ready, hit "OK", or "Cancel" to exit.

Edit Device

The Edit screen for a new RF device shows the device's serial number, which the user cannot change, the device's current Label, which can be changed by the user, and the Transmit Rate for the device. The Label cannot be the same as the label used by any current device. The Transmit Rate is in seconds. The default value is 60. OneSix uses the Transmit Rate to determine if a device has gone offline.

1-Wire

When "Search and Add Devices" is selected, OneSix™ searches the network for new devices and adds any devices found to the device list and assigns default names and startup parameters. The new information is then written to the onesix.ini file.

Manually Add Devices

1-Wire

OneSix will present the Manually Add Devices dialog window. Use the feature to manually (instead of One Six automatically adding devices) add devices to the 1-Wire™ network. In order to use this feature you will need the serial number (ROM code) of the device you want to install. Some devices have more than one 1-Wire™ component associated with the device. You will need know these serial numbers as will. To manually add devices using this window is a two step process. First, enter all serial numbers for the device using the "Add to List" button to move the serial number to the serial number list. Second, click the "Update Device List" button. One Six will then manually add this device to its list and gather any configuration information from the device.

The default node address is 1, which is correct for all single host adaptors. If you are running OneSix in MultiHA5 mode you will need to fill in the correct node address for each device you are adding. It is only possible to add devices to one node at a time. Follow the procedure outlined above for adding devices but make sure you have the correct node address entered in the box provided.

It is best to install one device at a time using this feature.

Wireless

OneSix does not support manual add for wireless devices. Use the OneSix Edit Utility to add devices by serial number.

ReConfigure Net

OneSix™ clears device information from the onesix.ini startup file and completes a SearchAdd to build a new device list as if starting for the first time. **Use caution here**; *all changes that you have made manually to the device information contained in the onesix.ini file will be deleted and overwritten*. OneSix™ does not modify the global configuration parameters contained in the sections Server and Logging of the onesix.ini file.

View DDE Variables

Clicking this menu option shows the most commonly used DDE variables that are online. Clicking the **show all** check box will display all available DDE Variables. By selecting a variable and clicking the **copy link** button, the link is copied onto the windows clipboard where it can then easily be pasted into another program such as Excel™ or Word™ to automatically create the DDE link. To paste the link in Excel™ or Word™ after it is on the clipboard, select **paste special** from the **edit** menu; at the next window select **paste as link**.

Errors

OneSix displays the Communication Errors window. This window shows the last error that occurred while polling devices.

Tries Count: Number of tries that have occurred before a device is considered offline.

Major Error Count: Number of errors that (after tries) took the device offline.

Last Error Code: Number error code of the last error

Last Error Device: The name of the device that had the last error.

Last Error Time: The time when the last error occurred.

Last Error Message: A description of the last error.

Click the Reset button clear the errors and reset the counts.

Quit

When you select this menu option, OneSix™ terminates.

Help

You select the the Help file's contents or index window or you can display the OneSix™ About window.

OneSix Server .INI File Format

onesix.ini

The ONESIX.INI file contains the configuration parameters for the server and the devices being polled or received. When OneSix™ finds a new device it adds the device to the INI file and assigns default initial parameters. These parameters can be changed by using a text editor or the OneSixINI Editing Utility discussed below. (The installation program for OneSix™ Server created short cut to the ONESIX.INI file. Double clicking will automatically start WordPad or Notepad.)

The following rules must be kept when editing the INI file.

1) All device label names must be unique.

2) The NumberDevices in the [Server] section must equal the largest device section [Device*n*].

3) The must be no missing device sections [Device*n*] in the list.

Note: OneSix™ looks at the ini file only at startup or after adding a new device. Changes made to the ini file will not be reflected in OneSix™ until OneSix™ is closed and restarted.

The following is a list of the ONESIX.INI parameters.

SERVER

[Server] - Section name.

PortType - Communication medium (AUTO, HA9/TZR Wireless, Point View, etc.).

PortNo - Port number.

ComAddr - Com Address in ASCII HEX when *PortNo=5* (Default is 3F8) (16 bit version only)

BaudRate –options are: 115,200; 38,400; 19,200; or 1200. The default is 115,200 for HA5 and MultiHA5. The default is 38,400 for HA6. The default is 19,200 for PointView Receiver and HA9. For 1-Wire Host Adaptors, the BaudRate is not used. Make sure the value in the ini file matches the value set in the hardware.

NumberDevices - Number of devices to be polled by the server.

OfflinePollTime - How often to poll for offline devices. Milliseconds

Tries - No. of tries during polling before a device is marked offline.

HostAdapterCheck - Enables/disables the detection of Host Adapter. (0 or 1) (16 bit version only)

DefaultTempUnits - specifies the starting units of Temperature devices

DefaultPressUnits - specifies the starting units of Pressure devices

DefaultForceUnits - specifies the starting units of Force and Scale devices

Parasitic – specifies if One Six expects the device to be powered. If you expected most of the devices in your system to be powered then set to 0. If most devices are to be powered by the 1-Wire network then set to 1. You can later change the individuals devices' Parasitic flag.

TempBoostTemp – (DS1820/18S20 only) specifies the temperature conversion time for parasitic power in milliseconds (default is 800 ms).

PoweredTempTime – (DS1820/18S20 only) specifies the temperature conversion time for external power in milliseconds (default is 800 ms).

ConfigureEnable – 0 – disables the Setup Menu option on the main menu. The user then cannot make any changes to the configuration. 1 – (default) – enable the Setup Menu option on the main menu.

SmallINI – suppresses the listing of minor variables in order to make the ini file smaller (useful for large networks since under Windows, an ini file may be a maximum of 64K).

DefaultThermistorCorrID – overrides the default thermistor Correction ID (normally 128). All thermistors will be assigned this Correction ID unless the ID is overridden in the individual device section.

ForwardAll – species if all packets are forwarded or only packets associated with OneSix's device list get forwarded across a TCP/IP connection. A TCP/IP client must initiate the connection to the port specified with the ForwardPort in the [IP Addresses] section. Each packet has the following format: *sensor packet,node number, node description, sensor label*.

Logging

[Logging] - section name

LogRate - 0: no loging; 1-100000 logging interval in seconds. 5 sec or greater OneSix opens and closes the log file at every logging interval. Less than 5 OneSix leaves the file open until OneSix is terminated or Server!LogRate is set to 0 or 5 and greater.

LogFile - full pathname for the log file - default "ONESIX.LOG"

OfflineIndicator - character or string to indicate offline - NULL: last value

LogErrors - 1 - log runtime network errors to ONESIX.ERR. 0 - do not log errors

IP Addesses

[IP Addresses] – section name (for TCP/IP receivers)

IPWaitPort – port number that OneSix will wait for connections initiated by an IP receiver. (default is 1060)

ForwardPort – port number that OneSix will wait for connections to forward received packets. (default is 0 which means disabled) OneSix will forward received packets through this connection. Each packet has the following format: *sensor packet,node number, node description, sensor label*. The INI parameter FowardAll in the [Server] section controls what packets get sent.

DIGITAL I/O

[Device*n*]- Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=DIGITAL)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic).

Log - Enable/disable logging of device.

OutputOnDemand - Output is initiated on demand or during poll.

DescriptionA - General purpose description.

DescriptionB - General purpose description.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

TEMPERATURE Setup

(Point Temperature, DS1820/1920/18S20/18B20/1822)

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=TEMP)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

Parasitic - Indicate if the sensor is parasitically powered. (0 or 1)

Units - See DDE variables list under input.units for description.

Cal1Raw - Two point calibration point 1.

Cal2Raw - Two point calibration point 2.

Cal1Engr - Two point calibration engineering units for point 1.

Cal2Engr - Two point calibration engineering units for point 2.

DeadBand - Dead band filtering value in ºC. (Default 0.01)

FilterElements - running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=3,MEDIAN". Example: FilterElements1=6,AVERAGE

Resolution – (DS18B20/1822 only) – resolution that One Six sets the part and expects data. One Six adjusts the temperature conversion time based on the set resolution. Values 9 to 12. 12 is the default.

Description - General purpose description.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum

ANALOG MULTIPLE CHANNEL

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=MANALOG)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

Enable*m* - Indicates if the *m* channel is polled. (0 or 1)

Units*m* - units label for the analog channel.

Cal1Raw*m* - Two point calibration point 1.

Cal2Raw*m* - Two point calibration point 2.

Cal1Engr*m* - Two point calibration engineering units for point 1.

Cal2Engr*m* - Two point calibration engineering units for point 2.

DeadBand*m* - Dead band filtering value in % of full scale.

FilterElements*m* – running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=1,MEDIAN" – no filtering. Example: FilterElements1=3,AVERAGE

Description*m* - General purpose description.

where *m* is 1 to 8 channels.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID*m* – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum.

PRESSURE

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress –(only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=PRESS)

PollTime - How often to poll the device.(ms)

Label - Name used for the device (and Topic).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

PollSamples - No. of samples per poll (must be odd no.) If >0 Median Filter if <0 then Average Filter is used.

Temperature - Indicates if sensor needs temperature compensation. (0 or 1).

DwellTime - Time allowed to charge the probe. (ms)

PostDwell - Time after reading to allow bus to recover. (ms)

OverSampleDwell - Time before each over sample to charge probe. (ms)

DeadBand - Dead band filtering value in % of full scale.

Parasitic - Sensor parasitically powered flag. (0 or 1)

Description - General purpose description.

Cal1Raw - Two point calibration point 1.

Cal2Raw - Two point calibration point 2.

Cal1Engr - Two point calibration engineering units for point 1.

Cal2Engr - Two point calibration engineering units for point 2.

Units - See DDE tags list under input.units for desc.

TempCoeff - PPM temperature coefficient of the sensor.

TempCalib - Temperature at which the calibration was done.

TempAddress - 1-Wire(TM) address of the temperature sensor in the probe.

MacroFilterElements - running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=3,MEDIAN". Example: FilterElements1=6,AVERAGE

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum.

PROBE3 (3000 Series sensors)

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=PROBE3)

PollTime - How often to poll the device.(ms)

Label - Name used for the device (and Topic).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

Temperature - Indicates if sensor needs temperature compensation. (0 or 1).

DeadBand - Dead band filtering value in % of full scale.

Parasitic - Sensor parasitically powered flag. (0 or 1)

Description - General purpose description.

Cal1Raw - Two point calibration point 1.

Cal2Raw - Two point calibration point 2.

Cal1Engr - Two point calibration engineering units for point 1.

Cal2Engr - Two point calibration engineering units for point 2.

Units – depends on the sensor type.

UnitsType – See below.

TempCoeff - PPM temperature coefficient of the sensor.

TempCalib - Temperature at which the calibration was done.

MacroFilterElements - running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=3,MEDIAN". Example: FilterElements1=6,AVERAGE

---if the probe has a sensor for temperature compensation the following fields will be in the ini file:

TempUnits – units in which temperature will be reported.

TempDeadband – Dead band filtering in degrees Celsius

TempFilterElements – running macro filter of the sampled temperature data.

TempLogDecimalPlaces – Number of decimal places (precision) used in logging temperature data.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum.

NOTE: UnitsType should be one of the following numbers :

0 (User-defined) , 1 (temperature), 2 (Solar), 3 (Humidity), 4 (pressure), 5 (Force), 6 (Flow), 7 (Length), 8 (soil)

See the DDE variables sections for units. In the case of Solar, the units are 'watts/m2'. Flow is 'GPM' (gallons/minute). Length is any of the following: 'inches', 'feet', 'yards', 'miles', 'mm', 'cm', 'meters', or 'kilometers'.

CONDUCTIVITY

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=CONDUCT)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

PollSamples - No. of samples per poll (must be odd no.) If >0 Median Filter if <0 then Average Filter is used.

Temperature - Indicates if sensor needs temperature compensation. (0 or 1).

DwellTime - Time allowed to charge the probe. (ms)

PostDwell - Time after reading to allow bus to recover. (ms)

OverSampleDwell - Time before each over sample to charge probe. (ms)

DeadBand - Dead band filtering value in % of full scale.

Parasitic - Sensor parasitically powered flag. (0 or 1)

Description - General purpose description.

Cal1Raw - Two point calibration point 1.

Cal2Raw - Two point calibration point 2.

Cal1Engr - Two point calibration engineering units for point 1.

Cal2Engr - Two point calibration engineering units for point 2.

Units –

TempCoeff - PPM temperature coefficient of the sensor.

TempCalib - Temperature at which the calibration was done.

TempAddress - 1-Wire(TM) address of the temperature sensor in the probe.

MacroFilterElements - running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=3,MEDIAN". Example: FilterElements1=6,AVERAGE

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum.

PH

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=PH)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

PollSamples - No. of samples per poll (must be odd no.) If >0 Median Filter if <0 then Average Filter is used.

Temperature - Indicates if sensor needs temperature compensation. (0 or 1).

DwellTime - Time allowed to charge the probe. (ms)

PostDwell - Time after reading to allow bus to recover. (ms)

OverSampleDwell - Time before each over sample to charge probe. (ms)

DeadBand - Dead band filtering value in % of full scale.

Parasitic - Sensor parasitically powered flag. (0 or 1)

Description - General purpose description.

Cal1Raw - Two point calibration point 1.

Cal2Raw - Two point calibration point 2.

Cal1Engr - Two point calibration engineering units for point 1.

Cal2Engr - Two point calibration engineering units for point 2.

Units - "pH".

TempCoeff - PPM temperature coefficient of the sensor.

TempCalib - Temperature at which the calibration was done.

TempAddress - 1-Wire(TM) address of the temperature sensor in the probe.

MacroFilterElements - running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=3,MEDIAN". Example: FilterElements1=6,AVERAGE

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum.

FORCE

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=FORCE)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

PollSamples - No. of samples per poll (must be odd no.) If >0 Median Filter if <0 then Average Filter is used.

Temperature - Indicates if sensor needs temperature compensation. (0 or 1).

DwellTime - Time allowed to charge the probe. (ms)

PostDwell - Time after reading to allow bus to recover. (ms)

OverSampleDwell - Time before each over sample to charge probe. (ms)

DeadBand - Dead band filtering value in % of full scale.

Parasitic - Sensor parasitically powered flag. (0 or 1)

Description - General purpose description.

Cal1Raw - Two point calibration point 1.

Cal2Raw - Two point calibration point 2.

Cal1Engr - Two point calibration engineering units for point 1.

Cal2Engr - Two point calibration engineering units for point 2.

Units - See DDE tags list under input.units for desc.

TempCoeff - PPM temperature coefficient of the sensor.

TempCalib - Temperature at which the calibration was done.

TempAddress - 1-Wire(TM) address of the temperature sensor in the probe.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum.

HUMIDITY

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=HUMIDITY)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

PollSamples - No. of samples per poll (must be odd no.) If >0 Median Filter if <0 then Average Filter is used.

Temperature - Indicates if sensor needs temperature compensation. (0 or 1).

DwellTime - Time allowed to charge the probe. (ms)

PostDwell - Time after reading to allow bus to recover. (ms)

OverSampleDwell - Time before each over sample to charge probe. (ms)

DeadBand - Dead band filtering value in % of full scale.

Parasitic - Sensor parasitically powered flag. (0 or 1)

Description - General purpose description.

Cal1Raw - Two point calibration point 1.

Cal2Raw - Two point calibration point 2.

Cal1Engr - Two point calibration engineering units for point 1.

Cal2Engr - Two point calibration engineering units for point 2.

Units - % Relative Humidity

TempCoeff - PPM temperature coefficient of the sensor.

TempCalib - Temperature at which the calibration was done.

TempAddress - 1-Wire(TM) address of the temperature sensor in the probe.

MacroFilterElements - running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – “AVERAGE”, “MEDIAN” and “AVERAGEMEDIAN”. Default is “=3,MEDIAN”. Example: FilterElements1=6,AVERAGE

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum.

7B Sensor

[Devicen] - Section name

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device Type (DeviceType=7B)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic)

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

PollSamples - No. of samples per poll (must be odd no.) If >0 Median Filter if <0 then Average Filter is used.

Temperature - Indicates if sensor needs temperature compensation. (0 or 1).

DwellTime - Time allowed to charge the probe. (ms)

PostDwell - Time after reading to allow bus to recover. (ms)

OverSampleDwell - Time before each over sample to charge probe. (ms)

DeadBand - Dead band filtering value in % of full scale.

Parasitic - Sensor parasitically powered flag. (0 or 1)

Description - General purpose description.

Cal1Raw - Two point calibration point 1.

Cal2Raw - Two point calibration point 2.

Cal1Engr - Two point calibration engineering units for point 1.

Cal2Engr - Two point calibration engineering units for point 2.

Units -

TempCoeff - PPM temperature coefficient of the sensor.

TempCalib - Temperature at which the calibration was done.

TempAddress - 1-Wire(TM) address of the temperature sensor in the probe.

MacroFilterElements - running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=3,MEDIAN". Example: FilterElements1=6,AVERAGE

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum.

Weather WindVane

[Devicen] - Section name

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - DeviceType = WINDVANE

PollTime - How often to poll the device (ms)

Label - Name used for the device (and Topic) (typically WINDVANE1)

Log - enable/disable logging of device.

LogDecimalPlaces - number of decimals to create when logging.

DirectionOffset - offset in degrees to correct direction indicator.

DirectionAddr1 - North

DirectionAddr2 - NorthEast

DirectionAddr3 - East

DirectionAddr4 - SouthEast

DirectionAddr5 - South

DirectionAddr6 - SouthWest

DirectionAddr7 - West

DirectionAddr8 - NorthWest

WindSpeedAddr - address of the Wind Speed Device (DS2423).

TempAddr - address of Temperature Device - all zeros if no temp.

SpeedUnits - units of the wind speed (see OPC "speed.units").

SpeedAveSamples - number of samples used in the running average filter for speed. (Default is 4).

ReverseDirection – reverses the perceived wind direction. For example, if the raw reading is 35 degrees, the output will be 360-35 or 325 degrees.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

Wireless Weather System

[Devicen] - Section name

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - DeviceType = WIRELESSWW

PollTime - How often to poll the device (ms)

Label - Name used for the device (and Topic) (typically WWeather1)

Log - enable/disable logging of device.

RadioAddress – address of radio transmitter to receive (1 to 7) (7 is default).

WVDirectionOffset – offset in degrees to correct direction indicator.

WVSpeedUnits – units of the wind speed (see OPC "speed.units").

WVSpeedAveSamples – number of samples used in the running average filter for speed. (Default is 4).

HumidMacroFilterElements – running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=3,MEDIAN". Example: FilterElements1=6,AVERAGE

HumidCal1Raw – Two point calibration point 1.

HumidCal1Engr – Two point calibration point 2.

HumidCal2Raw – Two point calibration engineering units for point 1.

HumidCal2Engr – Two point calibration engineering units for point 2.

HumidDeadBand – Dead band filtering value in % of full scale.

HumidLogDecimalPlaces – number of decimals to create when logging.

TempCal1Raw – Two point calibration point 1.

TempCal1Engr – Two point calibration point 2.

TempCal2Raw – Two point calibration engineering units for point 1.

TempCal2Engr – Two point calibration engineering units for point 2.

TempUnits – See DDE variables list under TEMP1! input.units for description.

TempDeadBand – Dead band filtering value in ºC. (Default 0.01)

TempFilterElements – - No. of samples used in the median filter (Default 3).

TempLogDecimalPlaces – number of decimals to create when logging.

RainScaleA – Multiplier for Counter (default: 0.01).

RainUnitsA – Generic label for units - no function (default: inches).

RainLogDecimalPlaces – number of decimals to create when logging.

LightningScaleA – Multiplier for Counter (default: 1.0).

LightningUnitsA – Generic label for units - no function (default: strikes).

LightningLogDecimalPlaces – number of decimals to create when logging

SolarMacroFilterElements – running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=3,MEDIAN". Example: FilterElements1=6,AVERAGE

SolarUnits – Engineering units label (no function) (default: "Watts/m2")

SolarCal1Raw – Two point calibration point 1.

SolarCal1Engr – Two point calibration point 2.

SolarCal2Raw – Two point calibration engineering units for point 1.

SolarCal2Engr – Two point calibration engineering units for point 2.

SolarDeadBand – Dead band filtering value in % of full scale.

SolarLogDecimalPlaces - number of decimals to create when logging.

Description – General purpose description field.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

COUNTER

[Devicen] - Section name

Address - serial number of the sensor.

NodalAddress –(only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - DeviceType=COUNTER

PollTime - How often to poll the device (ms).

Label - Name used for the device (and Topic) (typically COUNTER1)

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

ScaleA - Multiplier for Counter A

ScaleB - Multiplier for Counter B

UnitsA - Generic label for units - no function

UnitsB - Generic label for units - no function

DescriptionA - Generic description field

DescriptionB - Generic description field

EnableCounterB - Enable the gather of Counter B

Battery - Indicates if the counter has battery back up - no function

IDLabel - ID name that was programmed in the OTP tag for the device.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

Analog Output

[Devicen] - Section name

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - DeviceType=ANALOGOUT

PollTime - How often to poll the device (ms).

Label - Name used for the device (and Topic) (typically ANALOGOUT1)

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

Cal1Raw - Two point calibration point 1 (default 0).

Cal2Raw - Two point calibration point 2 (default 4095).

Cal1Engr - Two point calibration engineering units for point 1 (default 0).

Cal2Engr - Two point calibration engineering units for point 2 (default 10).

Units - units label (default "Volts").

OutputOnDemand - 1 - send output immediately; 0 - send output on next poll

Description - Generic description field

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

DS2450 (AIDO)

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=AIDO)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic) (typically AIDO1).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

OutputOnDemand - 1 - send output immediately; 0 - send output on next poll

Precision - number of precision for the analog conversion (1-16) (default 9).

FullVoltage - 1 - 0 to 5.12 v input range; 0 - 0 to 2.56 v input range.

Units*m* - Units label (ex. "Volts").

Cal1Raw*m* - Two point calibration point 1 (ex. 0).

Cal2Raw*m* - Two point calibration point 2 (ex. 65535).

Cal1Engr*m* - Two point calibration engineering units for point 1 (ex. 0).

Cal2Engr*m* - Two point calibration engineering units for point 2 (ex. 5).

DeadBand*m* - Dead band filtering value in % of full scale (Cal2Engr).

FilterElements*m* – running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=1,MEDIAN" – no filtering. Example: FilterElements1=3,AVERAGE

Description*m* - General purpose description.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID*m* – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum.

where *m* is channels 1 to 4 corresponding to channels A to D on the DS2450.

POT (DS2890)

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress –(only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=POT)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic) (typically POT11).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

OutputOnDemand - 1 - send output immediately; 0 - send output on next poll

ChargePump – 1 - turn on the charge pump of the DS2890 (part must be powered); 0 – turn off the charge pump (part can be parasitically powered)

Description – General purpose description field

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

DIGCOUNTER and Directional Counter

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress –(only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=DIGCOUNTER or DIRECTCNT)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic) (typically DIGCOUNTER1 or DIRECTCNT1).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

ScaleA - Multiplier for Counter A

ScaleB - Multiplier for Counter B

UnitsA - Generic label for units - no function

UnitsB - Generic label for units - no function

DescriptionA - Generic description field

DescriptionB - Generic description field

EnableCounterB - Enable the gather of Counter B

Description – General purpose description field

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CNTTEMP

[Device*n*] - Section name.

Address – serial number of the sensor.

NodalAddress –(only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=CNTTEMP)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic) (typically CNTTEMP1).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

Scale - Multiplier for Counter A

Units - Generic label for units - no function

Description - Generic description field

temp.Units - See DDE variables list under temp.input.units for description.

temp.Cal1Raw - Two point calibration point 1.

temp.Cal2Raw - Two point calibration point 2.

temp.Cal1Engr - Two point calibration engineering units for point 1.

temp.Cal2Engr - Two point calibration engineering units for point 2.

temp.DeadBand - Dead band filtering value in ºC. (Default 0.00)

temp.FilterElements - running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=0". Example: FilterElements1=6,AVERAGE

temp.Description - General purpose description.

temp.CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

FASTCNTTEMP

[Device*n*] - Section name.

Address – serial number of the sensor.

NodalAddress –(only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=FASTCNTTEMP)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic) (typically FASTCNTTEMP1).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

Scale - Multiplier for Counter A

Units - Generic label for units - no function

Description - Generic description field

rate.Scale – multiplier for the raw rate.

rate.Units – Generic label for the rate units – no function

temp.Units - See DDE variables list under temp.input.units for description.

temp.Cal1Raw - Two point calibration point 1.

temp.Cal2Raw - Two point calibration point 2.

temp.Cal1Engr - Two point calibration engineering units for point 1.

temp.Cal2Engr - Two point calibration engineering units for point 2.

temp.DeadBand - Dead band filtering value in ºC. (Default 0.00)

temp.FilterElements - running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=0". Example: FilterElements1=6,AVERAGE

temp.Description - General purpose description.

temp.CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

ALARMTEMP

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=TEMP)

PollTime - How often to poll the device. (ms)

Label - Name used for the device (and Topic).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

Parasitic - Indicate if the sensor is parasitically powered. (0 or 1)

Units - See DDE variables list under input.units for description.

Cal1Raw - Two point calibration point 1.

Cal2Raw - Two point calibration point 2.

Cal1Engr - Two point calibration engineering units for point 1.

Cal2Engr - Two point calibration engineering units for point 2.

DeadBand - Dead band filtering value in ºC. (Default 0.01)

FilterElements - running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=3,MEDIAN". Example: FilterElements1=6,AVERAGE

Resolution – (DS18B20/1822 only) – resolution that One Six sets the part and expects data. One Six adjusts the temperature conversion time based on the set resolution. Values 9 to 12. 12 is the default.

Description - General purpose description.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum

Analog Input

[Devicen] - Section name

Address - 1-Wire™ address of the device

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - DeviceType=AI

PollTime - How often to poll the device (ms).

Label - Name used for the device (and Topic) (typically AI1)

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

Cal1Raw - Two point calibration point 1 (default 0).

Cal2Raw - Two point calibration point 2 (default 4095).

Cal1Engr - Two point calibration engineering units for point 1 (default 0).

Cal2Engr - Two point calibration engineering units for point 2 (default 100).

DeadBand - Dead band filtering value in % of full scale (Cal2Engr).

FilterElements – running macro filter of the sampled data. Syntax: [*no. of elements*, *type*] where *no. of elements* is number of samples to filter; *type* – "AVERAGE", "MEDIAN" and "AVERAGEMEDIAN". Default is "=1,MEDIAN" – no filtering. Example: FilterElements=3,AVERAGE

Units - units label (default "%").

Description - Generic description field

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

CorrID – What CorrectionID to use with this device. Must be found in the "Correction.INI" file that you write and keep in the same directory as the OneSix.ini file. 0 is the default and means no correction; 127 is the maximum.

Access/Control Reader

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=IDR)

PollTime - How often to poll the device or wait for packet. (ms)

Label - Name used for the device (and Topic).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

Description - General purpose description.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

Repeater

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – (only used for MultiHA5 or HA6). Possible values are 1..27 and the default is 1.

DeviceType - Device type. (DeviceType=REPEATER)

PollTime - How often to poll the device or wait for packet. (ms)

Label - Name used for the device (and Topic) (typically REPEATER1).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

Description - General purpose description.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

Dual Discrete Output

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – node where the output will be sent. Defaults to 0 for Point View and HA9/TZR Receivers. For TCP/IP Receiver the node number represents each receiver starting at 1. If set to 0, OneSix will send the output packet to all nodes.

DeviceType - Device type. (DeviceType= OUTPUTDISC)

PollTime - How often to poll the device or wait for packet. (ms) Defaults to 0 for outputs.

Label - Name used for the device (and Topic) (typically OUTPUTDISC1).

Log - Enable/disable logging of device.

Description - General purpose description.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

TriggerOnB – 0 – OneSix will send the output packet when either items "outputa" or "outputb" is written to (DDE poke); 1 – OneSix will send the outputpacket when only item "outputb" is written to (DDE poke). Set "outputa" before writing the value to "outputb". Both outputs A and B are set through one packet.

OutTries – number of times to send the output packet. (default 10)

OutInterval – (in milliseconds) – the interval of time between sending output packets. (default 1000).

Wireless Analog Output

[Device*n*] - Section name.

Address - serial number of the sensor.

NodalAddress – node where the output will be sent. Defaults to 0 for Point View and HA9/TZR Receivers. For TCP/IP Receiver the node number represents each receiver starting at 1. If set to 0, OneSix will send the output packet to all nodes.

DeviceType - Device type. (DeviceType= OUTPUTANALOG)

PollTime - How often to poll the device or wait for packet. (ms) Defaults to 0 for outputs.

Label - Name used for the device (and Topic) (typically OUTPUTANALOG1).

Log - Enable/disable logging of device.

LogDecimalPlaces - Number of decimals to create when logging.

Description - General purpose description.

Media- Whether the device is Wireless ("Radio") or wired ("1-Wire")

Cal1Raw - Two point calibration point 1 (default 0).

Cal2Raw - Two point calibration point 2 (default 4095).

Cal1Engr - Two point calibration engineering units for point 1 (default 0).

Cal2Engr - Two point calibration engineering units for point 2 (default 100).

Units - units label (default "%").

Description - Generic description field

OutTries – number of times to send the output packet. (default 10)

OutInterval – (in milliseconds) – the interval of time between sending output packets. (default 1000).

OneSix INI Backup

Any time One Six Server modifies the INI file, One Six will create a copy of the previous INI file. OneSix Server names this backup file the same file name as the INI file but names the file extension as ".B*nn*" where *n* is from 01 to 20. One Six will create up to 20 backups. If One Six needs to create more, it will overwrite the oldest backup which will be typically starting at ".B01".

If you need to revert to a backup just rename the original ONESIX.INI file and then rename the backup to ONESIX.INI.

OneSix Server DDE Variables

Using DDE Links

Application: When you use a Microsoft® Windows® application to obtain data from another Windows® application, you must provide the name of the Application you wish to respond to your data requests. Application names are sometimes called service names.

Topic: Available Topics are determined by the application. The application asking for the data must choose an available topic, or data exchange cannot take place. Topics are general classifications with multiple data items.

Item: After the Application and Topic, the application must provide the specific Item. The Item determines the data information related to an application topic.

Digital Input Example:

Application Name is: **ONESIX**

Topic is: **digital1.1**

Item is: **inputA**

Example of what you would enter into an Excel® spreadsheet cell to read the state of digital input one from a T8 series card via OneSix™ Server.

=ONESIX|DIGITAL1.1!INPUTA

Analog Input Example:

Application Name is: **ONESIX**

Topic is: **analog1**

Item is: **input1**

Example of what you would enter into an Excel® spreadsheet cell to read the value of analog input one from a T8 series card via OneSix™ Server.

=ONESIX|ANALOG1!INPUT1

Topic Names

Topic names define a device. A device contains a nodal address. When "Search and Add Devices" is selected from setup, OneSix™ Server will automatically assign label names that will be used as the Topic name for each new device found. The user can then change the name if they choose.

Default names are:

DIGITAL*n* (Digital I/O)

TEMP*n* (DS1820/1920/18S20/18B20/1822, Point Sensor Temperature, Point Probe, Point Thermistor)

ANALOG*n* (Analog Multi-Channel)

PRESS*n* (Pressure Sensor, Point Sensor Pressure)

TPRESS*n* (Temperature device built-in to a pressure probe)

3000 series: The label name for 3000 series devices is defined in the memory of the physical device. If the name is for example "SOIL", OneSix will create a Topic name called **SOIL***n* where *n* is a number starting at 1 and numbers uniquely each sensor that has the name "SOIL". GPCOUNTERS work the same way.

FORCE*n* (Force probe)

TFORCE*n* (Temperature device built-in to a force probe)

CONDUCT*n* (Conductivity probe)

TCONDUCT*n* (Temperature device built-in to a conductivity probe)

PH*n* (PH probe)

TPH*n* (Temperature device built-in to a pH probe)

HUMIDITY*n* (Humidity probe)

THUMIDITY*n* (Temperature device built-in to a humidity probe, Point Sensor Temp/Humidity)

7B*n*

WI NDVANE*n* (Weather system)

TWINDVANE*n* (Temperature device built-in to a weather vane)

COUNTER*n* (counter)

ANALOGOUT*n* (Analog output)

AIDO*n* (DS2450) (Analog input- Digital output)

POT*n* (DS2890)

Ai*n* (Point Sensor Analog 5V, 10V, 20mA)

DIGCOUNTER*n* (Point Sensor DSCI or Point Sensor IR Counter)

CNTTEMP*n* (Point Counter Temperature)

FASTCNTTEMP*n* (Point Fast Counter Temperature)

ALARMTEMP*n* (Point Alarm Temperature)

IDR*n* (Access/Control Reader)

DIRECTCNT*n* (Directional Counter)

REPEATER*n* (Repeater)

OUTPUTDISC*n* (Dual Discrete Output)

OUTPUTANALOG*n* (Wireless Analog Output)

Where n is a number starting at 1. OneSix™ Server guarantees that each label assigned will be unique (No Duplicates Allowed).

If the device is part of an association, OneSix™ assigns all devices of that association group the same label post fixed with a ".x"; where x is the position of the associated device starting at 1 (Devicen.x).

Example: three digital devices associated together would appear as:

Device1.1 First Device.

Device1.2 Second Device.

Device1.3 Third Device.

Device Item Names

The Server uses Item names to allow access to the I/O data and specific operational information. Item names are dependent on the type of devices used.

Temperature

(Point Temperature, DS1820/1920/18S20/18B20/1822): TEMP*n* or 'T' Prefixed Topic Names

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

input - current temperature in engineering units

input.raw - current temperature in ºC

input.parasitic - indicates if the sensor is in parasitic mode (0 or 1)

** **input.units** - engineering units of the input variable (ºC, ºF, or ºK)

** **input.deadband** - filtering deadband in ºC

** **input.conv** - calibration and unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr)

input.desc - General identification description field.

Note: when **input.units** is written, it changes the temperature scale in the **input** variable. **input.units** must be one of the following: "Celsius", "Fahrenheit", or "Kelvin".

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Digital I/O (DS2406): DIGITAL*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated

inputA - level of PIOA (0 or 1)

inputB - level of PIOB (0 or 1)

counterA - counter of the activity latch of PIOA

counterB - counter of the activity latch of PIOB

counterAB - counter A minus counter B

** **outputA** - (I/O) current value of the output of PIOA (0 or 1)

** **outputB** - (I/O) current value of the output of PIOB (0 or 1)

** **outputA.ondemand** - (I/O) 1 - output on demand; 0 - on next poll (0 or 1)

** **outputB.ondemand** - (I/O) 1 - output on demand; 0 - on next poll (0 or 1)

input.descA - General identification description field for channel A.

input.descB - General identification description field for channel B.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Analog Multiple Channel Sensor (DS2406+): ANALOG*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

input*n* - current value of the analog channel in engineering units

input*n*.raw - current value in binary units (0-4095)

** **input*n*.conv** - engineering unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr)

input*n*.units - units of the engineering value

** **input*n*.deadband** - deadband filtering in % of engineering Units

input*n*.desc - General identification description field.

where *n* is the channel number starting at 1.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Pressure Probe: PRESSn

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address “a”.

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

input - current value of the pressure sensor in engineering units

input.raw - current value in inches of Hg.

input.binary - current value in binary units from the pressure probe.

** **input.conv** - calibration and unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr,TempCoeff,TempCalib)

** **input.units** - units of the engineering value

** **input.deadband** - deadband filtering in % of engineering Units

input.desc - General identification description field.

tempflag - Indicates if a temperature sensor is present in probe.

Note: when **input.units** is written, it changes the pressure scale in the **input** variable. **input.units** must be one of the following "in of H2O", "cm of H2O", "in of Hg", "mm of Hg", "PSI", "Kilopascal", "Millbar", "kgf/cm2", or "Atmosphere".

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Probe3 (3000 series): <Label>n

NOTE: The label name for 3000 series of devices is defined in the memory of the physical device. If the name is for example "SOIL", OneSix will create a Topic name called **SOIL***n* where *n* is a number starting at 1 and numbers uniquely each sensor that has the name "SOIL".

ALSO: if the device includes temperature compensation, OneSix will create a virtual Temperature DDE object with the topic name T<Label>n. For example, THumidity1. See the section on the temperature probe for more information about these DDE links.

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address “a”.

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

input - current value of the pressure sensor in engineering units

input.raw - current value in inches of Hg.

input.binary - current value in binary units from the pressure probe.

** **input.conv** - calibration and unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr,TempCoeff,TempCalib)

** **input.units** - units of the engineering value

** **input.deadband** - deadband filtering in % of engineering Units

input.desc - General identification description field.

tempflag - Indicates if a temperature sensor is present in probe.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Force Probe: FORCE*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address “a”.

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

input - current value of the pressure sensor in engineering units

input.raw - current value in inches of Hg.

input.binary - current value in binary units from the pressure probe.

** **input.conv** - calibration and unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr,TempCoeff,TempCalib)

** **input.units** - units of the engineering value

** **input.deadband** - deadband filtering in % of engineering Units

input.desc - General identification description field.

tempflag - Indicates if a temperature sensor is present in probe.

tare - Set to 1 for zero, -1 to remove tare, returns to 0 when complete.

calibrate - Set the high (full scale) value.

calibratezero - Set the low (zero) value.

Note: when **input.units** is written, it changes the force scale in the **input** variable. **input.units** must be one of the following "ounces", "oz", "grams", "gm".

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Conductivity Probe: CONDUCT*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address “a”.

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

input - current value of the pressure sensor in engineering units

input.raw - current value in inches of Hg.

input.binary - current value in binary units from the pressure probe.

** **input.conv** - calibration and unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr,TempCoeff,TempCalib)

** **input.units** - units of the "10^xIonMoles/Liter".

** **input.deadband** - deadband filtering in % of engineering Units

input.desc - General identification description field.

tempflag - Indicates if a temperature sensor is present in probe.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

pH Probe: PH*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

input - current value of the pressure sensor in engineering units

input.raw - current value in inches of Hg.

input.binary - current value in binary units from the pressure probe.

** **input.conv** - calibration and unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr,TempCoeff,TempCalib)

** **input.units** - units of the PH.

** **input.deadband** - deadband filtering in % of engineering Units

input.desc - General identification description field.

tempflag - Indicates if a temperature sensor is present in probe.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Humidity Probe: HUMIDITY*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address “a”.

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

** **updatecount** – value is incremented whenever the sensor is updated.

serialno - registration number of the temperature sensor

input - current value of the pressure sensor in engineering units

input.raw - current value in inches of Hg.

input.binary - current value in binary units from the pressure probe.

** **input.conv** - calibration and unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr,TempCoeff,TempCalib)

** **input.units** - units of the "%RH".

** **input.deadband** - deadband filtering in % of engineering Units

input.desc - General identification description field.

tempflag - Indicates if a temperature sensor is present in probe.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

7B Sensor: 7B*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds); 0: no polling

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address “a”.

serialno - registration number of the analog device

** **updatecount** – value is incremented whenever the sensor is updated.

input - current value of the analog channel in engineering units

input.raw - current value in inches of Hg

input.binary - current value in binary units read from pressure probe

** **input.conv** - calibration and unit conversion parameters string. (Cal1Raw,Cal1Engr, Cal2Raw,Cal2Engr,TempCoeff,TempCalib)

input.units - units of the engineering value ("volts")

** **input.deadband** - deadband filtering in % of Engineering Units

input.desc - general purpose description field

tempflag - indicates if a temperature sensor is present

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Weather Vane Sensor: WEATHERVANE*n*

online - indicates if the device is communicating

** **polltime** - how often to poll the device (in milliseconds); 0: no polling

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address “a”.

serialno - registration number of the analog device

** **updatecount** – value is incremented whenever the sensor is updated.

speed - current wind speed

speed.raw - current wind speed in "miles/hour"

** **speed.units** - units of wind speed

speed.count - count given by the DS2423.

** **direction** - current wind direction (set direction value to adjust) in degrees.

direction.raw - unadjusted wind speed in degrees.

** **windchill** - current wind chill in units of the temperature device

** **windchill.raw** - current wind chill in Celcius

desc - general purpose description field

Note: speed.units - when written, changes the engineering unit conversion for speed. speed.units must be on of the following:
"miles/hour","MPH","meters/second","M/S", "knots".

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Counter Sensor (DS2423): COUNTER*n*

online - indicates if the device is communicating

** **polltime** - how often to poll the device (in milliseconds); 0: no polling

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the analog device

** **updatecount** – value is incremented whenever the sensor is updated.

benable - gather data from counter 4 in addition to counter 3

battery - indicates if battery backup is being used

{if one channel ...}

input - current count from external counter 3

** **input.diff** - current count from the time the OneSix™ was started or when the variable was reset to 0.

input.time - time in milliseconds of last update (uses Win API GetTickCount())

** **input.scale** - value multiplied against the counts

input.units - units description field

input.desc - general purpose description field

{if two channels ...}

inputA - current count from external counter 3

** **inputA.diff** - current count from the time the OneSix™ was started or when the variable was reset to 0.

inputA.time - time in milliseconds of last update (uses Win API GetTickCount())

** **inputA.scale** - value multiplied against the counts

inputA.units - units description field

inputA.desc - general purpose description field

inputB - current count from external counter 4

** **inputB.diff** - current count from the time the OneSix™ was started or when the variable was reset to 0.

inputB.time - time in milliseconds of last update (uses Win API GetTickCount())

** **inputB.scale** - value multiplied against the counts

inputB.units - units description field

inputB.desc - general purpose description field

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Analog Output: ANALOGOUT*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

** **ondemand** - 1 - send output immediately; 0 - send output on next poll

** **output** - current output value in engineering units

** **output.raw** - current output value in binary units

** **input*n*.conv** - engineering unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr)

output.units - units of the engineering value

output.desc - General identification description field.

where *n* is the channel number starting at 1.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

DS2450 Analog Inputs and Digital Outputs: AIDO*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

** **ondemand** - 1 - send output immediately; 0 - send output on next poll

input*n* - current value of the analog channel in engineering units

input*n*.raw - current value in binary units (0-4095)

** **input*n*.conv** - engineering unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr)

input*n*.units - units of the engineering value

** **input*n*.deadband** - deadband filtering in % of engineering Units

input*n*.desc - General identification description field.

where *n* is channels 1 to 4 corresponding to channels A to D on the DS2450.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

DS2890 Digital Potentiometer: POT*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

serialno - registration number of the DS2890.

** **updatecount** – value is incremented whenever the sensor is updated.

** **ondemand** - 1 - send output immediately; 0 - send output on next poll

** **output.value** - current output value in engineering units (default 0 to 100%)

** **output.raw** - current output value in binary units (0 to 255)

** **input*n*.conv** - engineering unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr)

output.units - units of the engineering value (default is %)

output.desc - General identification description field.

features - Shows the features register of the DS2890

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Wireless Weather System

The Wireless Weather System device does not have any DDE parameters itself but consists of six standard device components that have their own DDE parameters. The components are:

Weather Vane Sensor,

Humidity Sensor,

Temperature configured as THUMIDITY,

Solar Radiation uses PRESS probe DDE object with solar units,

Rain Gauge – uses the Counter device object (with 1 input), and

Lightning Strike monitor – uses the Counter device object (with 1 input).

The first Wireless Weather System device will have the Topic names: WINDVANE1, HUMIDITY1, THUMIDITY1, SOLAR1, RAIN1 and LIGHTNING1.

Also note that Wireless Weather System is treated as 1 device in the "number of devices" and in the INI file but will be treated as six components in the DDE Variables window.

DIGCOUNTER and Directional Counter: DIGCOUNTER*n* and DIRECTCNT*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

dio.inputa – state of the input, 1 or 0

dio.inputb – state of the input, 1 or 0

cnt.inputA - current count of dio.inputa

** **cnt.inputA.diff** - current count from the time the OneSix™ was started or when the variable was reset to 0.

cnt.inputA.time - time in milliseconds of last update (uses Win API GetTickCount())

** **cnt.inputA.scale** - value multiplied against the counts

cnt.inputA.units - units description field

cnt.inputA.desc - general purpose description field

cnt.inputB - current count of dio.inputb

** **cnt.inputB.diff** - current count from the time the OneSix™ was started or when the variable was reset to 0.

cnt.inputB.time - time in milliseconds of last update (uses Win API GetTickCount())

** **cnt.inputB.scale** - value multiplied against the counts

cnt.inputB.units - units description field

cnt.inputB.desc - general purpose description field

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

CNTTEMP: CNTTEMP*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

dio.inputa – state of the input, 1 or 0

dio.inputb – state of the input, 1 or 0

cnt.input - current count of dio.inputa

** **cnt.input.diff** - current count from the time the OneSix™ was started or when the variable was reset to 0.

cnt.input.time - time in milliseconds of last update (uses Win API GetTickCount())

** **cnt.input.scale** - value multiplied against the counts

cnt.input.units - units description field

cnt.input.desc - general purpose description field

temp.input - current temperature in engineering units

temp.input.raw - current temperature in ºC

temp.input.parasitic - indicates if the sensor is in parasitic mode (0 or 1)

** **temp.input.units** - engineering units of the input variable (ºC, ºF, or ºK)

** **temp.input.deadband** - filtering deadband in ºC

** **temp.input.conv** - calibration and unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr)

temp.input.desc - General identification description field.

Note: when **input.units** is written, it changes the temperature scale in the **input** variable. **input.units** must be one of the following: "Celsius", "Fahrenheit", or "Kelvin".

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

FASTCNTTEMP: FASTCNTTEMP*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

cnt.inputA - current count from the sensor

** **cnt.inputA.diff** - current count from the time the OneSix™ was started or when the variable was reset to 0.

cnt.inputA.time - time in milliseconds of last update (uses Win API GetTickCount())

** **cnt.inputA.scale** - value multiplied against the counts

cnt.inputA.units - units description field

cnt.inputA.desc - general purpose description field

cnt.inputB – 8 bit time in seconds of when the sensor captured the last count

cnt.inputB.units - units description field

cnt.inputB.desc - general purpose description field

rate.value – current frequency of the counter scaled

rate.raw – current frequency of the counter in counts per second

** **rate.scale** - value multiplied to the raw rate

rate.units – units label for the rate value (just a description label)

temp.input - current temperature in engineering units

temp.input.raw - current temperature in ºC

temp.input.parasitic - indicates if the sensor is in parasitic mode (0 or 1)

** **temp.input.units** - engineering units of the input variable (ºC, ºF, or ºK)

** **temp.input.deadband** - filtering deadband in ºC

** **temp.input.conv** - calibration and unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr)

temp.input.desc - General identification description field.

Note: when **input.units** is written, it changes the temperature scale in the **input** variable. **input.units** must be one of the following: "Celsius", "Fahrenheit", or "Kelvin".

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

ALARMTEMP: ALARMTEMP*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the temperature sensor

** **updatecount** – value is incremented whenever the sensor is updated.

input - current temperature in engineering units

input.raw - current temperature in ºC

input.parasitic - indicates if the sensor is in parasitic mode (0 or 1)

** **input.units** - engineering units of the input variable (ºC, ºF, or ºK)

** **input.deadband** - filtering deadband in ºC

** **input.conv** - calibration and unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr)

input.desc - General identification description field.

Isalarm – is the temperature above the set point (0 or 1).

Isalarmtime – has the temperature been above the set point for more than the alarm time (0 or 1).

Marker – general purpose 8 bit identifier from the sensor

Alarmtemp – set point for the high temperature alarm in engineering units

Alarmtime – set point for the amount of time that must pass for the input to be above the alarmtemp to trigger the alarmtime alarm

Note: when **input.units** is written, it changes the temperature scale in the **input** variable. **input.units** must be one of the following: "Celsius", "Fahrenheit", or "Kelvin".

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

AnalogInput: AI*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the sensor

** **updatecount** – value is incremented whenever the sensor is updated.

input - current value of the sensor as percent of full scale

** **input.conv** - unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr)

** **input.units** – default is '%'.

input.desc - General identification description field.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Access/Control Reader: IDR*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the access/control reader

** **updatecount** – value is incremented whenever the sensor is updated.

** **access** – 1- access granted; 0 – access denied. When the client pokes a value, the Server takes the contents of the id, serialno, unlocked, striketime and shunttime items and creates a packet that is sent to the reader.

batterylevel – last voltage measurement of the battery.

batterylow – indicates that the battery is low (0 – okay; 1 – low battery)

** **striketime** – the amount of time in seconds to engage the door lock. (0 to 255)

** **shunttime** – the amount of time in seconds that the door can remain open before a "Door Held Open" alarm event will be generated. (0 to 255)

doorheldopen – indicates that the door is being held open (0 – normal door operation; 1 – held open)

dooropen – indicates the current state of the door (0 – door is closed; 1 – door is open)

** **id** – 8 digit identifier of the last read card. A client can poke an id value to send with the output packet when the access item is poked. The id is represented as SSSIIIII where SSS is the 3 digit site code and IIIII is the 5 digit card number.

info – an tag of information from the reader (0 to 255).

idtype – type of ID media. 0 – proximity card, 1 – IR fob.

** **unlocked** – property to lock or unlock the door (0 – locked; 1 – unlocked).

** **tamper** – indicates if the read has detected tampering (0 – reader okay; 1 – tampered)

Note: the Server will send an output packet when the **access** item is poked to respond to a reader request to open the door. The Server uses the contents of the **serialno**, **id**, **access**, **unlocked**, **striketime** and **shunttime** items to generate the output packet. These fields (mainly **id**, **unlocked**, **striketime** and **shunttime**) must be set before the **access** item is poked by the client.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Repeater: REPEATER*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the sensor

** **updatecount** – value is incremented whenever the sensor is updated.

battery - current battery voltage (in volts)

locatorid – location id of the repeater. This is the id that is attached to sensor packets and is represented by the DDE item "repeateraddress". (starts with "a")

cnt418 – count of packets received through the 418 Mhz receiver.

cnt900 – count of packets received through the 900 Mhz transiever.

radionetwork – describes the set 900 Mhz network class of the repeater

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Dual Discrete Output: OUTPUTDISC*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the sensor

** **updatecount** – value is incremented whenever the sensor is updated.

** **outputa** – set to 0 or 1. Packet will be sent if TriggerOnB=0 option is set in the INI file.

** **outputb** – set to 0 or 1. Packet will be sent when written.

outstatus – 1 – OneSix is in the process of sending output packets based on tries and interval; 0 – OneSix has finished sending the output packet.

** **outinterval** – time in milliseconds between OneSix sending the output packet.

** **outtries** – the number of times OneSix will send the output packet. -1 – never stops sending packets.

** **outservice** – When written to, OneSix sends output packets to the output device for training.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Wireless Analog Output: OutputAnalog*n*

online - indicates if the device is communicating (0 or 1)

** **polltime** - how often to poll the device (in milliseconds)

nodaladdress – (only appears for MultiHA5, HA6 or TCP/IP Receiver). The node address starts at 1. For TCP/IP Receivers, the node address corresponds to the node number for the TCP/IP connection. For HA5/HA6, the node address corresponds to the node address of the HA5/HA6 where 1 is address "a".

repeateraddress – (Wireless mode) contains the location identifier of the Point Repeater. If the sensor did not pass through a repeater then the value is 0.

serialno - registration number of the sensor

** **updatecount** – value is incremented whenever the sensor is updated.

** **output** – set variable to the desired Engineering value

** **output.raw** – set the variable to the binary value (0 to 4095).

output.nits - units label assigned to the analog output.

** **output.conv** - unit conversion parameters string. (Cal1Raw,Cal1Engr,Cal2Raw,Cal2Engr)

** **outinterval** – time in milliseconds between OneSix sending the output packet.

** **outtries** – the number of times OneSix will send the output packet. -1 – never stops sending packets.

** **outservice** – When written to, OneSix sends output packets to the output device for training.

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Server Item Names

Server Items are made available when the "Server" Topic is defined. These items give information about the Server and the current operational status of the Server.

PortType - communication medium: "COM", "LPT", etc.

PortNo - port number.

PollTries - No. of missed polls before a device is considered offline.

** **NumberDevices** - number of devices in the system

** **OfflinePolltime** - how often to poll for off-line devices Milliseconds

NoDevicesOffline - how many devices currently offline

Online - indicates if communicating with any devices (0 or 1)

Ready - 0 indicates that OneSix™ is not polling (initializing or do a restart function); 1-OneSix™ is polling the 1-Wire™ network.

LastError - the last error that occurred (string); see below.

LastErrorNo - the last error that occurred (enumerated); see below.

LastErrorDevice - the topic name of the device that had the error

* **ErrorCount** - number of errors that have occurred since Server started

* **ErrTempConv** - number of temperature conversion errors

* **ErrCom** - number of CRC errors

* **ErrNoSensors** - number of errors from Reset/Presence

** **ErrNoHostAdapter** - number of errors from not detecting the Adapter

CntNode – the number of nodes OneSix is maintaining. For HA5/HA6, this is the number of HA5/HA6s in the system. For TCP/IP Receiver, this is the number of connections to receivers. Use this number to access the Node*n* items.

Node*n*.online – indicates if the node is online or offline (0 – offline; 1 – online). ***n*** is the node number starting at 1.

Node*n*.address – address of the node. For HA5/HA6 the addresses start at 1 where a = 1. For TCP/IP Receiver, the address is the IP address. ***n*** is the node number starting at 1.

Node*n*.description– the description string for the node. ***n*** is the node number starting at 1.

CntASSOC - number of associated groups of devices. (For instance, two T8Ds would be 2)

CntDIGITAL - number of digital devices

CntMANALOG - number of multiplexed analog devices

CntTEMP - number of temperature devices

CntPRESS - number of pressure devices

CntFORCE - number of force devices

CntCONDUCT - number of conductivity devices

CntPH - number of pH devices

CntHUMIDITY - number of humidity devices

Cnt7B - number of 7B interface cards attached

CntWINDVANE - number of windvanes

CntRAIN - number of rain gauges

CntCOUNTER - number of counter devices

CntANALOGOUT - number of analog output devices

CntAIDO - number of DS2450 devices

CntPOT – number of DS2890 devices

CntAI – number of Analog Input devices

CntDigCounter – number of DSCIs.

CntCNTTEMP – number of Counter Temperature

CntFASTCNTTEMP – number of Fast Counter Temperature

CntALARMTEMP – number of Alarm Temperature

CntIDR – number of Access/Control Readers

CntDIRECTCNT – number of Directional Counters

CntREPEATER – number of Repeaters

CntOUTPUTDESC – number of Dual Discrete Outputs

CntOUTPUTANALOG – number of Wireless Analog Outputs

Cnt????????? - where ???????? is obtained from the device's EPROM memory and is used in creating the Label name. This applies to the 3000 series of sensors. A 3000 series sensor of type SOIL would be tallied under the variable CntSOIL. This variable only exists when types of this device exist.

** **LogRate -** indicates the rate of logging in seconds. 0 - no logging. Less than 1 is accepted. Less than 5, OneSix™ leaves the file open until terminated or set to 0. When 5 or greater, OneSix™ opens, then logs, and then closes the file at every logging interval. When -1, OneSix™ logs once immediately; OneSix™ sets to 0 when completed.

LogFile - full pathname of the log file.

Desc - extra server variable clients can read/write for any purpose.

Ready - 0 OneSix™ is initializing; 1 OneSix™ is ready and polling.

Version - version number of OneSix™.

DDE Pokes from the client:

SearchAdd - Client must write any number to start SearchAdd function. When completed OneSix™ Server will set back to zero.

Quit - When Client sets value greater than 0, OneSix™ Server terminates and unloads.

Restart - Client must set as follows.

1 to 999 OneSix™ Server deletes the **onesix.ini** file device information, removes all device objects from memory, performs a SearchAdd function, creates a new device list, and rewrites this information to the **onesix.ini** file.

1000+ OneSix™ Server deletes the **onesix.ini** file device information and then restarts the program.

<0 OneSix™ Server restarts the program.

After successfully completing these operations, the **restart** variable is set back to zero.

*** All error counters have the following format 0.000 where 0.001 is added to the count for any device error that causes a retry and a 1.000 is added to the counter if the error causes the device to go offline.**

**** Indicates a write-able DDE link** - indicates that if the client changes this variable using a DDE Poke either the I/O point at the device will be changed, or an operation will be performed. All the variables can be changed by the client but, the Server ignores the change.

Description of LastErrorNo and LastError

-44	Communication Error: CRC16 or Time Out
-46	Cannot Find Receiver.
-45	Error In Temperature Conversion.
-48	No Devices Attached.
-34	Two Devices of Same Association Have Same Position Number.
-39	OneSix™ Server Internal Error.
-33	Invalid calibration entered.
-25	Communication error with Host Adaptor.

OneSix Server Data Logging

Features

OneSix™ Server can log data to an ASCII file at a programmable interval. The default name of the file is ONESIX.LOG. The name can be changed by changing the INI file. The log rate (programmable interval) can be changed in the INI file (*[Logging] LogRate=*) or by a client application through DDE (SERVER!LogRate). By default, all devices listed in the INI file are logged. You can disable logging for each device (*[Devicen] Log=*). The number of decimal places logged for each device can be changed as well (*[Devicen] LogDecimalPlaces=*). When a device is offline, OneSix™ uses the last value gathered when online for logging. If the device has never been online, then -999999.0 is logged. You can have OneSix™ place a character such as '*' or a string such as 'offline' in the logging string instead of the last value(*[Logging] OfflineIndicator=*). All these parameters can be changed in the INI file.

A client application can control logging through DDE variables. The client can start/stop logging or just log a single record. You can have your client application synchronize logging by keying off of one of the device's DDE variable changing values and then have OneSix™ log one record. The client can also control the log rate. The full path name of the log file is also available to a client application through DDE. See the section "DDE Server Items" for more information.

For diagnostic purposes, OneSix™ can log errors that have occurred on the 1-Wire network while OneSix™ is polling devices. This feature is enabled in the INI file (*[Logging] LogErrors=*). OneSix™ places the errors in a file called ONESIX.ERR. See the section "ONESIX.INI: Logging" for more information.

File Management

If the log rate is greater than or equal to 5 seconds, OneSix™ opens and then closes the log file each time data is logged. If the log rate is less than 5 seconds, OneSix™ leaves the file open and closes the file when OneSix™ is terminated, the log rate is set to zero, or the log rate is set to 5 or greater.

The log file can be viewed while OneSix™ is logging. The other program must not write to the file, lock the file, set the file as read only, or set the file's sharing parameters so that OneSix™ cannot log.

When processing data with another program, the best approach is to either rename the file if the log rate is 5 or greater (OneSix™ will automatically create a new file on the next log) or make a copy of the log file. When renaming, you may get an access error. Just try again.

The Error log file is opened and then closed whenever OneSix™ writes an error record.

File Format - Data Log

The log record format is as follows:

hh:mm:ss,mm/dd/yyyy,data1,data2,data3, ... , data**n**

Where:

hh:mm:ss is the time (ex. 23:40:50)

mm/dd/yyyy is the date (ex. 02/03/99)

*data**n*** - is the data for each device that has logging enabled.

Most devices have one value for the data. However 4 devices have different formats.

MANALOG: value1,value2,...,valuen, where n is the number of analog channels enabled.

DIGITAL: inputa, inputb, outputa, outpub, countera, counterb,

COUNTER: countera, counterb - if both channels are enabled

AIDO: ai1,ai2,ai3,ai4,do1,do2,do3,do4

PROBE3: value, temp (only for Probe3 that has temperature compensation)

DIGCOUNTER: dio.inputa, dio.inputb,cnt.inputa,cnt.inputb

CNTTEMP: dio.inputa, dio.inputb,cnt.input, temp.input

FASTCNTTEMP: cnt.inputa,cnt.inputb,temp.input

ALARMTEMP: value,isalarm,isalarmtime,alarmtemp,alarmtime

IDR: cardid,access,islocked,isbattery,doorheldopen,tamper, irtype, batterylevel,into,strike,shunt,dooropen

DIRECTCNT: dio.inputa, dio.inputb,cnt.inputa,cnt.inputb

REPEATER: batteryvoltage, locatorid, 418cnt, 900cnt, netclass

OUTPUTDISC: outputa,outputb

OUTPUTANALOG: output

Example log of a Pressure probe with temperature compensation and a Temperature probe.

10:33:45,03/30/1998,21.28,21.15,29.71

10:34:05,03/30/1998,21.29,21.15,29.71

10:34:25,03/30/1998,21.29,21.15,29.71

10:34:45,03/30/1998,21.33,21.15,29.71

File Format - Error Log

The log record format is as follows:

hh:mm:ss,mm/dd/yyyy,Error number,Error Msg,Online Flag,DeviceLabel

Where:

hh:mm:ss is the time (ex. 23:40:50)

mm/dd/yyyy is the date (ex. 02/03/99)

Error number-- Error number as documented for DDE variable SERVER!ErrorNo

Error Msg - Error message string

Online Flag - Is device considered Online?

DeviceLabel - Name of the device

Example error log:

10:26:30,04/16/1998,-48,One Wire Bus Error or No Devices Attached!,1,TEMP8

10:26:30,04/16/1998,-48,One Wire Bus Error or No Devices Attached!,1,TEMP9

10:26:41,04/16/1998,-45,Error in Temperature Conversion,0,TEMP4

10:26:43,04/16/1998,-45,Error in Temperature Conversion,0,TEMP14

10:26:47,04/16/1998,-45,Error in Temperature Conversion,0,TEMP14

10:26:48,04/16/1998,-45,Error in Temperature Conversion,0,TEMP15

TCP/IP Receivers

Overview of TCP/IP Receivers

OneSix can connect to TCP/IP receivers and receive and process sensors packets. As of this writing, the Point Server (operating in "Pass-Thru" mode) is the only supported TCP/IP receiver. OneSix makes a TCP/IP connection to a receive and then waits for sensor packets. OneSix then processes the sensor packet as if it came from a receiver (like PointView). OneSix can initiate the connection or it can receive connections from a receiver. OneSix can maintain multiple connections simultaneously (only limited by the resources of the host computer).

Setting up to use TCP/IP Receivers

The TCP/IP Receiver must be assigned a valid IP address and IP port number. If the TCP/IP Receiver is going to initiate the connection to OneSix then the TCP/IP Receiver will need to be set up with the IP address and port number of the computer that is running OneSix. See the manual for the specific receiver you want to use for information about setting the IP addresses.

To prepare OneSix to use a TCP/IP Receiver, make sure the Port type is "TCP/IP Receiver". If you are using OneSix for the first time, select "TCP/IP Receiver" from the "Select Communication Port" option when it shows a window displaying "Cannot find Host Adapter". OneSix will then display the "All Current IP Addresses" window. You can select this window from the Setup menu. Click the "Add New" button to add a new IP connection. Click the "Test" button to test the selected IP Address. Click the "Edit" button to change the properties of a connection. When you click the OK button, OneSix will attempt to the list of specified IP addresses. You can view the status of the connections by selection the Setup Menu and then select "View IP Connection Status" to display the "IP Status" window.

Notes

If OneSix initiates the connection then OneSix will continually try to establish the connection when the connection is disrupted or never accepted. If the receiver initiates the connection, OneSix relies on the receiver to re-establish the connection if the connection is disrupted.

Each TCP/IP connection is assigned a "node number". This "node number" gets associated with a sensor's data. The "nodaladdress" DDE item is updated with the "node number".

Information about the individual TCP/IP connection is made available through the Server DDE parameters (the "Node*x*" DDE item). See the Server DDE topic for more information.

OneSix can wait for connections initiated by the receiver. OneSix will automatically add the TCP/IP connection to its list. OneSix waits on port 1060 by default. You can change this port number through the OneSix INI file. In the "IP Addresses" section change the "IPWaitPort=" line.

OneSix and HA6/MultiHA5s

Overview

OneSix can be used with multiple host adaptors. OneSix treats all devices polled or received by the host adaptors as part of one network. This allows you to have a larger network than may otherwise be possible.

Setup

MultiHA5s

OneSix has been reconfigured to work with more than one HA5, therefore expanding the limits of your 1-Wire network. Refer to the HA5 documentation for instructions on configuring your system. You will need to set up an RS485 network to communicate with HA5s. When you are ready start OneSix. Go to 'Setup---Change Port.' From the drop down Port Type list select 'MultiHA5'. Then do a Search-Add or Reconfigure Net to gather all the devices on the net. All the other procedures remain the same as any other OneSix operation.

NodalAddress is a DDE variable for each device. Under 'DDE Variables' click 'Show All' to see the link to each device's node address.

When OneSix is running with MultiHA5s and a device falls offline, the device's node adress will be listed in the 'Offline Devices' box along with the device's label and serial number as before.

The checksum must be enabled for all HA5s. Each HA5 must have consecutive node addresses. OneSix's default baudrate is 115,200. You can set the baud rate using the BaudRate parameter in the [Server] section of the INI file. All HA5s must be set to the same baudrate as OneSix Server.

Use a RS485 adapter that uses transmit time (Send Date) rather than the RTS handshake line to enable the RS485. OneSix does not support use of the RTS handshake line.

HA6s

Use the HA6 Setup Utility to set up the HA6s and TZR. Each HA6 must have consecutive node addresses. The TZR must be programmed to a baudrate of 38400.

Polling

OneSix views the system of HA6/ HA5s with multiple 1-Wire networks as one single 1-Wire network. OneSix will poll the system beginning with the first device on the first node, then the first device on the second node, and so on until it gets to the first device on the last node. The next device polled will be the second device on the first node, then the second device on the second node, and so on. In order to determine the best case sampling time you must add up all the individual acquisition times. For example, suppose you have a system that has 4 HA5s and 200 counters (DS2423) collecting channels A and B. It takes 80 ms to acquire the counts for each counter. It takes 16 seconds (80 ms * 200 counters) for OneSix to poll all the counters.

Device Management

Online/Offline

OneSix will check if an HA5 or HA6 is online. If OneSix determines that an HA5 or HA6 is offline, it will mark all devices attached to that HA5 or HA6 as 'Offline.' When the HA5 or HA6 is brought back online, OneSix will mark all the devices attached to the HA5 or HA6 as back online—unless and until it needs to mark a device individually as offline.

Search and Add Devices

OneSix will search all available HA5s/HA6s for new devices. HA5s/HA6s are expected to be addressed sequentially. OneSix will attempt to connect with three HA5s/HA6s that are not present before stopping the search. If you need to remove up to two nodes OneSix will continue to find the rest of the HA5s/HA6s.

Using OneSix with Multiple Ports

Multiple Ports Overview

OneSix™ can interface to more than one 1-Wire™ simultaneously. A copy or instance of OneSix™ is loaded for each 1-Wire™ port. The client application communicates to the instance that is managing the desired port.

Multiple Port Details

To create an instance of OneSix for each communications port, just create a copy of the OneSix.EXE file for each instance you desire. Name each copy so that you can identify that instance. For example suppose you have two 1-Wire networks using 2 HA4s. You make a copy of the OneSix.EXE file and then rename the file to "OneSix1.EXE" for the instance that will manage the network attached to COM1. Make another copy and rename that to "OneSix2.EXE" for the instance that will manage the network attached to COM2.

When you are first setting up each instance, only have that instance running. Select the desired port and host adapter type for this instance. OneSix will try to automatically find a host adapter. Verify that OneSix has found the right port. If not, then select the correct port. After you have setup each instance, then you can have all the instances running simulataneously.

OneSix creates an INI file with same name as the itself. For instance if OneSix1.EXE is executed, OneSix will create an INI file called OneSix1.INI. OneSix's knowledge about the attached network is kept in that INI file. You can use the OneSix INI Edit Utility to edit this INI file.

To interface a client DDE application, use the name of instance you have created for the DDE Application Name instead of "OneSix".

Error Messages

Communication Errors

OneSix™ Server communication initialization errors will cause the 'OneSix Initializing Error!" window to be displayed to help you determine what the cause of failure may be and to allow you to retry or abort the program.

No Devices Attached!

Reconnect any disconnected devices. Ensure that all devices that require power are powered.

Error In Finding/Initializing Port.

You have specified a port that does not exist on your computer. Use **Setup** to change to a port you have, or select **Auto** and let OneSix™ Server search for you.

Cannot Open Communication Port; Already In Use.

Close application that is using the communication port or provide OneSix™ Server with another RS-232 communications port.

Cannot Find Receiver!

This indicates OneSix™ Server cannot find the receiver. If the receiver is connected with a serial cable, try another serial cable. The receiver uses the RS232 handshake lines for power. Is the power LED on?

Try another port.

Communication Error with Host Adaptor

OneSix cannot communicate with the Host Adaptor. This error will only be reported when OneSix is running in MultiHA5 mode. During communication with an HA5, OneSix encountered a checksum or timeout error.

One Wire Bus Error!

OneSix™ Server had a CRC error reading from the net or the bus could be shorted to cause this error.

Cannot find and load TMEX lowlevel driver!

OneSix™ Server cannot find one of its DLL drivers. Either the driver is missing from the Windows\System folder or the registry entry got erased. Try reinstalling the program. If problem continues, please consult your dealer.

Runtime Errors

Communication Error: CRC16 or Time Out

1-Wire

Bus may be shorted or a device such as a temperature sensor may be set for non-parasitic mode while actually running in parasitic mode.

Wireless

Sensor transmission was not received within the polltime or the transmission got corrupted.

Cannot Find Receiver.

Receiver has become disconnected from the computer.

Error In Temperature Conversion.

The temperature sensor is sending bad data to the server.

Two Devices of Same Association Have Same Position Number.

One of the two must be changed for OneSix™ to function correctly.

OneSix Server Internal Error.

Contact your dealer.

Corrections

Why Use Corrections

Some devices, like a thermistor, need their data corrected in order to be meaningful. A thermistor uses resistance to measure temperature. The relation between the resistance and the temperature is non linear. A thermistor uses a set of correction parameters to convert resistance to temperature. Other devices may need to correct data values in a non linear way. OneSix provides a way for the user to correct the data from a sensor using a set of correction parameters or a lookup table.

Correction IDs

Tell OneSix that you want a device to use a correction by specifying the Correction ID in the OneSix INI file. Within the device section add a line saying "CorrID=" and the number of the correction that you want OneSix to use. The Corrections are found in the Correction.INI file that you write.

Pre-defined Thermistor

The thermistor made by the vendor Thermometrics has the part number RL0503-5820-97-MS . This thermistor's parameters are predefined in OneSix and given the Correction ID 128. (In the device section of the OneSix.INI file, write "CorrID=128" to use this type.)

The parameters for the predefined thermistor are as follows:

NTC_DIVIDERES = 10000

NTC_NORMALRES = 10000

Fitting constants:

A = 3.3539264E-3

B = 2.5609446E-4

C = 1.9621987E-6

D = 4.6045930E-8

How to use Corrections

Correction.INI file

After adding the Correction ID to the device section of the OneSix.INI file, you must set up the Correction used by that Correction ID. If you're using a correction that is not predefined, you must write a Correction.INI file. This file should be stored in the same directory as the OneSix.INI file. Name your file "Correction.INI" and include all the correction sections you like, from 1 to 127. Include a CorrType, either 1 (for correction parameters) or 2 (for a lookup table).

Lookup Table

A lookup table is a CorrType of 2. If you do a lookup table, you must observe a couple of rules. You must specify Elements, which is the number of values in the list. The RawValues list and the CorrectedValues list must be of the same length. The RawValues list must go from smallest to largest (numbers must be in increasing order). When writing your lookup values, do not put a space after the comma. There is a functional limit to the number of elements in the lists; roughly, 500 integer values or 300 real values, for the RawValues and CorrectedValues taken individually.

Here is an example of a lookup correction section:

[Correction5]

CorrType=2

Elements=5

RawValues=400,450,500,550,600

CorrectedValues=1400,1450,1500,1550,1600

Thermistor NTC Parameters

You may choose to correct your sensor's data with a group of parameters. Use CorrType 1. If you decide to use parameters, specify the parameters as in the following example:

[Correction2]

CorrType=1

DivideRes=10000

NormalRes=10000

A=3.3539264E-3

B=2.5609446E-4

C=1.9621987E-6

D=4.6045930E-8

Predefined Corrections

The device type "Thermistor" will default to a correction ID of 128. To override this default you can add a CorrID parameter to the device section of the OneSix.INI file, or you can set a new default for all thermistors in the Server section of the OneSix.INI file. To change the default CorrectionID for all thermistors add the parameter "DefaultThermistorCorrID=" to the Server section, and set it to whatever CorrectionID you wish.

Predefined Corrections use the following equations. The first equation normalizes the passed value, and the second produces the conversion to degrees Celsius.

Parameters:

NTC_DIVIDERES = 10000

NTC_NORMALRES = 10000

Fitting constants:

A = 3.3539264E-3

B = 2.5609446E-4

C = 1.9621987E-6

D = 4.6045930E-8

R_{normal} = NTC_DIVIDERES / (32767 / value) - 1 / NTC_NORMALRES

Value in degrees Celsuis = 1 / (A + B*R_{normal} + C*R_{normal}^2 + D*R_{normal}^3)

Application Notes

Installing New Devices on a Large Network

1-Wire only

Installing many devices on a large network can be difficult especially establishing a correlation between a physical device and the device's label name. One way to work around this problem is to add one device at a time and note the physical location with the device's label and serial number.

With long length networks, you might starting getting errors when doing a SearchAdd function. The SearchAdd operation is sensitive to noise and thus long cable lengths. To work around the problem, just disconnect the network at the host adapter. Temporarily attach the device to be added. Implement the SearchAdd function. Remove the device and install it in its intended location. Note the physical location, the device label, and serial number. As an added note, make a backup copy of the ONESIX.INI file when you are completed.

OneSix.ini Editing Utility

Using the Editing Utility

OneSix Editing Utility was designed to edit the OneSix Server INI file. It allows the user to view and edit the INI file parameters in an environment that is more user friendly than a standard text editor. Upon starting the editing utility, go to File--Open and select the OneSix.ini file or one of the OneSix backups such as OneSix.b01. When the file has been opened, the form will be filled with the current parameters in the OneSix Server file. The Utility has been arranged to follow the form of the ini file. The Utility has separate sections for Server parameters, Logging parameters, and parameters for each individual device.

Backup Files

Every time you open a OneSix ini file with this utility, a backup is created and placed in the same directory. The backup will have the same filename as the ini file; if the ini file is OneSix32.ini the backup will be called something like OneSix32.b05. The extension is always .b* where * is the number signifying the most recent backup. A temporary file is created to record the changes you make as you go along. You can always abandon your changes by choosing not to save them. If you do save them and you are unhappy with them you can open the most recent backup with this utility and save the backup as the ini file. You can change the name of the file you are editing by going to File—Save As.

Server and Logging Tabs

The first screen you will see is the server parameters. This screen allows you to quickly set the default units for all devices. If you click the 'Advanced' button you will be able to edit other parameters that OneSix normally uses defaults for. Click 'OK' on the advanced screen to return to the Server parameters. There is a separate tab for each section. You can click between 'Edit Server', 'Edit Logging', 'Edit Device', and 'Rearrange Devices.' The 'Edit Server' and 'Edit Logging' tabs give access to the parameters in those sections.

Device Tab

General

The 'Edit Device' tab shows a list of devices and the standard parameters for each device. You can choose a device on the list and press the 'Advanced' button to edit parameters specific to each device. You can choose a device by using the drop-down combo box. If you select the combo box (the text will be highlighted), you can use the keyboard up and down arrows to scroll through all the devices. The devices are listed in the combo box in the order in which they appear in the ini file, by label. If you know a specific device you want to edit you can enter its label into the 'Go To..' text box and press return or double click it. This will find the desired label in the combo box, if it is there.

Remove From Net

Also on this screen is found the 'Remove from net' button. Pressing this button and confirming your choice will remove the selected device from the ini file, and therefore from the whole 1-Wire net. If you need to restore this device to the net you will need to run OneSix Server and do a Search/Add or Manual Add.

Address Field

If you edit the address field and don't like your changes you can press 'Esc' to restore the original address that was read in from the .ini file. Be careful editing the address field! When you are in the Edit Device window the address field will not accept a blank or zeroed out address.

Rearrange Devices Tab

1-Wire only

This tab allows you to change the order that the devices are listed in the INI file; thereby changing the order in which each device will be polled. There are a couple of ways to change the order of the devices. If you highlight (single-click) a device in the list and click the up button or down button on the screen then the device selected will move up or down one place on the list. Also you can click in the device's Number column and drag it to its new place on the list. The list includes the following information about each device: its serial number, label, and description, plus '-prev', which is the place on the list that the device originally held at the start of the editing session.

Application Notes

Replacing a Device

If you want to remove a device from the net and add another in its place, follow these steps:

1) First, open up the ini file in the editing utility.

2) Go to the Rearrange tab and find the device you want to remove. Make a note of its position in the device list.

3) Now go to the Edit Device tab and remove the device from the ini file by selecting the device from the Device List and pressing the 'Remove from Net' button.

4) Start OneSix Server and hook up the device you want to add.

5) Have OneSix Server do a Search/Add routine so that it will find the new device.

6) Close OneSix.

7) Now open up the INI file again with the editing utility and go to the Rearrange tab.

8) Find the new device and click and drag in its leftmost column (Device Number) to drag it to the position you want it.

9) Now open OneSix and it will poll the new device in the correct order.

You can use the combination of 'Remove From Net' and 'Rearrange' for many different things. You can remove a device and reset its parameters and then put it back. You can put it back on the list anywhere you want to.