DATASHEET

Crestron to Heatmiser v3 Interface

Revision: 3.00
Date: 10 April 2013

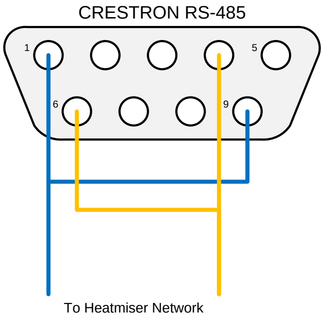
Summary

This datasheet relates to Ultamation's Heatmiser v3 interface module for Crestron control systems. It provides the essential information for integration between the Heatmiser system and the Crestron control processor, and for programming of the module with a host Crestron program.

Installation Notes

The Crestron system is connected to a normal, standalone, Heatmiser installation directly via the Heatmiser RS-485 data bus, which in turn connects each of the Heatmiser thermostats, using standard Crestron COM ports configured for RS-485 operation.

The pin-out for the Crestron COM port is given below, along with the configuration settings for the program.



Setting	Value
Baud	4800
Data bits	8
Parity	None
Stop bits	1
Hardware flow control	None
Software flow control	None

Although a manifold (e.g. UH-1) will normally be present, which makes a convenient connection point for the Crestron processor, the thermostats can be connected directly to the processor (given an appropriate power supply), which can be useful when testing communications and programming.

You may connect the heating system to the Crestron systems with only RS232 ports using an RS232 to RS485 converter. This has been successfully tested using powered converters.

The module has been tested in conjunction with the Heatmiser TouchPad. In such installations, the Crestron processor is connected to the C & D connections while the Heatmiser network is connected to the Y & B connections.

Programming Notes

Each of the module files should be placed either in the host program's project folder, or to make the Heatmiser interface available to all Crestron programs, in the SIMPL Windows installation's User Macro (for .umc files) and User SIMPL+ (for .usp and .ush files) directories. This pdf should be placed in both directories for SIMPL's F1 help function to work properly.

The module is broken into two parts:

1. A single core marshalling module that handles all communications between the thermostat modules (see 2) and the RS-485 communications to the physical thermostats.
2. One or more thermostat modules that present the control and feedback signals to the host program. It is normal to have one instance for each physical thermostat.

The Core Module

This module sits between the Crestron COM port and the thermostats.

1. Connect the module's RS485_Rx$ input signal to the rx$ output signal on the appropriate COM port.
2. Connect the module's RS485_Tx$ output signal to the tx$ input signal on the same COM port as step 1.
3. Connect each of the module's To_TStat_Addr_n$ output signals (where n is the numeric address of the physical thermostat) to the From_Core$ input signal of each thermostat module in the program.
4. The module's From_Tstats$ input will share a common signal with all the thermostat modules' To_Core$ ouptput.

Unless otherwise required, leave this parameter at its default setting of 'MC'.

Troubleshooting

If you are having difficulty communicating with the thermostats and you're sure that the wiring is correct, please ensure the following have been checked:

1. Addressing the thermostat modules correctly is critical. While the thermostats are addressed using a standard decimal format, the addresses in the program must use a two-character Hexadecimal notation.
    a. Example 1
    Thermostat at address 7 should be configured in the SIMPL program as "07". Note there are no leading "\x" or trailing "h" as sometimes used in SIMPL Windows.
    b. Example 2
    Thermostat at address 12 should be configured in the SIMPL program as "0C".
    c. Example 3
    Thermostat at address 18 should be configured in the SIMPL program as "12".
2. Ensure you're reading the correct temperature value, and the thermostat is configured to use the correct temperature sensor. An incorrect configuration her will result in a reading of 6553.5 degress.
3. Do not configure thermostats in the SIMPL program that have not been commissioned in Heatmiser system. This may lead to buffer overruns and sluggish performance.

The Thermostat Module

The From_Core$ and To_Core$ signals should be connected to the core module as described above.

Init

Instructs the module to query the physical thermostat for basic settings such as model type, and calibration information. This only needs to be triggered once, and can be given a value of ‘1’ as the core module will handle the queuing of requests. Edge triggered.

Poll_Frequently

When low, the module will automatically poll the stat for the specified attributes (see below) at the Idle_Poll_Period.

When high, the module will poll the stat at the Frequent_Poll_Period. This allows you to connect a “show page (or subpage)” join to the Poll_Frequently signal to automatically provide more responsive feedback to the user.

Power_On

Instruct the thermostat to power on. The current state is shown in the corresponding feedback output signal. Edge triggered. This is normally not required.

Power_Off

Instruct the thermostat to power off. The current state is shown in the corresponding feedback output signal. Edge triggered. This is normally not required.

RunMode_Normal

Instruct the thermostat to use the “normal” run mode and modify heating based on the thermostats clock settings. The current state is shown in the corresponding feedback output signal. Edge triggered.

RunMode_Frost_Protect

Instruct the thermostat to use the “frost protect” run mode and call for heating only when the temperature may approach freezing conditions. The current state is shown in the corresponding feedback output signal. Edge triggered.

HotWater_Request_Use_Timer

On a rising edge, the thermostat will use its internal time clock to request Hot Water

HotWater_Request_On

On a rising edge, the thermostat will request Hot Water, overriding the internal time clock.

SetPoint_x10#

Set the current set-point temperature. The temperature should be expressed as an analogue value representing the current temperature scale, multiplied by 10. Therefore for a thermostat set to the centigrade scale, 18°C should be expressed (using Crestron decimal notation) as 180d.

Simply passing a value to this signal will communicate the new set point to the thermostat. As such, it would be unwise to connect this to a rapidly ramping symbol which would generate large amounts of traffic on the Heatmiser data bus. Using an analog increment, with an increment of 10d, is suitable.

SetPoint_Raise

This is now the preferred way to adjust the set-point. Feedback is for the user immediate (from the SetPoint_x10_Fb# signal) but setting the stat is delayed until the adjustment has stopped for a few seconds.

SetPoint_Lower

As above.

Hold_Temp_ Duration_Mins#

Send an analogue value to set the thermostat's hold temp mode. The value is in minutes. Send 0 to cancel the mode.

Heating_Demand_Fb

Provides feedback over the current heating demand state.

HotWater_Demand_Fb

Provides feedback over the current hot water demand state (on Hot Water models only)

SetPoint_x10_Fb#

Provides the thermostat's current set-point as an analogue value. Again, this value is 10x the current degree (i.e. 170 would represent 17.0°C).

BuiltIn_Air_Temp_Fb#

Provides the thermostat's current built-in temperature sensor reading (when available) as an analogue value. Again, this value is 10x the current degree (i.e. 225 would represent 22.5°C).

Remote_Air_Temp_Fb#

Provides the thermostat's current remote temperature sensor reading (when available) as an analogue value. Again, this value is 10x the current degree (i.e. 140 would represent 14.0°C).

Floor_Temp_Fb#

Provides the thermostat's current floor slab temperature reading (when available). The value is 10x the current degree.

Calibration_Offset_Fb#

Provides the thermostat's current calibration offset as an analogue value. The representation of this value is undocumented.

Model_*

Identifies the currently recognised model.

(Parameter) Address

This parameter must holds the hexadecimal representation of the address (i.e. address 20 is set as 14) since the value is substituted into command strings.

(Parameter) Idle_Poll_Period

The interval between polls when "Poll_Frequently" is LOW

(Parameter) Frequent_Poll_Period

The interval between polls when “Poll_Frequently” is HIGH

(Parameter) Poll_Updates_*

Each of these parameters can take one of two settings – Ignore, or Query.

For a parameter set to Query, the attribute will be requested from the state on each update poll (determined by the periods described above)

If set to Ignore, the attribute will never be requested.

For example, if the user will never change the set-point on the physical stat, you can safely set “Poll_Updates_SetPoint” to Ignore. This will reduce the traffic on the RS485 bus and may improve overall performance.

Licence

Purchase of this software gives you (or the company, if purchased on behalf of an organisation) the right to implement to software in any number of your own projects only. The software may not be passed on to any 3rd party, even in a modified form.

Future revisions of the software, whether bug fixes or additional features, will be provided free of charge to existing customers. If additional features are requested, there may be an additional charge, and resulting fixes or features may become part of the standard module, therefore benefitting the existing customer base.

The software is provided in an unencrypted form allowing you to view the internal workings of each component, or make additions or modifications if you wish to do so. Ultamation cannot provide support for such modifications, though we will always provide assistance on a best efforts basis.