# ClimateTalk 2.0
# Command Reference

Document revision: V01
Release: June 12, 2013

## Abstract

ClimateTalk is a universal language for innovative, cost-effective solutions that optimize performance, efficiency and home comfort. The ClimateTalk Open Standards define a set of messages and commands to enable interoperability, enhanced user interface, and machine to machine control independent of the physical layer connecting the devices.

This document provides a single repository for all control commands and messages in the ClimateTalk Common Information Model that enable interoperable applications over any physical medium. ClimateTalk Applications are fully defined at Layer 7 of the OSI model by a combination of a Device Specific Application Profile, the Generic Application Specification and this document.

## Updates

This specification may be updated at any time and may be superseded by a more recent version or amended to from time to time. Users should be certain they are using the current ClimateTalk version and the latest revision of the documents.

The released versions of all specifications are available at http://www.ClimateTalk.org

## Version History

| ClimateTalk Version | Document Revision | Release Date | Comments |
|---|---|---|---|
| V 0.9 | | 2008-11-07 | Pre-Release |
| V 1.0 | | 2009-08-24 | Initial Release |
| V 1.1 | | 2011-06-23 | Errata Package |
| V 1.3 | | 2011-11-02 | Additional Errata Updates, Revised Formatting |
| V 2.0 | 00 | 2013-01-18 | Version 2.0 Release - Incorporate Zoning, Water Heater, Crossover & update Remote Access Control Commands. Split CT-485 API requirements into separate document. |
| V 2.0 | 01 | 2013-06-12 | Updated 5.14.1.4.1 Network Node List Data Payload to remove references to addresses.<br><br>Updated Table 153 – Furnace Configuration Data to show dehumidification capability is a bit not a byte.<br><br>Updated Table 158 – Zone Controller Configuration Data to include report for humidification or dehumidification capable. |

## Contributors

The following is a list of ClimateTalk Alliance member companies that were actively engaged in the development of this standard:

A.O. Smith Water Products Company

ecobee inc.

EDC

Emerson Electric, Co.

EWC Controls, Inc.

Microchip Technologies, Inc.

Nogginhaus, LLC.

Research Products Corp.

Rheem Manufacturing Company

Zonefirst

# Table of Contents

# List of Figures



# List of Tables

# 1.0 Overview

## 1.1 Purpose

ClimateTalk is an open standard that defines a set of messages and commands to enable interoperability, enhanced user interface, and machine to machine control independent of the physical layer connecting the devices.

The messages and commands defined by ClimateTalk Information Model (CIM) are the presentation and application layers as defined by the OSI Model[1]. ClimateTalk Applications are fully defined at Layer 7 of the OSI model by a combination of a Device Specific Application Profile, the Generic Application Specification and the Command Reference.

ClimateTalk messages can be carried over any physical medium following the OSI model. The ClimateTalk Presentation Layer defines how messages are executed over the various physical mediums in use.

CT-485 and CT-LWP are wired serial physical and network layers designed to support the formation of ClimateTalk networks and transport ClimateTalk messages, but other OSI based protocols – including wireless transports - can be used as well.

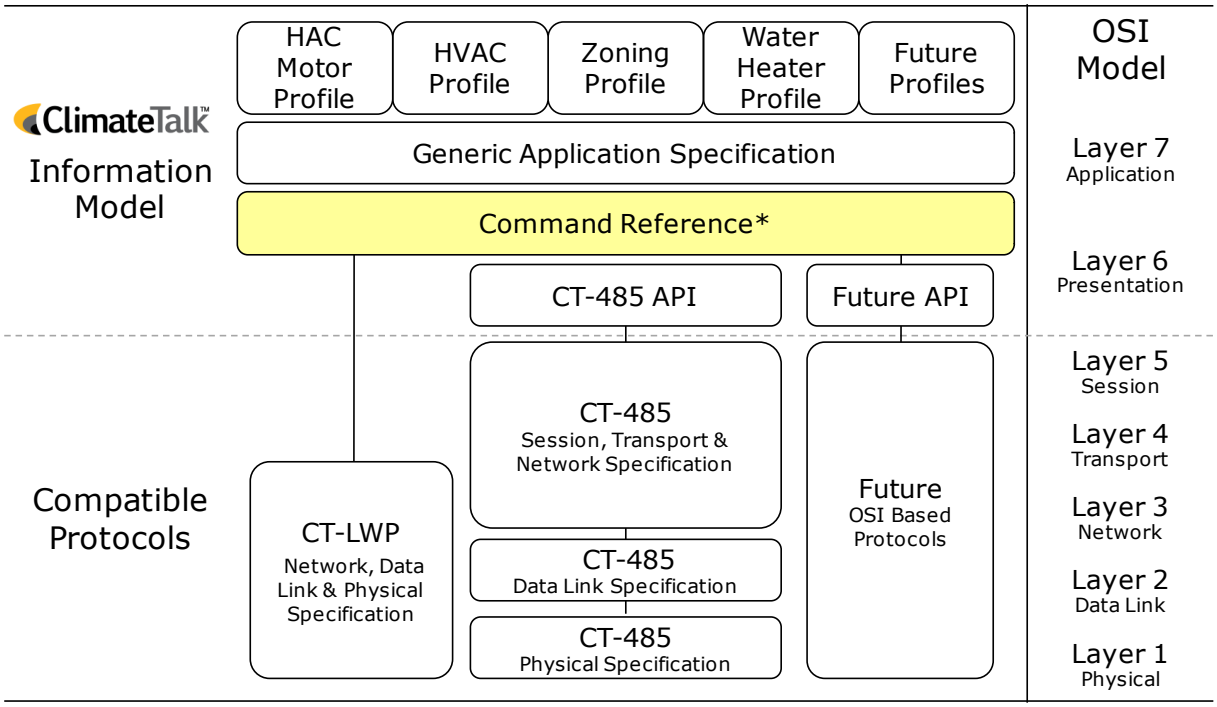
## 1.2 Scope

This document provides a single repository for all control commands and messages in the ClimateTalk Common Information Model. ClimateTalk Applications are fully defined at Layer 7 of the OSI model by a combination of a Device Specific Application Profile, the Generic Application Specification and this document.

The ClimateTalk Open Standards package shown in Figure 1 - OSI Layers for ClimateTalk Implementation prescribes the mandatory requirements to ensure proper network formation of interoperable devices. Membership in the ClimateTalk Alliance as well as successful completion of mandatory conformance testing is required for listing a product as a ClimateTalk Certified Device.

[1] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=20269

**Figure 1 - OSI Layers for ClimateTalk Implementation**




*This document

This document also defines the testable requirements used to validate that a device is behaving properly within a ClimateTalk network. Each device must comply with the mandatory requirements defined in this document as well as all other ClimateTalk standards applicable to the device functionality.

# 2.0 Normative References

A good understanding of the most recent version of the following documents is required to apply the contents of this specification correctly.

ClimateTalk Generic Device Specific Application Specifications

- *HVAC Application Profile*
- *HAC Motor Application Profile*
- *Zoning Application Profile*
- *Water Heater Application Profile*

*ClimateTalk Generic Application Specification*

*ClimateTalk CT-485 Application Protocol Interface*

*ClimateTalk CT-485 Networking Specification*

*ClimateTalk CT-485 Data Link Specification*

*ClimateTalk CT-485 Physical Specification*

*ClimateTalk CT-LWP Specification*

# 3.0 Terminology

## 3.1 Definitions

| | |
|---|---|
| **ACK** | Message to reflect proper receipt and ability to handle request in the data payload. |
| **Alpha-numeric ASCII** | Characters 0x20 to 0x7e of the ASCII character set. |
| **Control ID** | Unique 2-byte Identifier assigned to each device by the ClimateTalk Alliance that is used to identify the control application. |
| **HVAC Subsystem Nodes** | Any subsystem involved in the control of HVAC operation is considered a HVAC Subsystem Node. Examples: Thermostat, Furnace, Air Handler, Air Conditioner, and Heat Pump. |
| **Manufacturer ID** | Unique 2-byte Identifier assigned to each manufacturer by the ClimateTalk Alliance that is used to identify the subsystem. |
| **NAK** | Message to reflect a misunderstanding or failure to handle the request in the data payload. |
| **Profile** | Set of rules governing the implementation of certain aspects of the protocol, which will include timings and communication rules to function properly. |

## 3.2 Acronyms

**CT** ClimateTalk

**DBID** Database Identification

**IEEE** Institute of Electrical and Electronics Engineers

**MDI** Message Data Interface

**RPM** Revolutions Per Minute

**TBC** To Be Calculated

**TBD** To Be Determined

**XOVER** Crossover formerly known as the OBBI

## 3.3 Number Notation

Hexadecimal values shall be preceded by "0x", e.g. 0x11, 0x0F.
Numbers written with no precedent shall be assumed to be decimal, e.g. 17, 9.

## 3.4 Word Usage

The conventions used in this document are modelled after the definitions of the 2009 IEEE Standards Style Manual. The IEEE Standards Style Manual may be obtained from http://standards.ieee.org/guides/style/.

**can** Equivalent to *is able to* or *is capable of*.

**may** Equivalent to *is permitted to* or *is allowed to*. The use of *may* means that something is optional, and does not imply a requirement.

**must** Used to describe situations where no other course of action is possible.

**shall** Equivalent to *is required to*. Use of the word *shall* means that the specification shall be implemented exactly as described in order to ensure correct operation and interoperability with other devices.

**should** Equivalent to *is recommended that*. This is used in situations where there are several possible options, but one option is preferable to the others.

# 4.0 Message Rules

## 4.1 CIM Format

The Command Reference defines the structure of the routing and data payload for each message type;

**Table 1 – CIM Message Structure Example**

| Element |
| --- |
| Routing |
| Data Payload |

## 4.2 Routing

For all CIM Get/Set transactions, the application will need to specify routing parameters to be passed to the network in order to understand the intention of the initiating transaction.

The CIM specified routing parameters shall include the following four parameters:

Send Method – Type of routing

Send Parameter 1 - Variable Support Data for the Sent Method

Send Parameter 2 - Variable Support Data for the Sent Method

Source Node – Refer to Annex B – Node Type Table

Please refer to the *ClimateTalk Generic Application Specification* for more detailed information of the type of routing the CIM can implement.

In this specification each different CIM command reference has routing parameters detailed using the following structure reference.

```
struct Routing {
        UInt8    SendMethod;
        UInt8    SendParam1;
        UInt8    SendParam2;
        UInt8    Source Node;        // If not set, RFD/FFD sets
};
```

## 4.3 Data Payload

For all CIM Set/Get transactions, the application shall provide an indication of the data payload passed to the network.

The CIM specified data payload parameters shall include the following two parameters:

Length – The size of the Data Payload
DataPayload - Pointer to Data Payload

Please refer to the ClimateTalk Generic Application Specification for more detailed information of the type of routing the CIM can implement.

In this specification each different CIM command reference has data payload detailed using the following structure reference.

```
struct Data {
        UInt8    Length;
        UInt8*   DataPayload;
};
```

# 5.0 Command Reference

The following sections define the different commands available to applications.

Each command is shown as a generic pseudo-code prototype. This format allows the functions to be identified in generic terms while allowing the underlying API reference documentation to define protocol specific message format.

## 5.1 Get Configuration

CIM applications have mandatory and optional configuration data as defined in the applicable application profile.

This section details the how a CIM application can initiate and process requests and responses.

### 5.1.1 GetConfiguration

CIM applications can initiate a requesting transaction to receive a subsystem configuration data, if applicable.

#### 5.1.1.1 Prototype:

void GetConfiguration(Routing, RequestFunctPtr callback)

#### 5.1.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response

#### 5.1.1.3 Result

Returns Nothing.

#### 5.1.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.1.2 GiveConfiguration

The format of the configuration data returned in the response is device-specific, and is defined in the Configuration Message Data Interface (MDI) found in Section 7.3 or applicable application profile.

#### 5.1.2.1 Prototype:

Data GiveConfiguration(ResponseFunctPtr callback)

#### 5.1.2.2 Parameters

- ➔ callback — A callback function that process the request

#### 5.1.2.3 Result

Returns the length of and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.1.2.4 Comments

If a CIM application has configuration data to be utilized by other CIM applications, this method is utilize of transmitted the application's configuration data via the network.

## 5.2 Get Status

CIM applications have mandatory and optional status data as defined in the applicable application profile.

This section details the how a CIM application can initiate and process requests and responses.

### 5.2.1 GetStatus

CIM applications can initiate a requesting transaction to receive a subsystem status data, if applicable.

#### 5.2.1.1 Prototype:

Void GetStatus(Routing, RequestFunctPtr callback)

#### 5.2.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response

#### 5.2.1.3 Result

Returns Nothing.

#### 5.2.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.2.2 GiveStatus

The format of the status data returned in the response is device-specific, and is defined in the Status Message Data Interface (MDI) found in Section 7.4 or located in the applicable application profile.

#### 5.2.2.1 Prototype:

Data GiveStatus(ResponseFunctPtr callback)

#### 5.2.2.2 Parameters

➔ callback A callback function that process the request

#### 5.2.2.3 Result

Returns the length of the data and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.2.2.4 Comments

If a CIM application has status data to be utilized by other CIM applications, this method is utilize of transmitted the application's status data via the network.

## 5.3 Set Control Command

Control Commands provide for control of a subsystem over the network. Only one control command shall be transmitted in each Control Command packet.

Control Commands may require additional data to fulfill the command. Command data is defined with each control command. If more data is required than is allowed in a single ClimateTalk packet, multiple packets may be used. The command code shall be included with each packet.

The Command Data field in the Control Command response packet may be either an echo of the request data or left empty. In the latter case, the payload of the response will consist solely of the command code. Devices shall be prepared to accept either response as legitimate.

The command code and command data shall be transmitted in Little Endian byte order.

The Control Commands for each profile are defined in that profile's specification.

This section details the how a CIM application can process control command information.

### 5.3.1 SetControlCommand

CIM applications can initiate a set transaction to transmit a control command to modify a subsystem, if applicable.

#### 5.3.1.1 Prototype:

void SetControlCommand (Routing, RequestFunctPtr callback, CtrlCmdCode, OptionalCCData)

#### 5.3.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ CtrlCmdCode — Refer to Section 6.0
- ➔ OptionalCCData — Refer to Section 6.0

#### 5.3.1.3 Result

Returns Nothing.

#### 5.3.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.3.2 ConfirmControlCommand

The format of the control command data response in return to the control command sent to a CIM application is generic.

#### 5.3.2.1 Prototype:

Data ConfirmControlCommand(ResponseFunctPtr callback)

#### 5.3.2.2 Parameters

- ➔ callback — A callback function that process the request

#### 5.3.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.3.2.4 Comments

The response data payload will have either two methods of properly receiving a response to a control command. Both of these methods received would be considered a proper and correct response to a request being made.

#### 5.3.2.4.1 Echoed Response

An echoed response would contain a data payload identical to the data payload received.

#### 5.3.2.4.2 Confirmation Code Response

A confirmation code response would contain a data payload consisting of two bytes: Confirmation code, <0xAC>, followed by Acknowledgment byte, <0x06>.

## 5.4 Set Display Message

Display devices such as thermostats or PC applications may allow any subsystem to display generic data and generic subsystem information. The display device will not know what is being displayed so it is up to the requesting subsystem to control the message and format.

Set Display Message is used to give display devices the message to display.

### 5.4.1 SetDisplayMessage

CIM applications can initiate a set transaction to transmit a display message to be displayed on a subsystem, if applicable.

#### 5.4.1.1 Prototype:

void SetDisplayMessage(Routing, RequestFunctPtr callback, NodeType, MessageLength, Message)

#### 5.4.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ NodeType — Node with Fault (Refer to Section 0)
- ➔ MessageLength — Length of the Diagnostic Message
- ➔ Message — Alpha-Numeric ASCII (Max 30 Characters)

#### 5.4.1.3 Result

Returns Nothing.

#### 5.4.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.4.2 ConfirmDisplayMessage

The format of the Display Message data response in return to the message sent to a CIM application is generic.

#### 5.4.2.1 Prototype:

Data ConfirmDisplayMessage(ResponseFunctPtr callback)

#### 5.4.2.2 Parameters

➔ callback A callback function that process the request

#### 5.4.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.4.2.4 Comments

The response data payload will have either two generic methods of properly receiving a response to send a display message. Both of these methods received would be considered a proper and correct response to a request being made.

##### 5.4.2.4.1 Echoed Response

An echoed response would contain a data payload identical to the data payload received.

##### 5.4.2.4.2 Confirmation Code Response

A confirmation code response would contain a data payload consisting of two bytes: Confirmation code, <0xAC>, followed by Acknowledgment byte, <0x06>.

## 5.5 Set Diagnostics

Devices on the network could periodically query subsystems to determine if there is a failure; however, it is up to the subsystem to inform the thermostat of its inability to perform its required function.

The diagnostic packet provides two types of fault codes. Major fault should be used for situations where the subsystem is inoperable. Minor fault should be used for situations where there is a problem or when maintenance is required but the subsystem remains operable.

A message may be included in the packet for display on a user-interface device.

### 5.5.1 SetDiagnosticMessage

CIM applications can initiate a set transaction to transmit a diagnostic message to be displayed on a subsystem, if applicable.

#### 5.5.1.1 Prototype:

void SetDiagnosticMessage(Routing, RequestFunctPtr callback, FaultMessage)

#### 5.5.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ NodeType — Node with Fault Refer to Section 0
- ➔ MajorCode — Major Fault Code (Hex)
- ➔ MinorCode — Minor Fault Code (Hex)
- ➔ MessageLength — Length of the Diagnostic Message
- ➔ FaultMessage — Optional Alpha-Numeric ASCII (Max 15 Characters)

#### 5.5.1.3 Result

Returns Nothing.

#### 5.5.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

If no two-digit alpha-numeric fault code is known or not needed to be displayed on the thermostat, then 0xFF would be transmitted in either the major or minor fault byte to the thermostat.

#### 5.5.1.5 Clearing a Diagnostics Message

The subsystem is responsible for clearing the diagnostic message if the fault condition goes away. To clear a diagnostic message, the subsystem shall send a Set Diagnostic message as defined in .

**Table 2 – Clearing a Set Diagnostics Message**

| **Field Name** | Node Type | Major Fault Code | Minor Fault Code | Message Length |
|---|---|---|---|---|
| **Field Length** | 1 Byte | 1 Byte | 1 Byte | 1 Byte |
| **Value** | Refer to Section 0 | 0x00 | 0x00 | 0x00 |

### 5.5.2 ConfirmDiagnosticMessage

The format of the Diagnostic Message data response in return to the message sent to a CIM application is generic.

#### 5.5.2.1 Prototype:

Data ConfirmDiagnosticMessage(ResponseFunctPtr callback)

#### 5.5.2.2 Parameters

- ➔ callback — A callback function that process the request

#### 5.5.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.5.2.4 Comments

The response data payload will have either two generic methods of properly receiving a response to send a display message. Both of these methods received would be considered a proper and correct response to a request being made.

##### 5.5.2.4.1 Echoed Response

An echoed response would contain a data payload identical to the data payload received.

##### 5.5.2.4.2 Confirmation Code Response

A confirmation code response would contain a data payload consisting of two bytes: Confirmation code, <0xAC>, followed by Acknowledgment byte, <0x06>.

## 5.6 Get Diagnostics

Subsystem could request diagnostic information to subsystem that may support this capability. The specific fault is identified by the major or minor fault code and the fault index. A fault index value of 0x00 indicates that the subsystem should return information on all faults.

If multiple faults are sent in a single packet, then a null (0x00) character shall separate each fault data field.

### 5.6.1 GetDiagnostics

CIM applications can initiate a requesting transaction to receive a subsystem fault message data, if applicable.

#### 5.6.1.1 Prototype:

void GetDiagnostics(Routing, RequestFunctPtr callback, FaultIndex)

#### 5.6.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ FaultIndex — Utilized to request specific fault (0x00 = All Faults)

#### 5.6.1.3 Result

Returns Nothing.

#### 5.6.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.6.2 GiveDiagnostics

The format of the fault message data returned in the response is device-specific and alpha-numeric ASCII with a maximum of 15 characters for each fault.

#### 5.6.2.1 Prototype:

Data GiveDiagnostics(ResponseFunctPtr callback)

#### 5.6.2.2 Parameters

➔ callback A callback function that process the request

#### 5.6.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.6.2.4 Comments

If a CIM application has fault data to be utilized by other CIM applications, this method is utilize of transmitted the application's configuration data via the network.

Data return shall be in the format of the following Fault Data Detail as seen in the following table:

**Table 3 - Fault Data Detail**

| Major Fault Code | Minor Fault Code | Fault Msg Length | Fault Message |
|---|---|---|---|
| 1 Byte | 1 Byte | 1 Byte | n Bytes |
| Manufacturer Defined | Manufacturer Defined | n | 0-15 Bytes |

### 5.6.3 Diagnostics Example

As defined in the Set Diagnostic message, the Fault Message is limited to a maximum of 15 characters. Thus, the fault data field for each fault is limited to a maximum of 18 bytes.

The following are examples of responses of request made to an application with a single and double fault.

**Example Payloads**

Single Fault Response:

Payload Length = 3 + FaultMsgLength(15)= 18
Major Fault Code = 0x12
Minor Fault Code = 0x00
Fault Msg Length = 15
Fault Message = "123456789012345"

Double Fault Response:

Payload Length = (3 + FaultMsgLength1(15) + 1 + 3 + FaultMsgLength2(5) = 27
Major Fault Code = 0x12
Minor Fault Code = 0x00
Fault Msg Length = 15
Fault Message = "123456789012345"
Null separator = 0
Major Fault Code = 0x34
Minor Fault Code = 0x00
Fault Msg Length = 5
Fault Message = "12345"

If an application receives a request for all faults or at a specific index when no fault is active, the application shall respond with the following data payload:

Payload Length = 4
Major Fault Code = 0
Minor Fault Code = 0
Fault Msg Length = 0

## 5.7 Get Sensor Data

CIM applications have mandatory and optional sensor data as defined in the applicable application profile.

This section details the how a CIM application can initiate and process requests and responses.

### 5.7.1 GetSensor

CIM applications can initiate a requesting transaction to receive a subsystem sensor data, if applicable.

#### 5.7.1.1 Prototype:

void GetSensor(Routing, RequestFunctPtr callback)

#### 5.7.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response

#### 5.7.1.3 Result

Returns Nothing.

#### 5.7.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.7.2 GiveSensor

The format of the sensor data returned in the response is device-specific, and is defined in the Sensor Message Data Interface (MDI) found in Section 7.5 or the applicable application profile.

#### 5.7.2.1 Prototype:

Data GiveSensor(ResponseFunctPtr callback)

#### 5.7.2.2 Parameters

- ➔ callback — A callback function that process the request

#### 5.7.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.7.2.4 Comments

If a CIM application has sensor data to be utilized by other CIM applications, this method is utilize of transmitted the application's sensor data via the network.

## 5.8 Set Identification

The Identification MDI describes a device and its identity. The information contained in it is unique for each device type. The Identification MDI is defined in the profile specification for each device.

Set Identification allows the optional part of the Identification MDI to be written to over the network, permitting on-site configuration of devices.

To ensure that data set previously is retained, a device intending to Set Identification information in another device should first send a Get Identification request to the other device. The identification information received should then be updated with the desired information while retaining any previously stored identification data.

This section details the how a CIM application can initiate and process requests and responses.

### 5.8.1 SetIdent

CIM applications can initiate a set transaction to transmit new identification data to modify a subsystem, if applicable.

#### 5.8.1.1 Prototype:

void SetIdent (Routing, RequestFunctPtr callback, OptionalIdent)

#### 5.8.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ OptionalIdent — Refer to Section 7.2.2

#### 5.8.1.3 Result

Returns Nothing.

#### 5.8.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.8.2 ConfirmIdent

The format of the set identification response in return to the request sent to a CIM application is generic.

#### 5.8.2.1 Prototype:

Data ConfirmIdent(ResponseFunctPtr callback)

#### 5.8.2.2 Parameters

- ➔ callback — A callback function that process the request

#### 5.8.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.8.2.4 Comments

The response data payload will have either two methods of properly receiving a response to a identification. Both of these methods received would be considered a proper and correct response to a request being made.

##### 5.8.2.4.1 Echoed Response

An echoed response would contain a data payload identical to the data payload received.

##### 5.8.2.4.2 Confirmation Code Response

A confirmation code response would contain a data payload consisting of two bytes: Confirmation code, <0xAC>, followed by Acknowledgment byte, <0x06>.

## 5.9 Get Identification

The Identification MDI format is specific to each device type, and is defined in the profile specification for the device.

This section details the how a CIM application can initiate and process requests and responses.

### 5.9.1 GetIdent

CIM applications can initiate a requesting transaction to receive a subsystem identification data, if applicable.

#### 5.9.1.1 Prototype:

void GetIdent(Routing, RequestFunctPtr callback)

#### 5.9.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response

#### 5.9.1.3 Result

Returns Nothing.

#### 5.9.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.9.2 GiveIdent

The format of the identification data returned in the response is device-specific, and is defined in the Identification Message Data Interface (MDI) found in Section 7.2.1.

#### 5.9.2.1 Prototype:

Data GiveIdent(ResponseFunctPtr callback)

#### 5.9.2.2 Parameters

➔ callback A callback function that process the request

#### 5.9.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.9.2.4 Comments

If a CIM application has Identification data to be utilized by other CIM applications, this method is utilize of transmitted the application's identification data via the network.

## 5.10 Set Application Shared Data to Network

Any subsystem using shared data and requiring network replication of shared data for future population in replacement controls shall use this message to send its own shared data to the network for storage. Shared data shall be sent out to the network when one of the following events occurs:

1. The application validates new shared data from any source, either due to a Set Factory Shared Data to Application as mentioned in section 5.21 or by voluntarily seeking and accepting data from any of the available sources.
2. A network node list is received.
3. The shared data of a device changes due to internal changes in configuration or user interaction.

This section details the how a CIM application can initiate and process requests and responses.

### 5.10.1 SetAppNetSharedData

CIM applications can initiate a set transaction to transmit new application shared data to the network, if applicable.

#### 5.10.1.1 Prototype:

void SetAppNetSharedData (Routing, RequestFunctPtr callback, AppSD)

#### 5.10.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ AppSD — Refer to Section 5.10.1.4.1

#### 5.10.1.3 Result

Returns Nothing.

#### 5.10.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

##### 5.10.1.4.1 Application Shared Data Payload

The first byte of the application shared data payload is the called the sector node type. The sector node type is used to be sent to the network to properly store and access the application shared data which follows this byte. The application's node type is used as this index for when the data needs to be retrieved.

The bytes following the sector are what are known as the Application Shared Data. The Application Shared Data shall have the Shared Data Length, Control ID, Manufacturer ID, and Application node type.

The first byte in the Application Shared Data would be the Shared Data Length. The maximum Application Shared Data size is 200 bytes.

The next two bytes will be the Control ID which is provided by CT Alliance. This ID is a two byte integer which is transmitted little endian. Following these bytes, the Manufacturer ID will be transmitted which is another two byte CT Alliance provided integer that is little endian. Control ID and Manufacturer ID shall be represented as follows: If company A provides a circuit board to company B for the use is a company B subsystem, the Control ID would be the ClimateTalk OEM ID for company A and the Manufacturer ID would be company B.

Following these bytes, the application node type will be sent to be used for validation of the data at a later time.

Finally, the application specific data would be sent that would include the unique data to be stored on the network and later retrieved.

**Table 4 – Set Application Shared Data to Network Request Format**

| Field Name | Sector Node Type | Shared Data Length | Control ID | Manufacturer ID | App Node Type | Application Data |
|---|---|---|---|---|---|---|
| **Field Length** | 1 Byte | 1 Byte | 2 Bytes | 2 Bytes | 1 Byte | 0-194 Bytes |
| **Value** | Refer to Section 0 | 6-200 | Assigned by ClimateTalk Alliance | Assigned by ClimateTalk Alliance | Refer to Section 0 | TBC |

### 5.10.2 ConfirmAppNetSharedData

The format of the application shared data response in return to the message sent to a CIM application is generic.

#### 5.10.2.1 Prototype:

Data ConfirmAppNetSharedData (ResponseFunctPtr callback)

#### 5.10.2.2 Parameters

➔ callback A callback function that process the request

#### 5.10.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.10.2.4 Comments

The response data payload will have one method of properly receiving a response to a request of data.

##### 5.10.2.4.1 Echoed Response

An echoed response would contain a data payload identical to the data payload received.

## 5.11 Get Application Shared Data from Network

This section details the how a CIM application can initiate and process requests and responses.

### 5.11.1 GetAppNetSD

CIM applications can initiate a requesting transaction to receive a subsystem applications shared data, if applicable.

#### 5.11.1.1 Prototype:

void GetAppNetSD(Routing, RequestFunctPtr callback, SectorNodeType)

#### 5.11.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ SectorNodeType — Refer to Section 5.10.1.4.1

#### 5.11.1.3 Result

Returns Nothing.

#### 5.11.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.11.2 GiveAppNetSD

The format of the application shared data returned in the response is device-specific, and is defined in the format as defined in Section 5.10.1.4.1.

#### 5.11.2.1 Prototype:

Data GiveAppNetSD(ResponseFunctPtr callback)

#### 5.11.2.2 Parameters

- ➔ callback — A callback function that process the request

#### 5.11.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.11.2.4 Comments

Get Application Shared Data from Network is sent by the subsystem to the coordinator to request that shared data previously stored on the network be transmitted to the subsystem.

The coordinator uses the sector node type in the request as an index to locate the correct set of shared data to transmit to the subsystem.

The response follows the same format as specified in Section 5.10.1.4.1

## 5.12 Set Manufacturer Device Data

This message type transmits manufacturer specific configuration data.

The Set Manufacturer Device Data will be responded to with a data payload echoing the data payload from the original Set Request.

This section details the how a CIM application can initiate and process requests and responses.

### 5.12.1 SetMfgDeviceData

CIM applications can initiate a set transaction to transmit manufacturer specific data to modify a subsystem, if applicable.

#### 5.12.1.1 Prototype:

void SetMfgDeviceData (Routing, RequestFunctPtr callback, MfgData)

#### 5.12.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ MfgData — Manufacturer Defined

#### 5.12.1.3 Result

Returns Nothing.

#### 5.12.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.12.2 ConfirmMfgDeviceData

The format of the manufacturing device data response in return to the set sent to a CIM application is generic.

#### 5.12.2.1 Prototype:

Data ConfirmMfgDeviceData (ResponseFunctPtr callback)

#### 5.12.2.2 Parameters

➔ callback A callback function that process the request

#### 5.12.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.12.2.4 Comments

The response data payload will have either two methods of properly receiving a response to a identification data. Both of these methods received would be considered a proper and correct response to a request being made.

##### 5.12.2.4.1 Echoed Response

An echoed response would contain a data payload identical to the data payload received.

##### 5.12.2.4.2 Confirmation Code Response

A confirmation code response would contain a data payload consisting of two bytes: Confirmation code, <0xAC>, followed by Acknowledgment byte, <0x06>.

## 5.13 Get Manufacturer Device Data

This message allows the retrieval of Manufacturing Device Data sent to the device by the Set Manufacturer Device Data message.

This section details the how a CIM application can initiate and process requests and responses.

### 5.13.1 GetMfgDeviceData

CIM applications can initiate a requesting transaction to receive a subsystem manufacturer device data, if applicable.

#### 5.13.1.1 Prototype:

void GetMfgDeviceData(Routing, RequestFunctPtr callback)

#### 5.13.1.2 Parameters

➔ Routing Refer to Section 4.2
➔ callback A callback function that process the response

#### 5.13.1.3 Result

Returns Nothing.

#### 5.13.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.13.2 GiveMfgDeviceData

The format of the manufacturer data returned in the response is device-specific, and is defined by each manufacturer.

#### 5.13.2.1 Prototype:

Data GiveMfgDeviceData(ResponseFunctPtr callback)

#### 5.13.2.2 Parameters

➔ callback A callback function that process the request

#### 5.13.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.13.2.4 Comments

If a CIM application has status data to be utilized by other CIM applications, this method is utilize of transmitted the application's status data via the network.

## 5.14 Set Network Node List

This message type is sent from the coordinator to subsystems. It lists which subsystems are active on the network. When a subsystem sees itself on the list, it knows it is considered active on the network.

The response to the Network Node List is an echoed response to confirm from that subsystem a valid understanding of the network.

This section details the how a CIM application can initiate and process requests and responses.

### 5.14.1 Set Network Node List

Coordinators of network protocols can initiate a set network node list transaction to transmit new network node lists.

#### 5.14.1.1 Prototype:

void SetNetworkNodeList(Routing, RequestFunctPtr callback, AppSD)

#### 5.14.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ NodeList — Refer to Section 5.14.1.4.1

#### 5.14.1.3 Result

Returns Nothing.

#### 5.14.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

#### 5.14.1.4.1 Network Node List Data Payload

The packet helps identify the nodes that have been addressed by the coordinator and available on the network.

**Table 5 – Set Network Node List Request**

| **Field Name** | Data | | | |
|---|---|---|---|---|
| **Field Length** | 1 Byte | 1 Byte | … | 1 Byte |
| **N** | Coordinator Node Type | Index 1 Node Type | | Index N Node Type |

The entire node list must be inspected for nodes. A zero in the node list shall not be considered the end of the node list. For instance, a coordinator would replace a node type in the node list with a zero if that node were to be removed from the network.

### 5.14.2 Confirm Network Node List

The format of the set network node list response in return to the message sent to a CIM application is generic.

#### 5.14.2.1 Prototype:

Data ConfirmNetworkNodeList(ResponseFunctPtr callback)

#### 5.14.2.2 Parameters

- ➔ callback — A callback function that process the request

#### 5.14.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.14.2.4 Comments

The response data payload will have either two methods of properly receiving a response to a request of data. Both of these methods received would be considered a proper and correct response to a request being made.

##### 5.14.2.4.1 Echoed Response

An echoed response would contain a data payload identical to the data payload received.

##### 5.14.2.4.2 Confirmation Code Response

A confirmation code response would contain a data payload consisting of two bytes: Confirmation code, <0xAC>, followed by Acknowledgment byte, <0x06>.

## 5.15 Get Direct Memory Access (DMA) Read

Information stored in Message Data Interfaces (MDIs) is accessed either through Message Type Code that retrieve the entire set of data in the MDI (e.g. Get Status), or through DMA messages that permit the retrieval of specific pieces of information from the MDI. Information in the MDI is referenced using a Database Identification (DB ID) number.

This section details the how a CIM application can initiate and process requests and responses.

### 5.15.1 GetDMARead

CIM applications can initiate a requesting transaction to receive a subsystem applications specific range of data, if applicable.

#### 5.15.1.1 Prototype:

void GetDMARead(Routing, RequestFunctPtr callback, MessageReadCode, MDIPacketID, DBID, Range)

#### 5.15.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ MessageReadCode — Refer to Table 7
- ➔ MDIPacketID — MDI Packet Identifier (Default = 0)
- ➔ DBID — Initial DB ID

➔ Range[2] Number of bytes after Initial DB ID requested

#### 5.15.1.3 Result

Returns Nothing.

#### 5.15.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

##### 5.15.1.4.1 DMA Read Data Payload

The following packet outlines how a DMA Read request would be made to retrieve a specific piece of data from a device.

**Table 6 - DMA Request Format**

| Field Name | MDI Read Code | Packet ID | Start DB ID | Range |
|---|---|---|---|---|
| **Field Length** | 1 Byte | 1 Byte | 1 Byte | 1 Byte |
| **Value** | See Table 7 | 0x00 | MDI Defined | Length Bytes Returned |

The DMA Read request may be one DB ID or a range of consecutive DB IDs. If non-consecutive data are required, multiple requests must be made, one for each DB ID or range of IDs.

**Table 7 – MDI Read Codes**

| MDI | Value |
|---|---|
| Configuration | 0x01 |
| Status | 0x02 |
| Sensor | 0x07 |

Every MDI Code that has data will be given a database identification number called the DB ID. For each Packet ID of data per that message type, the DB ID will start over with the first byte being one. DMA Read Requests can only ask for a complete byte of data starting at that DB ID.

[2] Bytes returned from DB will be Range + 1. Range equaling zero means one byte of data will be returned.

Please refer to the associated message ID's MDI for the proper DB ID number to be requested. The Start DB ID is where in the block of data the request is specifically requesting. And, the last byte tells the requesting device the range of data bytes requested starting from that DB ID number. A range of zero means to return one byte.

### 5.15.2 GiveDMARead

The format of the returned in the response is Message Type Code MDI specific, and is defined in Section 0.

#### 5.15.2.1 Prototype:

Data GiveDMARead(ResponseFunctPtr callback)

#### 5.15.2.2 Parameters

➔ callback A callback function that process the request

#### 5.15.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.15.2.4 Comments

A CIM application would process and respond according to the defined MDI per the MDI code received to the requesting CIM application.

## 5.16 Set Direct Memory Access (DMA) Write

Information stored in Message Data Interfaces (MDIs) is accessed either through Message Type Code that modifies the entire set of data in the MDI, or through DMA messages that permit the modification of specific pieces of information from the MDI. Information in the MDI is referenced using a Database Identification (DB ID) number.

This section details the how a CIM application can initiate and process requests and responses.

### 5.16.1 SetDMAWrite

CIM applications can initiate a transaction to a subsystem applications to change specific range of data, if applicable.

#### 5.16.1.1 Prototype:

void SetDMAWrite(Routing, RequestFunctPtr callback, MessageIDCode, MDIPacketID, DBID, Range)

#### 5.16.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ MessageIDCode — Refer to Table 9
- ➔ MDIPacketID — MDI Packet Identifier (Default = 0)
- ➔ DBID — Initial DB ID
- ➔ Range[2] — Number of bytes after Initial DB ID requested
- ➔ Value[Range[2]] — Array of Values used to Write

#### 5.16.1.3 Result

Returns Nothing.

#### 5.16.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

##### 5.16.1.4.1 DMA Write Data Payload

Direct Memory Access Write is used to write data directly into an MDI. The message gives the MDI to write to and the starting Database ID value. The database data format is defined by each profile.

**Table 8 – Direct Memory Write Access Data Format**

| Field Name | Field Length | Value |
|---|---|---|
| MDI Code | 1 Byte | See Table 9 |
| Packet ID | 1 Byte | 0x00 |
| Start DB ID | 1 Byte | Per Profile Specification |
| DB ID Value 1 | | |
| DB ID Value 2 | | |
| … | | |
| DB ID Value n | | |

The DMA Write request may be one DB ID or a range of consecutive DB IDs. If non-consecutive data are required, multiple requests must be made, one for each DB ID or range of IDs.

**Table 9 – MDI Write Codes**

| MDI | Value |
|---|---|
| Configuration | 0x01 |

### 5.16.2 ConfirmDMAWrite

The format of the set network node list response in return to the message sent to a CIM application is generic.

#### 5.16.2.1 Prototype:

Data ConfirmDMAWrite(ResponseFunctPtr callback)

#### 5.16.2.2 Parameters

➔ callback A callback function that process the request

#### 5.16.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.16.2.4 Comments

The response data payload will have either two methods of properly receiving a response to a request of data. Both of these methods received would be considered a proper and correct response to a request being made.

##### 5.16.2.4.1 Echoed Response

An echoed response would contain a data payload identical to the data payload received.

##### 5.16.2.4.2 Confirmation Code Response

A confirmation code response would contain a data payload consisting of two bytes: Confirmation code, <0xAC>, followed by Acknowledgment byte, <0x06>.

## 5.17 Set Manufacturer Generic Data

Set Manufacturer Generic Data is used to transmit proprietary data. This message may be used to encapsulate proprietary data payloads. The data may be in any format provided it conforms to the rules for payload length, etc.

This section details the how a CIM application can initiate and process requests and responses.

### 5.17.1 SetMfgGenericData

CIM applications can initiate a transaction to a subsystem applications to change manufacturer proprietary specific data, if applicable.

#### 5.17.1.1 Prototype:

void SetMfgGenericData (Routing, RequestFunctPtr callback, ManufactureID, ManufacturerData)

#### 5.17.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ ManufactureID — Refer to CT Alliance Adopter ID
- ➔ ManufacturerData — Generic Data

#### 5.17.1.3 Result

Returns Nothing.

#### 5.17.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.17.2 ConfirmMfgGenericData

The format of the set manufacturer generic data response in return to the message sent to a CIM application is generic.

#### 5.17.2.1 Prototype:

Data ConfirmMfgGenericData(ResponseFunctPtr callback)

#### 5.17.2.2 Parameters

- ➔ callback — A callback function that process the request

#### 5.17.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.17.2.4 Comments

The response data payload will have either two methods of properly receiving a response to a request of data. Both of these methods received would be considered a proper and correct response to a request being made.

##### 5.17.2.4.1 Echoed Response

An echoed response would contain a data payload identical to the data payload received.

##### 5.17.2.4.2 Confirmation Code Response

A confirmation code response would contain a data payload consisting of two bytes: Confirmation code, <0xAC>, followed by Acknowledgment byte, <0x06>.

## 5.18 Get Manufacturer Generic Data

Get Manufacturer Generic Data is used to retrieve information stored in a proprietary format. Just as with the Set Manufacturer Generic Data, the first field of the payload shall be the Manufacturer ID, followed by the proprietary data.

This section details the how a CIM application can initiate and process requests and responses.

### 5.18.1 GetMfgGenericData

CIM applications can initiate a requesting transaction to receive a subsystem status data, if applicable.

#### 5.18.1.1 Prototype:

void GetMfgGenericData(Routing, RequestFunctPtr callback, ManufacturerID, ManufacturerData)

#### 5.18.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ ManufactureID — Refer to CT Alliance Adopter ID
- ➔ ManufacturerData — Optional Generic Data

#### 5.18.1.3 Result

Returns Nothing.

#### 5.18.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.18.2 GiveMfgGenericData

The format of the manufacturer generic data returned in the response is device-specific, and is defined by each manufacturer.

#### 5.18.2.1 Prototype:

Data GiveMfgGenericData(ResponseFunctPtr callback)

#### 5.18.2.2 Parameters

➔ callback A callback function that process the request

#### 5.18.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.18.2.4 Comments

If a CIM application has status data to be utilized by other CIM applications, this method is utilize of transmitted the application's status data via the network.

##### 5.18.2.4.1 Get Manufacturer Generic Data Response

The response back from this type of transaction is defined per each manufacturer's specific requirements. The first two bytes of the response message data payload will still be required to contain the Manufacturer ID as in the requesting message, but the remaining part of the data payload may be defined as required specific to the proprietary protocol in use.

## 5.19 Get User Menu

The ability for display devices like a thermostat or software application to interact with subsystems to provide the capability to display user configurable displays is a key concept in the development of Climatetalk. Subsystems have become over time devices that require more configurations in the field due to the many variations subsystems may control.

The User Menu procedure is discussed at length in the Generic Application Specification. Refer to that document for details on how user menus are requested and retrieved via ClimateTalk. Below is a short description of the meaning of each of the fields in the request and response messages.

This section details the how a CIM application can initiate and process requests and responses.

### 5.19.1 GetUserMenu

CIM applications can initiate a requesting transaction to receive a subsystem applications user menu data, if applicable.

#### 5.19.1.1 Prototype:

void GetUserMenu(Routing, RequestFunctPtr callback, MessageReadCode, MDIPacketID, DBID, Range)

#### 5.19.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ UserMenuData — Refer to Section 5.19.1.4.1

#### 5.19.1.3 Result

Returns Nothing.

#### 5.19.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

##### 5.19.1.4.1 Get User Menu Data Payload

ClimateTalk menus have 4 levels: subsystem (corresponds to Menu File), main menu, sub menu and child/children. Children are the specific options available. The Menu File, Main Menu and Sublevel fields specify which menu or menu items to retrieve.

**Table 10 – Get User Menu Request Format**

| Field Name | Field Length | Value |
|---|---|---|
| Menu File | 1 Byte | 0x01 |
| Main Menu | 1 Byte | 0x00-0x0A |
| Sublevel | 1 Byte | 0x00-0x0A |
| *Reserved (Future Fetch Offset)* | 2 Bytes | 0x0000 |
| Maximum Number of Bytes to Return | 1 Byte | |

#### 5.19.1.4.1.1 Menu File Field

It is possible for a subsystem to have the ability to display different types of user menu information. This byte is used to request a specific set of user menu information.

For example, a menu file of a one might correspond with a thermostat user menu while a menu file of a two might be a menu for a diagnostics device.

Currently only one subsystem level is supported. Therefore the Menu File value shall be 0x01.

#### 5.19.1.4.1.2 Main Menu / Sublevel Fields

The next two bytes specify the main menu and sublevel menu data to return. A 0x00 in the main menu value is a wildcard and indicates that the subsystem should send all Level 1 main menu names.

#### 5.19.1.4.1.3 Maximum Number of Bytes to Return

This byte specifies the maximum number of bytes the requesting subsystem is capable of receiving. This number shall include the size of the CTUM packet payload header including reserved bytes. As such, the maximum value identified inside Maximum Number of Bytes to Return shall be <= 240, 0xF0. If the response is larger than the value in this field, the response shall be broken up into multiple packets.

### 5.19.2GiveUserMenu

The format of the return is dependent on the type of request being made.

#### 5.19.2.1 Prototype:

Data GiveUserMenu(ResponseFunctPtr callback)

#### 5.19.2.2 Parameters

➔ callback A callback function that process the request

#### 5.19.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.19.2.4 Comments

The response data payload will have the User Menu data to be utilized for use on the display device.

##### 5.19.2.4.1 Give User Menu Data Payload

The response back to the client display device is dependent on the type of request being made. The data that is being sent back is the information required for displaying via that client display.

The values for Menu File, Main Menu, Sublevel and Fetch Offset shall be echoes of the values from the request. The Menu Data field will contain the actual User menu information to be displayed on the requesting display device.

**Table 11 – Get User Menu Response Format**

| Field Name | Field Length | Value |
|---|---|---|
| Menu File | 1 Byte | *Copied from request* |
| Main Menu | 1 Byte | *Copied from request* |
| Sublevel | 1 Byte | *Copied from request* |
| *Reserved (Future Fetch Offset)* | 2 Bytes | 0x0000 |
| Maximum Number of Bytes to Return | 1 Byte | *Copied from request* |
| Menu Data | n Bytes | |

## 5.20 Set User Menu Update

A client display device has the ability to update a subsystem with user menu data depending on if the data is able to be modified and has actually been altered. This section discusses the necessary requirements this modification of data is transmitted to the subsystem properly.

This section details the how a CIM application can initiate and process requests and responses.

### 5.20.1 SetUMUpdateData

CIM applications can initiate a transaction to update a User Menu configurable, if applicable.

#### 5.20.1.1 Prototype:

void SetUMUpdateData (Routing, RequestFunctPtr callback, UMUpdateData)

#### 5.20.1.2 Parameters

| | |
|---|---|
| ➔ Routing | Refer to Section 4.2 |
| ➔ callback | A callback function that process the response |
| ➔ ManufactureID | Refer to Section 5.20.1.4.1 |

#### 5.20.1.3 Result

Returns Nothing.

#### 5.20.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

#### 5.20.1.4.1 Set User Menu Update Data Payload

When a value has been modified via the client display device, the data payload will be set up to transmit the modified value to that specific subsystem.

The first three bytes are defined as discussed in the previous section. The updated value is surround by two file security codes to identify a proper modification is being requested. The first file security code is a 0x55 while the ending security code is a 0xAA.

The updated value is based on the type of Main Menu/Sublevel cluster type but would correspond to the appropriate value back to the subsystem that it would be able to evaluate that a proper modification has occurred.

**Table 12 – Update User Menu Request Format**

| Field Name | Field Length | Value |
|---|---|---|
| Menu File | 1 Byte | |
| Main Menu | 1 Byte | |
| Sublevel | 1 Byte | |
| File Security Code | 1 Byte | 0x55 |
| Updated Value | 2 Bytes | |
| File Security Code | 1 Byte | 0xAA |

### 5.20.2 ConfirmUMUpdateData

The format of the set User Menu Update data response in return to the message sent to a CIM application is defined in this section.

#### 5.20.2.1 Prototype:

Data ConfirmUMUpdateData(ResponseFunctPtr callback)

#### 5.20.2.2 Parameters

➔ callback A callback function that process the request

#### 5.20.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.20.2.4 Comments

The response data payload will have a data payload to assist in the confirmation that the update has been successfully made.

**Table 13 – Update User Menu Response Format**

| Field Name | Payload | |
|---|---|---|
| **Field Length** | 8 Bytes | |
| **Value** | Update Data | ACK/NAK |

The response back to the client display device is a sign that the data has been written successfully so that the client display interface can provide the proper feedback to highlight that the update was successful. The response echoes back the update with an appended ACK (0x06) at the end of those bytes to reflect a PASS condition. A NAK (0x15) will inform the requester that the update failed.

## 5.21 Set Factory Shared Data to Application

This message initializes a device's Shared Data at the time of manufacturing. This is not the same as the shared data transmission that occurs at the installed site. The data payload in this message would contain the Application Shared Data.

This section details the how a CIM application can initiate and process requests and responses.

### 5.21.1 SetFactoryAppSharedData

CIM applications can initiate a set transaction to transmit new factory application shared data to modify a subsystem, if applicable.

#### 5.21.1.1 Prototype:

void SetFactoryAppSharedData(Routing, RequestFunctPtr callback, FactoryAppSD)

#### 5.21.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ FactoryAppSD — Refer to Section 5.21.1.4.1

#### 5.21.1.3 Result

Returns Nothing.

#### 5.21.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

##### 5.21.1.4.1 Factory Shared Data Payload

The first byte in the data payload would be the Shared Data Length. The maximum Application Shared Data size is 200 bytes.

The next two bytes will be the Control ID which is provided by CT Alliance. This ID is a two byte integer which is transmitted little endian. Following these bytes, the Manufacturer ID will be transmitted which is another two byte CT Alliance provided integer that is little endian. Control ID and Manufacturer ID shall be represented as follows: If company A

provides a circuit board to company B for the use is a company B subsystem, the Control ID would be the ClimateTalk OEM ID for company A and the Manufacturer ID would be company B.

Following these bytes, the application node type will be sent to be used for validation of the data at a later time.

**Table 14 – Set Factory Shared Data to Application Request Format**

| **Field Name** | Shared Data Length | Control ID | Manufacturer ID | App Node Type | Application Data |
|---|---|---|---|---|---|
| **Field Length** | 1 Byte | 2 Bytes | 2 Bytes | 1 Byte | 0-194 Bytes |
| **Value** | 6-200 | Assigned by ClimateTalk Alliance | Assigned by ClimateTalk Alliance | Refer to Section 0 | TBC |

Finally, the application specific data would be sent that would include the unique data to be stored on the network and later retrieved.

### 5.21.2ConfirmFactoryAppSharedData

The format of the Factory Application Shared Data response in return to the SD sent to a CIM application is generic.

#### 5.21.2.1 Prototype:

Data ConfirmFactoryAppSharedData(ResponseFunctPtr callback)

#### 5.21.2.2 Parameters

➔ callback A callback function that process the request

#### 5.21.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.21.2.4 Comments

The response data payload will have one method of properly receiving a response to a request of data.

##### 5.21.2.4.1 Echoed Response

An echoed response would contain a data payload identical to the data payload received.

## 5.22 Get Shared Data from Application

This command requests the current shared data set being used by a device.

### 5.22.1 GetSDfromApp

CIM applications can receive a requesting transaction to transmit a subsystem applications shared data, if applicable.

#### 5.22.1.1 Prototype:

void GetSDfromApp(Routing, RequestFunctPtr callback)

#### 5.22.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response

#### 5.22.1.3 Result

Returns Nothing.

#### 5.22.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

### 5.22.2 GiveAppSD

The format of the application shared data returned in the response is device-specific, and is defined in the format as defined in Section 5.10.1.4.1.

#### 5.22.2.1 Prototype:

Data GiveAppSD(ResponseFunctPtr callback)

#### 5.22.2.2 Parameters

- ➔ callback — A callback function that process the request

#### 5.22.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.22.2.4 Comments

Get Shared Data from Application is sent to recover the shared data being used by a subsystem.

The response follows the same format as specified in Section 5.21.1.4.1

## 5.23 Set Echo Data

An Echo message is sent out to verify whether a subsystem can handle a 240-byte data payload. It also checks whether the communication lines are noise-resistant and can sustain a large data packet without errors.

This section details the how a CIM application can initiate and process requests and responses.

### 5.23.1 SetEchoData

CIM applications can initiate a set transaction to transmit any data to be responded back with the same data to a subsystem, if applicable.

#### 5.23.1.1 Prototype:

void SetEchoData (Routing, RequestFunctPtr callback, EchoData)

#### 5.23.1.2 Parameters

- ➔ Routing — Refer to Section 4.2
- ➔ callback — A callback function that process the response
- ➔ EchoData — Refer to Section 5.23.1.4.1

#### 5.23.1.3 Result

Returns Nothing.

#### 5.23.1.4 Comments

The callback function is called upon receipt after the transmitted request to the CIM application.

#### 5.23.1.4.1 Echo Data Payload

The data to be sent via this message can be of any value or length. This command is used to help test out a subsystems capability of receiving and transmitting back any amount of data the requestor may decide.

### 5.23.2ConfirmEchoData

The format of the data payload should be confirmed to be an echo of the data payload received during the request.

#### 5.23.2.1 Prototype:

Data ConfirmEchoData (ResponseFunctPtr callback)

#### 5.23.2.2 Parameters

➔ callback A callback function that process the request

#### 5.23.2.3 Result

Returns the length and pointer to the data in response to the request. Refer to Section 4.3 for more detailed information.

#### 5.23.2.4 Comments

The response data payload will have one method of properly receiving a response to a request of data.

##### 5.23.2.4.1 Echoed Response

An echoed response would contain a data payload identical to the data payload received.

# 6.0 Control Command Optional Data

This section will go into detail the optional data required to be sent for some control command code, if applicable.

## 6.1 General

Control commands are used for remotely modifying a device function.

The sections below define only the Payload portion of the packet. The first two bytes of the payload, termed "2- Way Command Code", contain the function being controlled. These bytes are loaded high byte first in the data payload.

The remaining bytes in the payload contain the parameters for the control command being invoked. The payload of the Control command message type is shown in Table 15 – Control Command Message Diagram.

**Table 15 – Control Command Message Diagram**

| **Payload** | | |
|---|---|---|
| **1 to 10 bytes** | | |
| 2-Way Command Code | | (1-8 bytes) |
| LSB | MSB | Command parameters |

## 6.2 Control Command Codes Table

Not all command codes are used by every profile. This section shall define which control command is utilized per each profile.

### 6.2.1 HVAC Profile Control Command Codes

The following table is the list of control command codes that is utilized by the HVAC Profile.

**Table 16 – HVAC Control Command Codes**

| Code (Hex) | Code (Decimal) | Description |
|---|---|---|
| 0x01 | 1 | Heat Set Point Temperature Modify |
| 0x02 | 2 | Cool Set Point Temperature Modify |
| 0x03 | 3 | Heat Profile Change |
| 0x04 | 4 | Cool Profile Change |
| 0x05 | 5 | System Switch Modify |
| 0x06 | 6 | Permanent Set Point Temp & Hold Modify |
| 0x08 | 8 | Hold Override |
| 0x0F | 15 | Real Time/Day Override |
| 0x45 | 69 | Restore Factory Defaults |
| 0x50 | 80 | Test Mode |
| 0x51 | 81 | Subsystem Installation Test |
| 0x61 | 97 | Subsystem Busy Status |
| 0x62 | 98 | Dehumidification Demand |
| 0x63 | 99 | Humidification Demand |
| 0x64 | 100 | Heat Demand |
| 0x65 | 101 | Cool Demand |
| 0x66 | 102 | Fan Demand |
| 0x67 | 103 | Back-Up Heat Demand |
| 0x68 | 104 | Defrost Demand |

| Code (Hex) | Code (Decimal) | Description |
|---|---|---|
| 0x69 | 105 | Aux Heat Demand |
| 0xE0 | 224 | Publish Price |

### 6.2.2 Zone Profile Control Command Codes

The following table is the list of control command codes that is utilized by the Zoning Profile.

**Table 17 – Zone Profile Control Command Codes**

| Code (Hex) | Code (Decimal) | Description |
|---|---|---|
| 0x01 | 1 | Heat Set Point Temperature Modify |
| 0x02 | 2 | Cool Set Point Temperature Modify |
| 0x03 | 3 | Heat Profile Change |
| 0x04 | 4 | Cool Profile Change |
| 0x05 | 5 | System Switch Modify |
| 0x06 | 6 | Permanent Set Point Temp & Hold Modify |
| 0x08 | 8 | Hold Override |
| 0x0F | 15 | Real Time/Day Override |
| 0x45 | 69 | Restore Factory Defaults |
| 0x50 | 80 | Test Mode |
| 0x51 | 81 | Subsystem Installation Test |
| 0x52 | 82 | Auto-Pairing Request |
| 0x53 | 83 | Pair Ownership Request |
| 0x60 | 96 | Damper Position Demand |
| 0x61 | 97 | Subsystem Busy Status |
| 0x62 | 98 | Dehumidification Demand |
| 0x63 | 99 | Humidification Demand |
| 0x64 | 100 | Heat Demand |

| Code (Hex) | Code (Decimal) | Description |
|---|---|---|
| 0x65 | 101 | Cool Demand |
| 0x66 | 102 | Fan Demand |
| 0x67 | 103 | Back-Up Heat Demand |
| 0x68 | 104 | Defrost Demand |
| 0x69 | 105 | Aux Heat Demand |
| 0xE0 | 224 | Publish Price |

### 6.2.3 Motor Profile Control Command Codes

The following table is the list of control command codes that is utilized by the Motor Profile.

**Table 18 – Motor Profile Control Command Codes**

| Code (Hex) | Code (Decimal) | Description |
|---|---|---|
| 0x6A | 106 | Set Motor Speed |
| 0x6B | 107 | Set Motor Torque |
| 0x6C | 108 | Set Airflow Demand |
| 0x6D | 109 | Set Control Mode |
| 0x6E | 110 | Set Demand Ramp Rate |
| 0x6F | 111 | Set Motor Direction |
| 0x70 | 112 | Set Motor Torque Percent |
| 0x71 | 113 | Set Motor Position Demand |
| 0x72 | 114 | Set Blower Coefficient 1 |
| 0x73 | 115 | Set Blower Coefficient 2 |
| 0x74 | 116 | Set Blower Coefficient 3 |
| 0x75 | 117 | Set Blower Coefficient 4 |
| 0x76 | 118 | Set Blower Coefficient 5 |

| Code (Hex) | Code (Decimal) | Description |
|---|---|---|
| 0x77 | 119 | Set Blower Identification 0 |
| 0x78 | 120 | Set Blower Identification 1 |
| 0x79 | 121 | Set Blower Identification 2 |
| 0x7A | 122 | Set Blower Identification 3 |
| 0x7B | 123 | Set Blower Identification 4 |
| 0x7C | 124 | Set Blower Identification 5 |
| 0x7F | 127 | Set Speed Limit |
| 0x80 | 128 | Set Torque Limit |
| 0x81 | 129 | Set Airflow Limit |
| 0x82 | 130 | Set Power Output Limit |
| 0x83 | 131 | Set Device Temperature Limit |
| 0x85 | 133 | STOP Motor by Braking |
| 0x86 | 134 | RUN/STOP Motor |
| 0x88 | 136 | Set Demand Ramp Time |
| 0x89 | 137 | Set Inducer Ramp Rate |
| 0x8A | 138 | Set Blower Coefficient 6 |
| 0x8B | 139 | Set Blower Coefficient 7 |
| 0x8C | 140 | Set Blower Coefficient 8 |
| 0x8D | 141 | Set Blower Coefficient 9 |
| 0x8E | 142 | Set Blower Coefficient 10 |

### 6.2.4 Water Heater Control Command Codes

Refer to Water Heater Application Profile for the list of control commands codes utilized by a Water Heater Application.

### 6.2.5 Generic Profile Control Command Codes

There are no mandatory or optional requirements for a Generic Node to implement a control command.

## 6.3 HVAC Control Command Refresh Timers

Certain control commands require refreshing after a specified amount of time depending. They include a refresh timer byte in the command giving the amount of time before the command is considered invalid. This time amount is used to ensure communications are valid and that runaway conditions are avoided.

If a command is not refreshed within the time period specified, then the subsystem will return to its state prior to the original request.

Table 19 – Control Command Refresh Timers Bit-By-Bit Breakdown defines the refresh timer byte format. Any value is acceptable, giving a maximum refresh timer of 956.25 seconds, or 15.9375 minutes.

**Table 19 – Control Command Refresh Timers Bit-By-Bit Breakdown**

| Refresh Timer (Bits) | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Minutes | | | | Seconds | | | |

Bits 7 – 4 hold the minute value while bits 3 – 0 define the seconds. The seconds shall be defined in 16$^{th}$ divisions of a minute. Each 1/16$^{th}$ is equaled to 3.75 seconds of time.

## 6.4 Heat Set Point Temperature Modify – Command Code 0x01

This command is used to modify the heat set point temperature.

**Table 20 – 2-Way Command Word - Heat Set Point Temperature Modify**

| **Field Name** | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x01 | 0x00 | Set Point Temperature |

Verification that the heat set point has been changed is done with a Get Status request. The data payload in the status reply shall reflect whether or not the heat set point temperature has actually been modified.

## 6.5 Cool Set Point Temperature Modify – Command Code 0x02

This command is used to modify the cool set point temperature.

**Table 21 – 2-Way Command Word - Cool Set Point Temperature Modify**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x02 | 0x00 | Set Point Temperature |

Verification that the cool set point has been changed is done with a Get Status request. The data payload in the status reply shall reflect whether or not the cool set point temperature has actually been modified.

## 6.6 Heat Profile Change – Command Code 0x03

This command is used to modify the heat program profile.

**Table 22 – Command Word – Heat Profile Change**

| Field Name | Command Code | Command Data | | | |
|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | 2 Bits (0-1) | 2 Bits (2-3) | 4 Bits (4-7) | n-3 Bytes |
| **Value** | 0x03 0x00 | Profile Interval Type | Profile Mode Type | 0x00 | Profile Data |

Table 23 – Profile Interval Type defines the Profile Interval Type Bits.

**Table 23 – Profile Interval Type**

| Profile Step Type | Value |
|---|---|
| 4 Step | 0 |
| 2 Step | 1 |
| Non-Programmable | 2 |
| Spare | 3 |

Table 24 – Profile Mode Type defines the Profile Mode Type bits.

**Table 24 – Profile Mode Type**

| Profile Mode Type | Value Code |
|---|---|
| Non-Programmable | 0 |
| 5-1-1 | 1 |
| 7 | 2 |
| 5-2 | 3 |

The response to a Get Configuration request shows which profiles may be modified. For example, if only a 7-Day – 4 Interval Program is shown, that is the only program which could be modified.

The format of the Get Configuration response payload corresponds to the profile data sent in the Heat Profile Change command. This allows for many types of profile data to be used and the burden on the person modifying the stat. If this command is sent with no profile data, then the default schedule shall be assumed.

If the profile type and profile mode are set to "non-programmable", then the profile data is the heat set point. If no profile data is included with the command, the default heat set point shall be assumed.

Verification that the heat profile has been changed is done with a Get Configuration request. The data payload in the reply shall reflect whether or not the heat profile has actually been modified.

## 6.7 Cool Profile Change – Command Code 0x04

This command is used to modify the cool program profile.

**Table 25 – Command Word – Cool Profile Change**

| Field Name | Command Code | Command Data | | | |
|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | 2 Bits (0-1) | 2 Bits (2-3) | 4 Bits (4-7) | n-3 Bytes |
| **Value** | 0x04 0x00 | Profile Interval Type | Profile Mode Type | 0x00 | Profile Data |

Refer to the tables for Heat Profile Change for the definition of the Profile Interval Type and Profile Mode Type fields.

The response to a Get Configuration request shows which profiles may be modified. For example, if only a 7-Day – 4 Interval Program is shown, that is the only program which could be modified.

The format of the Get Configuration response payload corresponds to the profile data sent in the Cool Profile Change command. This allows for many types of profile data to be used and the burden on the person modifying the stat. If this command is sent with no profile data, then the default schedule shall be assumed.

If the profile type and profile mode are set to "non-programmable", then the profile data is the heat set point. If no profile data is included with the command, the default heat set point shall be assumed.

Verification that the cool profile has been changed is done with a Get Configuration request. The data payload in the reply shall reflect whether or not the cool profile has actually been modified.

## 6.8 System Switch Modify – Command Code 0x05

This command modifies the current system switch state to either OFF, HEAT, COOL, AUTO, or EMER. The payload contains the new system switch state to use.

**Table 26 – Command Word – System Switch Modify**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x05 | 0x00 | System Switch Code |

Table 27 – System Switch Code defines the system switch state codes.

**Table 27 – System Switch Code**

| Switch Code Description | Value Code |
|---|---|
| Off | 0 |
| Cool State | 1 |
| Auto State | 2 |
| Heat State | 3 |
| Back-Up Heat State | 4 |

To confirm that the system switch change has occurred, the requesting device should issue a Get Status request. The status response will indicate the current system switch setting.

If any of the system switch commands are not active, an appropriate response will be returned to the commanding device. If five attempts have been made, the requester shall stop requesting this modification due to the system being off and not that a bad packet has been received.

### 6.9 Permanent Set Point Temperature and Hold Modify – Command Code 0x06

This command modifies the current set point temperature and places the system in a HOLD state. The payload contains the new hold temperature.

**Table 28 – Command Word – Permanent Set Point Temperature and Hold Modify**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x06 | 0x00 | Hold Temp |

To confirm that the set point temperature has been modified and the system placed in HOLD, the requesting device should issue a Get Status request. The status response will indicate the current settings.
If the system is OFF and this command is sent to the system, the system will acknowledge the request but ignore the command until the system switch has been activated.

### 6.10 Fan Key Selection – Command Code 0x07

This command is used to modify current fan operation of a thermostat and place the system in the different states fan operation is possible for that thermostat.

**Table 29 – Command Word – Fan Key Selection**

| Field Name | Command Code | | Command Data | |
|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 1 Optional Bytes |
| **Value** | 0x07 | 0x00 | Fan Control | Fan Definition |

The following table is used to command the requesting device to go to the appropriate fan control operation mode that would be set in the fan control and optional fan definition bytes:

**Table 30 – Fan Control Truth Table**

| Description | Fan Control | Fan Definition |
|---|---|---|
| Auto Fan | 0 | - |
| Manual Fan (ON) | 1 | % Demand |

To confirm that the fan has been modified and the system modified, the requesting device should issue a Get Status request. The status response will indicate the current settings.

## 6.11 Hold Override – Command Code 0x08

This command is used to command the thermostat in Hold Override Mode. The payload is a one-byte word that enables or disables the hold override. A value of "0x01" means to put the thermostat into Hold mode and "0x00" disables Hold.

**Table 31 – Command Word – Hold Override**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x08 | 0x00 | Enable (1)/ Disable (0) |

To confirm that the system switch change has occurred, the requesting device should issue a Get Status request. The status response will indicate the current system switch setting.

## 6.12 Beeper Enable – Command Code 0x09

This command is used to modify the configuration of the thermostat remotely to change the beeper audio indicator of the thermostat. The payload is a one-byte word that enables or disables the beeper. A value of "0x01" means to put enable the beeper and "0x00" disables the beeper.

**Table 32 – Command Word – Beeper Enable**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x09 | 0x00 | Enable (1)/ Disable (0) |

To confirm that the system switch change has occurred, the requesting device should issue a Get Config request. The configuration response will indicate the current system switch setting.

## 6.13 Fahrenheit/Celsius Display – Command Code 0x0C

This command is used to modify the configuration of the thermostat remotely to change the beeper audio indicator of the thermostat. The payload is a one-byte word that modifies the units of temperature that a subsystem would use for display. A value of "0x01" means to display temperature units in Fahrenheit and "0x00" means to display units in Celsius.

**Table 33 – Command Word – Fahrenheit/Celsius Display**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x0C | 0x00 | ºF (1) / ºC (0) |

To confirm that the system switch change has occurred, the requesting device should issue a Get Config request. The configuration response will indicate the current system switch setting.

## 6.14 Comfort Recovery (EMR) Modify – Command Code 0x0E

This command allows for remote modifications of the ability to change the enabling/disabling of the Comfort Recovery (EMR) capabilities of the thermostat.

**Table 34 – Command Word – Comfort Recovery Selection**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x0E | 0x00 | Configuration Data |

The following table is used to command the requesting device to go to the appropriate fan control operation mode that would be set in the fan control and optional fan definition bytes:

**Table 35 – Comfort Recovery Config Data**

| Configuration Bits | | |
|---|---|---|
| 7 | 6-1 | 0 |
| Capable | Spares | Enabled (1) / Disabled (0) |

The capable configuration bit seven shall have a higher priority and needs to be set before determining if this capability is enabled if set in bit zero.

To confirm that the system switch change has occurred, the requesting device should issue a Get Config request. The configuration response will indicate the current system switch setting.

## 6.15 Real Time/Day Override – Command Code 0x0F

This command sets the real time clock. The payload contains the new time and date.

**Table 36 – Command Word – Real Time/Day Override**

| Field Name | Command Code | | Command Data | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 6 Bytes | | | | | |
| **Value** | 0x0F | 0x00 | Year | Month | Date | Day | Hour | Min |

The year is the value in the field plus 2000. For example, the year 2000 is transmitted as 0x00, the year 2009 is transmitted as 0x09.

For the month field, 0x00 = January, 0x01 = February, etc. through 0x0B for December.

Date has a range from one to thirty-one.

For the day field, 0x00 = Monday, 0x01 = Tuesday, etc. through 0x06 for Sunday.
The hour field uses 24 hour time. Midnight is 0x00 and noon is 0x0C (12). 3 in the afternoon would be 0x0F (15).

Minutes have a range from zero to fifty-nine.

To confirm that the time and date have been set, the requesting device should issue a Get Status request. The status response will indicate the current real time and date.

## 6.16 Change Filter Time Remaining – Command Code 0x14

This command is used to command the filter timer remaining to a new amount of time to start decrementing when fan is in operation. In the command data, a byte is available to reset (1) back to the predetermined filter remaining time and then two optional bytes that contain the number of hours left remaining prior to the change filter alarm to be initiated.

**Table 37 – Command Word – Change Filter Time Remaining**

| Field Name | Command Code | | Command Data | |
|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 2 Optional Bytes |
| **Value** | 0x14 | 0x00 | Reset | Time (Hours) |

The number of hours will correspond to the number of hours which will then be used for whatever units of display to the end user. If a reset is not required, the reset byte would be sent with a zero in the data payload.

Verification that the thermostat has actually been changed has to be done by requesting a status request. The data payload in the status response shall reflect whether or not the thermostat has actually been modified.

To add clarity, the following are examples of the packet received from the network:

(1) Reset the timer without changing the total time preset at installation:

**Table 38 – Reset Filter Time Remaining Example**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x14 | 0x00 | 0x01 |

(2) Reset the timer and change the total time remaining to 18 hours:

**Table 39 – Reset and Change Time for Filter Time Remaining Example**

| Field Name | Command Code | | Command Data | | |
|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 2 Optional Bytes | |
| **Value** | 0x14 | 0x00 | 0x01 | 0x12 | 0x00 |

(3) Change only the total time remaining to 255 hours:

**Table 40 – Change Time for Filter Time Remaining Example**

| Field Name | Command Code | | Command Data | | |
|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 2 Optional Bytes | |
| **Value** | 0x14 | 0x00 | 0x00 | 0xFF | 0x00 |

## 6.17 Vacation Mode – Command Code 0x15

This command is used to modify the vacation mode state of the device.

The payload is a one-byte word that enables or disables vacation mode. A value of "0x01" means to put the thermostat into vacation mode and "0x00" disables vacation mode.

**Table 41 – Command Word – Vacation Mode**

| Field Name | Command Code | | Command Data | | |
|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 2 Optional Bytes | |
| **Value** | 0x15 | 0x00 | Enable (1)/ Disable (0) | Vacation Heat Set point ($^{o}$F) | Vacation Cool Set point ($^{o}$F) |

To confirm that the vacation mode change has occurred, the requesting device should issue a Get Status request. The status response will indicate the current vacation setting.

## 6.18 High Alarm Limit Change – Command Code 0x16

This command is used to modify the high alarm limit. The payload is the new high alarm limit temperature.

**Table 42 – Command Word – High Alarm Limit**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x16 | 0x00 | Temp |

## 6.19 Low Alarm Limit Change – Command Code 0x17

This command is used to modify the low alarm limit. The payload is the new low alarm limit temperature.

**Table 43 – Command Word – Low Alarm Limit**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x17 | 0x00 | Temp |

## 6.20 High Outdoor Alarm Limit Change – Command Code 0x18

This command is used to modify the outdoor high alarm limit. The payload is the new outdoor high alarm limit temperature.

**Table 44 – Command Word – Outdoor High Alarm Limit**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x18 | 0x00 | Temp |

## 6.21 Low Outdoor Alarm Limit Change – Command Code 0x19

This command is used to modify the outdoor low alarm limit. The payload is the new outdoor low alarm limit temperature.

**Table 45 – Command Word – Low Outdoor Alarm Limit**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x19 | 0x00 | Temp |

## 6.22 Temp Display Adj Factor Change – Command Code 0x1A

This command is used to command the thermostat to a different temperature offset that could be configured manually or via this command.

**Table 46 – Command Word – Set Point and Hold Modification**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x1A | 0x00 | Temp Offset |

The temperature offset sent will be two's complemented byte that contains the negative or positive value to be used to offset the currently displayed temperature.

## 6.23 Clear Compressor Run Time – Command Code 0x2D

This command resets the thermostat compressor run time and has no optional command data.

**Table 47 – Command Word – Clear Compressor Run Time**

| Field Name | Command Code | |
|---|---|---|
| **Field Length** | 2 Bytes | |
| **Value** | 0x2D | 0x00 |

## 6.24 Reset Control – Command Code 0x31

The Reset Control control command resets the device's application and has no optional command data. The device shall respond to the command before resetting itself.

**Table 48 – Command Word – Reset Microcontroller**

| Field Name | Command Code | |
|---|---|---|
| **Field Length** | 2 Bytes | |
| **Value** | 0x31 | 0x00 |

## 6.25 Compressor Lockout – Command Code 0x33

This command is used to command the thermostat to lockout the compressor for a period of time.

The payload is a one-byte word that enables or disables this feature. A value of "0x01" means to enable compressor lockout while "0x00" disables this feature

**Table 49 – Command Word – Compressor Lockout**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x15 | 0x00 | Enable (1)/ Disable (0) |

To confirm that the change has occurred, the requesting device should issue a Get Status request. The status response will indicate the current system switch setting.

## 6.26 Hold Release – Command Code 0x3D

This command removes any holds present in the system due to power failure or any other cause and has no optional command data.

**Table 50 – Command Word – Hold Release**

| Field Name | Command Code | |
|---|---|---|
| **Field Length** | 2 Bytes | |
| **Value** | 0x3D | 0x00 |

## 6.27 Program Interval Type Modification – Command Code 0x3E

This command is used to modify current profile interval type to either 4-Step, 2-Step or Non-Programmable. The payload is the new profile type to use.

**Table 51 – Command Word – Program Interval Type Modification**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x3E | 0x00 | See Table 23 – Profile Interval Type |

To verify that the profile type has been changed, the requesting device should issue a Get Configuration request. The configuration response will indicate the current profile type.

## 6.28 Communications Receiver On/Off – Command Code 0x3F

This command turns off the communications receiver to reduce power consumption or for any other reason.

**Table 52 – Command Word – Comm Receiver On/Off**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x3F | 0x00 | On/Off |

The payload consists of a 0x01 (On) or 0x00 (Off).

## 6.29 Force Phone Number Display – Command Code 0x40

On receiving this command, the thermostat or other display device will display the phone number stored in the system.

**Table 53 – Command Word – Force Phone Number Display**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x40 | 0x00 | Enable/Disable |

The payload consists of a 0x01 (Display Phone Number) or 0x00 (Stop Displaying Phone Number).

## 6.30 Restore Factory Defaults – Command Code 0x45

This command is used to command devices back to a factory default state of operation. The payload contains the type of restoration to perform on the subsystem.

**Table 54 – Command Word – Restore Factory Defaults**

| Field Name | Command Code | Command Data |
|---|---|---|
| **Field Length** | 2 Bytes | 1 Byte |

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Value** | 0x45 | 0x00 | Restore Type |

The following table is used to command the requesting device to go to the appropriate operational mode.

**Table 55 – Restore Types**

| Restore Type | Value Code |
|---|---|
| Return to Factory Defaults | 0 |

## 6.31 Custom Message Area Display Data – Command Code 0x46

This command is used to transmit and requesting generic message data from a displaying device. Following the command code will be the necessary information payload to identify the generic message configuring capabilities.

**Table 56 – Command Word – Comfort Recovery Selection**

| Field Name | Command Code | | Command Data | | | |
|---|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 1 Byte | 1 Optional Byte | TBC Optional Bytes |
| **Value** | 0x46 | 0x00 | Configuration | Active Array | Modified Index | Modified ASCII |

The first byte after the command code is the **message area identifier (id)** that would be used to inform which area that is being modified.

Following that would be the display cycle time for that message area that would be used to cycle from one message to the next. The display cycle time is per message area code and if a zero is sent then whatever was stored in the application previously or the default application time defined will be used.

The following table is the bit-by-bit breakdown of the custom message configuration byte:

**Table 57 – Custom Message Area Config Data**

| Configuration Bits | | |
|---|---|---|
| 7-5 | 4 | 3-0 |
| Area ID Code | Spare | Seconds (½ Sec Increments) |

The next byte is the active array byte that identifies which index is active or not. A maximum of 8 different messages can be stored per message area ID. More than 8 would simply require providing another ID code to the same area. Fewer indexes could be utilized as well. The normal API responses would identify the latter condition.

Following those bytes would be the modified index that would be used to either request what is stored in memory (see next subsection) or modify the data to be displayed and stored. Bits zero thru two within the byte allows for the indication of which array to be modified or requested to the device.

If the byte is used as modifying the data in the array, bits seven and six will be used as formatting bits of the modified ASCII data to be used as the message.

**Table 58 – Modified Index Byte Definition**

| Configuration Bits | | | |
|---|---|---|---|
| 7 | 6 | 5-3 | 2-0 |
| Blink | Reverse | Spares | Array Index |

After this modified index byte, optional bytes following would be the actual ASCII data to be used for display. A NULL (0x00) must be appended to this data. If the display device has the capability of displaying multiple lines, each line would be appended a NULL (0x00) to identify the next line.

For example, if the modified ASCII had the following <H><NULL><i><NULL><!><NULL>, then the following would be displayed:

**Table 59 – Three-Line Custom Message Display Example**

H
I
!

### 6.31.1 Custom Message Area Request

It is also possible to use this command to request the displaying device what it has stored in memory for the message areas by index. A packet with the missing optional modified ASCII data bytes would identify that a request is being made to respond back w/ the data stored in memory. In this case, the formatting bits within the modified index will be ignored as well.

The following packet below is an example of a request:

**Table 60 – Custom Message Request Example**

| Field Name | Command Code | | Command Data | | | |
|---|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 1 Byte | 1 Optional Byte | TBC Optional Bytes |
| **Value** | 0x46 | 0x00 | Configuration | Active Array | Modified Index | Modified ASCII |
| | | | 0x20 | 0x00 | 0x00 | - |

In the previous example, the request is being made of the display data located in Message Area ID one at index 0. Below is the appropriate response packet that would follow from this sample request with ASCII Data stored in memory.

**Table 61 – Custom Message Response Example**

| Field Name | Command Code | | Command Data | | | |
|---|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 1 Byte | 1 Optional Byte | TBC Optional Bytes |
| **Value** | 0x46 | 0x00 | Configuration | Active Array | Modified Index | Modified ASCII |
| | | | 0x20 | 0x00 | 0xC0 | "EXAMPLE"<NULL> |

Note that in Table 61 that the response properly reflects the addition to the interval display time and the proper formatted allocated for that indexed array.

## 6.32 Set Point Temp and Temporary Hold – Command Code 0x47

This command is used to modify current set temperature to the thermostat and place the system in an interval HOLD status.

**Table 62 – Command Word – Set Point and Hold Modification**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x47 | 0x00 | Hold Temp |

To confirm that the set point temperature has been modified and the system placed in HOLD, the requesting device should issue a Get Status request. The status response will indicate the current settings.

The range of the hold temperature maybe different in thermostats and even though the packet was accepted does not mean that the thermostat actually implemented this command until this validation has been done via the status transaction.

For example, If the system is OFF and this command is sent to the thermostat, the thermostat will accept the request but ignore the command until the system switch has been activated.

## 6.33 Continuous Display Light – Command Code 0x48

This command is used to modify the continuous display light configuration of the thermostat remotely.

The payload is a one-byte word that enables or disables this feature. A value of "0x01" means to enable compressor lockout while "0x00" disables this feature

**Table 63 – Command Word – Continuous Display Light**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x48 | 0x00 | Enable (1)/ Disable (0) |

To confirm that the change has occurred, the requesting device should issue a Get Config request. The configuration response will indicate the current system switch setting.

## 6.34 Advance Real Time/Day Override – Command Code 0x4E

This command is used to command the thermostat to a new day and time. Following the command code will be the necessary information payload to identify the generic message configuring capabilities.

**Table 64 – Command Word – Advance Real Time/Day Override**

| Field Name | Command Code | | Command Data | | | |
|---|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 1 Byte | 4 Optional Bytes | 4 Optional Bytes |
| **Value** | 0x4E | 0x00 | Control | GMT Offset | Date (Days) | Time (Seconds) |

After the control command code bytes, a real time/day control byte will be sent that would contain more information in the display of time/day. The table below illustrates this bit breakdown to be sent to a device for determining time modification lock and daylight savings time (DST).

Bit 7 is used to modify the time lockout function where the device receiving an enable (1) would not allow the user the ability to modify the time and is done remotely. A disable (0) byte received would unlock and allow the user to do their own time modifications if so desired. The last one modifying the real time/day will be the time change that is utilized.

**Table 65 – Advance Real Time/Day Control Byte**

| Configuration Bits | | |
|---|---|---|
| 7 | 6-1 | 0 |
| Lock | Spares | DST |

Also, in this control byte, bit zero allows the modification of the Daylight Savings Time (DST) capability. DST will be sent an enable (1) or disable (0) bit to put the device into this mode of daylight savings operation.

After the control byte, the GMT offset byte will be sent to make the determination what time zone the device is operating within. The GMT offset will be a signed byte representing the actual offset in hours multiplied by four. The offset is scaled by four to make the determination of quarter hour (15 Minutes) offset for the GMT.

For example, if the GMT offset was 5:45 hours the following calculations would occur:

Sender: 5:45 = 5.75 * 4 = 23 → 00010111
Receiver: 00010111 = 23 → 23/4 = 5.75 = 5:45

Another example, if the GMT offset was -3:30 hours the following calculations would occur:

Sender: -3:30 = -3.5 * 4 = -14 → 11110010
Receiver: 11110010 = -14 → -14/4 = -3.5 = -3:30

From the calculated value of GMT offset the actual time zone can be determined via a lookup table. Table 66 is an example of a GMT offset value sent and its corresponding time zone to be used for determination.

**Table 66 – Time Zone Example GMT Offset Table**

| Time Zone | GMT Offset |
|---|---|
| Hawaiian Standard Time | -10:00 |
| Alaskan Standard Time | -9:00 |

| | |
|---|---|
| Pacific Standard Time, Yukon | -8:00 |
| Mountain Standard Time | -7:00 |
| Central Standard Time | -6:00 |
| Eastern Standard Time | -5:00 |
| Atlantic Standard Time | -4:00 |
| Newfoundland | -3:30 |

The next four optional bytes contains the number of days from the start date of Jan 1, 2000 to compute the calendar day that control will be changed to remotely. The following example demonstrates how the date of 9/10/2009 would be packed with 3,540 days into these four bytes.

| | | | |
|---|---|---|---|
| 09/10/2009 | - 1/1/2009 | = 252 | days |
| | 2008 | = 366 | days |
| | 2007 | = 365 | days |
| | 2006 | = 365 | days |
| | 2005 | = 365 | days |
| | 2004 | = 366 | days |
| | 2003 | = 365 | days |
| | 2002 | = 365 | days |
| | 2001 | = 365 | days |
| | 2000 | = 366 | days |
| | Total | = **3540** | days |

Following the date, the last four optional bytes of the data payload will be the time in seconds from midnight to be calculated to determine the actual new time being changed to remotely.

| | | | |
|---|---|---|---|
| 01:04:19 UTC | 1 hour | = 3600 | seconds |
| | 4 minutes | = 240 | seconds |
| | 19 seconds | = 19 | seconds |
| | Total | = **3859** | seconds |

The above example demonstrates how the time of 1:04:19 would be packed with 3,859 seconds into these two bytes.

## 6.35 Keypad Lockout – Command Code 0x4F

This command is used to lock the keypad of the device either full or partial. .

The first byte in the command data enables or disables this feature. A value of "0x01" is to enable keypad lockout while "0x00" disables this feature. After that there will two bytes that will contain the code to lock the thermostat.

**Table 67 – Command Word – Keypad Lockout**

| Field Name | Command Code | | Command Data | | |
|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 1 Byte | 2 Bytes |
| **Value** | 0x48 | 0x00 | Enable (1)/ Disable (0) | Partial (0)/ Full (1) | Password |

When the thermostat is commanded to unlock the keypad, the correct code must be sent to the thermostat in order for the thermostat to unlock. The allowable code that is accepted by the thermostat is between 1 and 65535. The master code to unlock no matter what the code is defined will be 00000.

To confirm that the change has occurred, the requesting device should issue a Get Status request. The status response will indicate the current system switch setting.

## 6.36 Test Mode – Command Code 0x50

Subsystems may be configured to perform a test mode to speed up timers or help test it during manufacturing.

**Table 68 – Command Word - Control/Manufacturer Test Mode**

| Field Name | Command Code | | Command Data | |
|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes | 1 Byte |
| **Value** | 0x50 | 0x00 | Manufacturer ID | Test Code |

Table 70 – Command Word – System Installation Test defines the different types of test situations that each subsystem may or may not be required to perform.

**Table 69 – Test Mode Code Reference Table**

| Test Codes | |
|---|---|
| Test Description | Test Code |
| Manufacturing Test Mode Enable | 1 |
| Control Test Mode Enable | 2 |

| Test Codes | |
|---|---|
| Reliability Product Test Mode Enable | 3 |
| Reliability System Test Mode Enable | 4 |
| Exit Test Mode | 255 |

## 6.37 Subsystem Installation Test – Command Code 0x51

This command is used to command a subsystem to perform an installation operational test.

**Table 70 – Command Word – System Installation Test**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x51 | 0x00 | Start (1)/Stop (0) |

The payload is a single byte value, where 0x01 means to start the test and 0x00 means to stop the test.

## 6.38 Set Point Temp and Temporary Hold – Command Code 0x53

This command is used to modify current set temperature to the thermostat and place the system in predetermined time amount hold status. This is also known as timed hold status in a thermostat that would keep this new temperature until the timed hold status has expired.

The command data first byte is the hold temperature that the thermostat will be commanded to during the interval of time to be in that hold state.

**Table 71 – Command Word – Set Point and Temp Hold Modification**

| Field Name | Command Code | | Command Data | |
|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 2 Bytes |
| **Value** | 0x53 | 0x00 | Hold Temp | Variable Time (Min) |

The bytes after the hold temperature will be the amount of time that the thermostat will observe before returning to the normal state after an timed hold has been issued locally.

However, verification that the thermostat has actually been changed has to be done by requesting a Get Status. The status response shall reflect whether or not the thermostat has actually been modified.

The range of the hold temperature maybe different in thermostats and even though the packet was accepted does not mean that the thermostat actually implemented this command until this validation has been done via the status transaction.

For example, if the system is OFF and this command is sent to the thermostat, the thermostat will accept the request but ignore the command until the system switch has been activated.

## 6.39 Comfort Mode Modification – Command Code 0x55

This command is used to modify the thermostat's ability to go to second stage immediately or wait when a user is hitting the change in set point for all modes in a thermostat.

The payload is a one-byte word that enables or disables this feature. A value of "0x01" means to enable while "0x00" disables this feature

**Table 72 – Command Word – Continuous Display Light**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x55 | 0x00 | Enable (1)/ Disable (0) |

To confirm that the change has occurred, the requesting device should issue a Get Config request. The configuration response will indicate the current system switch setting.

## 6.40 Limited Heat and Cool Range – Command Code 0x56

This command allows for remote modifications of the highest limit to all the system modes ranges.

Following the control command code will be integer temperature value to be used as the highest level of cool that the thermostat can be configured to by an operator. After the cool mode byte, will be an integer temperature value to be used for the highest level of cool mode.

**Table 73 – Command Word – Limited Heat and Cool Range**

| Field Name | Command Code | | Command Data | |
|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 1Byte |
| **Value** | 0x56 | 0x00 | Min Cool Limited Temp | Max Heat Limited Temp |

However, verification that the thermostat has actually been changed has to be done by requesting a Get Config. The configuration response shall reflect whether or not the subsystem has actually been modified.

## 6.41 Auto-Pairing Request – Command Code 0x57

This command is used to initiate subsystems to perform the auto-pairing process of identifying which subsystem belongs to the appropriate zone. This command also is used to communicate if a zone is requested to be paired with the appropriate zone active in the process.

**Table 74 – Command Word – System Installation Test**

| Field Name | Command Code | | Command Data | |
|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 1 Byte |
| **Value** | 0x57 | 0x00 | Status Code | Action Code |

The following table identifies the different status code and definition to be used during auto-pairing being initiated:

**Table 75 – Status Code Reference Table**

| Status Codes | |
|---|---|
| Status Description | Code |
| No, this is not my zone | 0x00 (0) |
| Yes, this is my zone | 0x01 (1) |

The following table identifies the different action code and definition to be used during auto-pairing being initiated:

**Table 76 – Action Code Reference Table**

| Action Codes | |
|---|---|
| Action Description | Code |
| No Action | 0x00 (0) |
| Start Auto Pairing Process | 0x01 (1) |
| Cancel Auto Pairing Process | 0x02 (2) |
| Auto Process Complete | 0xFF (255) |

## 6.42 Pairing Ownership Request – Command Code 0x58

This command is used to command a subsystem to a specific zone identifier.

**Table 77 – Command Word – System Installation Test**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x58 | 0x00 | Zone Identifier |

The zone identifier can be whatever the requesting subsystem defines but is required to be stored at all times until the next request is made, if applicable.

## 6.43 Reversing Valve Configuration – Command Code 0x59

This command is used to configure the reversing valve operation of a subsystem.

**Table 78 – Command Word – System Installation Test**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x59 | 0x00 | O Mode (0) / B Mode (1) |

If the mode of operation is set to a zero, the reversing valve mode of operation will be O Mode while a one will be B mode.

## 6.44 DEHUM/HUM Configuration – Command Code 0x5A

This command is used to modify the dehumidification and humidification capabilities of a subsystem.

**Table 79 – Command Word – System Installation Test**

| Field Name | Command Code | | Command Data | |
|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes | |
| **Value** | 0x5A | 0x00 | DEHUM Config | HUM Config |

The following table is used to define the different configurations for both dehumidification and humidification of a subsystem:

**Table 80 – DEHUM/HUM Configuration Bit-By-Bit Reference Table**

| DEHUM/HUM Configuration Byte | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Independent Enabled | Spares | | | | | | Mode Enabled |

When a bit is set, then the function is enabled. If a device has only one of the two capabilities, the other would just be ignored but the packet would be accepted. Validation of the actual command being processed shall require a separate Get Configuration request from that device.

## 6.45 Change UV Light Maintenance Timer – Command Code 0x5B

This command is used to command the UV Light timer remaining to a new amount of time to start decrementing when fan is in operation. In the command data, a byte is available to reset (1) back to the predetermined remaining time and then two optional bytes that contain the number of hours left remaining prior to the change filter alarm to be initiated.

**Table 81 – Command Word – Change UV Light Time Remaining**

| Field Name | Command Code | | Command Data | |
|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 2 Optional Bytes |
| **Value** | 0x5B | 0x00 | Reset | Time (Hours) |

The number of hours will correspond to the number of hours which will then be used for whatever units of display to the end user. If a reset is not required, the reset byte would be sent with a zero in the data payload.

Verification that the thermostat has actually been changed has to be done by requesting a status request. The data payload in the status response shall reflect whether or not the thermostat has actually been modified.
To add clarity, the following are examples of the packet received from the network:

(1) Reset the timer without changing the total time preset at installation:

**Table 82 – Reset UV Light Time Remaining Example**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x5B | 0x00 | 0x01 |

(2) Reset the timer and change the total time remaining to 18 hours:

**Table 83 – Reset and Change Time for UV Light Time Remaining Example**

| Field Name | Command Code | | Command Data | | |
|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 2 Optional Bytes | |
| **Value** | 0x5B | 0x00 | 0x01 | 0x12 | 0x00 |

(3) Change only the total time remaining to 255 hours:

**Table 84 – Change Time for UV Light Time Remaining Example**

| Field Name | Command Code | | Command Data | | |
|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 2 Optional Bytes | |
| **Value** | 0x5B | 0x00 | 0x00 | 0xFF | 0x00 |

## 6.46 Change Hum Pad Maintenance Timer – Command Code 0x5C

This command is used to command the Humidifier Pad timer remaining to a new amount of time to start decrementing when fan is in operation. In the command data, a byte is available to reset (1) back to the predetermined remaining time and then two optional bytes that contain the number of hours left remaining prior to the change filter alarm to be initiated.

**Table 85 – Command Word – Change Hum Pad Time Remaining**

| Field Name | Command Code | | Command Data | |
|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 2 Optional Bytes |
| **Value** | 0x5C | 0x00 | Reset | Time (Hours) |

The number of hours will correspond to the number of hours which will then be used for whatever units of display to the end user. If a reset is not required, the reset byte would be sent with a zero in the data payload.

Verification that the thermostat has actually been changed has to be done by requesting a status request. The data payload in the status response shall reflect whether or not the thermostat has actually been modified.

To add clarity, the following are examples of the packet received from the network:

(1) Reset the timer without changing the total time preset at installation:

**Table 86 – Reset Hum Pad Time Remaining Example**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x5C | 0x00 | 0x01 |

(2) Reset the timer and change the total time remaining to 18 hours:

**Table 87 – Reset and Change Time for Hum Pad Time Remaining Example**

| Field Name | Command Code | | Command Data | | |
|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 2 Optional Bytes | |
| **Value** | 0x5C | 0x00 | 0x01 | 0x12 | 0x00 |

(3) Change only the total time remaining to 255 hours:

**Table 88 – Change Time for Hum Pad Time Remaining Example**

| Field Name | Command Code | | Command Data | | |
|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte | 2 Optional Bytes | |
| **Value** | 0x5C | 0x00 | 0x00 | 0xFF | 0x00 |

## 6.47 Damper Closure Position Demand – Command Code 0x60

This command turns closes and opens the damper.

**Table 89 – Command Word – Dehumidification Demand**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x60 | 0x00 | Damper Closure Position Demand |

The data payload is two bytes. The first is a refresh timer and the second is the amount of damper closure requested. If the command is not refreshed before the timer runs out, damper closure will cease when the timer expires.

**Table 90 – Damper Closure Position Demand Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Control Command Refresh Timer (Refer Section 6.3) | Damper Closure Position Demand Percentage |

Demand percentage can have a value between 0-200 with a precision of 0.5%. A demand percentage bytes equal to zero is represented to be 0% and the damper is open completely. While a demand percentage bytes equal to two hundred is represented to be 100% and the demand is at its maximum that would make the damper totally closed.

## 6.48 Subsystem Busy Status – Command Code 0x61

This command informs the thermostat if a subsystem is busy and cannot respond to a control command.

**Table 91 – Command Word – Subsystem Busy**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x61 | 0x00 | Sys Busy Data Payload |

**Table 92 – Subsystem Busy Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Control Command Refresh Timer (Refer Section 6.3) | Busy (1) / Ready (0) |

The payload consists of two bytes. The first is a refresh timer and the second is the subsystem status. A 0x01 means the system is busy, and a 0x00 means the subsystem is no longer busy and is now able to respond. If the subsystem fails to refresh the command before the timer runs out, the thermostat will assume the subsystem is no longer busy.

## 6.49 Dehumidification Demand – Command Code 0x62

This command turns on and off dehumidification.

**Table 93 – Command Word – Dehumidification Demand**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x62 | 0x00 | Dehumidification Demand |

The data payload is two bytes. The first is a refresh timer and the second is the amount of dehumidification requested. If the command is not refreshed before the timer runs out, dehumidification will cease when the timer expires.

**Table 94 – Dehumidification Demand Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Control Command Refresh Timer (Refer Section 6.3) | Dehumidification Demand Percentage |

Demand percentage can have a value between 0-200 with a precision of 0.5%. A demand percentage bytes equal to zero is represented to be 0% and the demand is off. While a demand percentage bytes equal to two hundred is represented to be 100% and the demand is at its maximum.

## 6.50 Humidification Demand – Command Code 0x63

This command turns on and off humidification.

**Table 95 – Command Word – Humidification Demand**

| Field Name | Message Type | Packet Number | Payload Length | Command Code | | Command Data |
|---|---|---|---|---|---|---|
| **Field Length** | 1 Byte | 1 Byte | 1 Byte | 2 Bytes | | 2 Bytes |
| **Value** | 0x03 | 0x20 | 4 | 0x63 | 0x00 | Humidification Demand |

The data payload is two bytes. The first is a refresh timer and the second is the amount of humidification requested. If the command is not refreshed before the timer runs out, humidification will cease when the timer expires.

**Table 96 – Humidification Demand Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Control Command Refresh Timer (Refer Section 6.3) | Humidification Demand Percentage |

Demand percentage can have a value between 0-200 with a precision of 0.5%. A demand percentage bytes equal to zero is represented to be 0% and the demand is off. While a demand percentage bytes equal to two hundred is represented to be 100% and the demand is at its maximum.

## 6.51 Heat Demand – Command Code 0x64

This command turns on and off the heat.

**Table 97 – Command Word – Heat Demand**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x64 | 0x00 | Heat Demand |

The data payload is two bytes. The first is a refresh timer and the second is the amount of heat demand requested. If the command is not refreshed before the timer runs out, heating will cease when the timer expires.

**Table 98 – Heat Demand Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Control Command Refresh Timer (Refer Section 6.3) | Heat Demand Percentage |

Demand percentage can have a value between 0-200 with a precision of 0.5%. A demand percentage bytes equal to zero is represented to be 0% and the demand is off. While a demand percentage bytes equal to two hundred is represented to be 100% and the demand is at its maximum.

## 6.52 Cool Demand – Command Code 0x65

This command turns on and off cooling.

**Table 99 – Command Word – Cool Demand**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x65 | 0x00 | Cool Demand |

The data payload is two bytes. The first is a refresh timer and the second is the amount of cooling demand requested. If the command is not refreshed before the timer runs out, cooling will cease when the timer expires.

**Table 100 – Cool Demand Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Control Command Refresh Timer (Refer Section 6.3) | Cool Demand Percentage |

Demand percentage can have a value between 0-200 with a precision of 0.5%. A demand percentage bytes equal to zero is represented to be 0% and the demand is off. While a demand percentage bytes equal to two hundred is represented to be 100% and the demand is at its maximum.

## 6.53 Fan Demand – Command Code 0x66

This command turns on and off the fan.

**Table 101 – Command Word – Fan Demand**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 3 Bytes + Optional |
| **Value** | 0x66 | 0x00 | Fan Demand |

The data payload is three bytes with an optional fourth byte.

The first byte is a refresh timer. If the command is not refreshed before the timer runs out, fan operation will cease when the timer expires.

**Table 102 – Fan Demand Data Payload**

| Command Data | | | |
|---|---|---|---|
| 3 Bytes + 1 Optional Bytes | | | |
| Control Command Refresh Timer (Refer Section 6.3) | Fan Mode | Fan Demand Percentage | Fan On/Off Rate |

The second byte is the fan mode. The fan mode is used by the subsystem to determine which fan demand has the highest priority in cases where more than one fan demand is active. Table 103 shows the priority list, where 0 is lowest priority and 5 is the highest.

For example, if a furnace has a fan demand from its internal Auxiliary Heat and receives a Defrost Fan mode, then the furnace would operate using the Defrost Fan demand percentage.

**Table 103 – Fan Mode of Operation**

| Fan Mode | | |
|---|---|---|
| **Operation** | **Code** | **Application Priority** |
| Manual | 0 | 0 |
| Cool | 1 | 1 |
| Heat | 2 | 2 |
| Aux Heat | 3 | 3 |
| Emer Heat | 4 | 4 |
| Defrost | 5 | 5 |

Fan demand may be sent with the Fan Demand control command or as part of another command, such as the Defrost Demand control command.

The third byte is the amount of fan demand requested. Demand percentage can have a value between 0-200 with a precision of 0.5%. A demand percentage bytes equal to zero is represented to be 0% and the demand is off. While a demand percentage bytes equal to two hundred is represented to be 100% and the demand is at its maximum.

The remaining bytes in the payload are optional. If they are not present in the payload, the subsystem shall use default values.

The first optional byte (fourth overall) is the rate at which the fan demand is met.

The fan demand rate byte may be followed by a delay timer, which delays the amount of time the fan should turn on or off and defines what percentage the fan should be running at. If this byte is not present, there is no delay and the fan will turn on immediately at the speed requested.

## 6.54 Back-Up Heat Demand – Command Code 0x67

This command turns on and off the Back-Up heat.

**Table 104 – Command Word – Back-Up Heat Demand**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x67 | 0x00 | Back-Up Heat Demand |

**Table 105 – Back-Up Heat Demand Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Control Command Refresh Timer (Refer Section 6.3) | Back-Up Heat Demand Percentage |

The data payload is two bytes. The first is a refresh timer and the second is the amount of Back-Up heat demand requested. If the command is not refreshed before the timer runs out, Back-Up heating will cease when the timer expires.

Demand percentage can have a value between 0-200 with a precision of 0.5%. A demand percentage bytes equal to zero is represented to be 0% and the demand is off. While a demand percentage bytes equal to two hundred is represented to be 100% and the demand is at its maximum.

## 6.55 Defrost Heat Demand – Command Code 0x68

This command turns on the auxiliary heat at the indoor unit while the outdoor heat pump is defrosting.

**Table 106 – Command Word – Defrost Heat Demand**

| **Field Name** | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x68 | 0x00 | Defrost Heat Demand |

**Table 107 – Defrost Heat Demand Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Control Command Refresh Timer (Refer Section 6.3) | Defrost Heat Demand Percentage |

The data payload is two bytes. The first is a refresh timer and the second is the amount of auxiliary heat demand requested. If the command is not refreshed before the timer runs out, auxiliary heating will cease when the timer expires.

Demand percentage can have a value between 0-200 with a precision of 0.5%. A demand percentage bytes equal to zero is represented to be 0% and the demand is off. While a

demand percentage bytes equal to two hundred is represented to be 100% and the demand is at its maximum.

## 6.56 Aux / Alt Heat Demand – Command Code 0x69

This command turns on and off the Auxiliary/Alternate heat.

**Table 108 – Command Word – Aux/Alt Heat Demand**

| **Field Name** | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x69 | 0x00 | Aux/Alt Heat Demand |

**Table 109 – Aux/Alt Heat Demand Data Payload**

| **Command Data** | |
|---|---|
| 2 Bytes | |
| Control Command Refresh Timer (Refer Section 6.3) | Auxiliary/Alternate Heat Demand Percentage |

The data payload is two bytes. The first is a refresh timer and the second is the amount of Auxiliary/Alternate heat demand requested. If the command is not refreshed before the timer runs out, Auxiliary/Alternate heating will cease when the timer expires.

Demand percentage can have a value between 0-200 with a precision of 0.5%. A demand percentage bytes equal to zero is represented to be 0% and the demand is off. While a demand percentage bytes equal to two hundred is represented to be 100% and the demand is at its maximum.

## 6.57 Set Motor Speed – Command Code 0x6A

**Table 110 – Command Word – Set Motor Speed**

| **Field Name** | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x6A | 0x00 | Motor Speed |

**Table 111 – Set Motor Speed Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Speed Low Byte | Speed High Byte |

This command sets the speed of the motor. The speed value is calculated as follows:

Speed in RPM * 4 = data payload value

## 6.58 Set Motor Torque – Command Code 0x6B

**Table 112 – Command Word – Set Motor Torque**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x6B | 0x00 | Motor Torque |

**Table 113 – Set Motor Torque Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Torque Low Byte | Torque High Byte |

This command sets the torque of the motor. The torque value is calculated as follows:
Torque in Newton-Meters * 2048 = data payload value

## 6.59 Set Airflow Demand – Command Code 0x6C

**Table 114 – Command Word – Set Airflow Demand**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x6C | 0x00 | Airflow Demand |

**Table 115 – Set Airflow Demand Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Airflow Low Byte | Airflow High Byte |

This command sets the amount of airflow demand required. The airflow value is calculated as follows:

Airflow in cubic feet/minute * 4 = data payload value

## 6.60 Set Control Mode – Command Code 0x6D

**Table 116 – Command Word – Set Control Mode**

| **Field Name** | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x6D | 0x00 | Control Mode |

**Table 117 – Control Mode Values**

| Value | Mode | Control Command |
|---|---|---|
| 0x00 | Speed | Set Motor Speed – 0x6A |
| 0x01 | Torque | Set Motor Torque – 0x6B |
| 0x02 | Airflow | Set Airflow Demand – 0x6C |

This command sets the mode of operation for the motor. Once the mode is set, the corresponding Set control command shall be used to specify the operating parameters for that mode. Table 117 – Control Mode Values show the control command to use to set the operating parameters for each mode.

## 6.61 Set Demand Ramp Rate – Command Code 0x6E

**Table 118 – Command Word – Set Demand Ramp Rate**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x6E | 0x00 | Ramp Rate in Seconds |

This command specifies how quickly the motor should ramp up from zero demand to 100%. The ramp rate is specified in the number of seconds the motor should take to reach 100% demand. The ramp rate applies to speed, torque and airflow demand parameters.

## 6.62 Set Motor Direction – Command Code 0x6F

**Table 119 – Command Word – Set Motor Direction**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x6F | 0x00 | 0x00 = Counter-clockwise<br>0x01 - Clockwise |

This command specifies the direction of the motor at the lead end.

## 6.63 Set Motor Torque in Percent – Command Code 0x70

**Table 120 – Command Word – Set Motor Torque in Percent**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x70 | 0x00 | Torque |

**Table 121 – Set Motor Torque in Percent Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Percent Torque Low Byte | Percent Torque High Byte |

This command specifies the amount of torque required as a percentage of the maximum torque available.

The torque percentage value is calculated as follows:

(Percent * 65535) / 100 = data payload value in decimal

For example, 0x0000 = 0% and 0xFFFF = 100%.

## 6.64 Set Motor Position Demand – Command Code 0x71

**Table 122 – Command Word – Set Motor Position Demand**

| **Field Name** | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x71 | 0x00 | See Table Below |

**Table 123 – Set Motor Position Demand Data Payload**

| **Command Data** | |
|---|---|
| 2 Bytes | |
| Position (0x00 = Home) | Position Range |

This command specifies the position the motor should turn to expressed as a percentage of the maximum position range available.

If the position value is equal to 0x00, then the position should be the Home position.
If the position is NOT equal to 0x00, then the range value = 2 * actual position percentage range.

## 6.65 Set Blower Coefficient 1-5 – Command Codes 0x72 - 0x76

**Table 124 – Command Word – Set Blower Coefficient**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x72-0x76 | 0x00 | Blower Coefficient |

**Table 125 – Set Blower Coefficient Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Blower Coefficient Low Byte | Blower Coefficient High Byte |

The Set Blower Coefficient commands set the coefficients for blowers 1, 2, 3, 4 and 5, where command code 0x72 corresponds to blower 1, 0x73 to blower 2, etc. Blower coefficients provide the motor with information on the fan, ducting and other specifications of the HVAC system in order to allow internal calculation of required operational parameters.

## 6.66 Set Blower Identification 0-5 – Command Codes 0x77-0x7C

**Table 126 – Command Word – Set Blower Identification**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x77-0x7C | 0x00 | Blower Identification |

**Table 127 – Set Blower Identification Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Blower Identifier Low Byte | Blower Identifier High Byte |

The Set Blower Identification commands set the identification parameters for blowers 0, 1, 2, 3, 4 and 5, where command code 0x77 corresponds to blower 0, 0x78 to blower 1, etc.

## 6.67 Set Speed Limit – Command Code 0x7F

**Table 128 – Command Word – Set Speed Limit**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x7F | 0x00 | Speed Limit |

**Table 129 – Set Speed Limit Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Speed Limit Low Byte | Speed Limit High Byte |

This command defines the maximum speed in revolutions per minute (RPM) that the motor may be operated at.

The speed limit value is calculated as follows:

(Speed Limit in RPM * 4) = data payload value

## 6.68 Set Torque Limit – Command Code 0x80

**Table 130 – Command Word – Set Torque Limit**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x80 | 0x00 | Torque Limit |

**Table 131 – Set Torque Limit Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Torque Limit Low Byte | Torque Limit High Byte |

This command defines the maximum torque in Newton meters that the motor may be operated at.

The torque limit value is calculated as follows:

(Torque Limit in Newton meters * 2048) = data payload value

## 6.69 Set Airflow Limit – Command Code 0x81

**Table 132 – Command Word – Set Airflow Limit**

| **Field Name** | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x81 | 0x00 | Airflow Limit |

**Table 133 – Set Airflow Limit Data Payload**

| **Command Data** | |
|---|---|
| 2 Bytes | |
| Airflow Limit Low Byte | Airflow Limit High Byte |

This command defines the maximum airflow in cubic feet per meter that the motor may provide.

The airflow limit value is calculated as follows:

(Airflow Limit in cubic feet/meter * 4) = data payload value

## 6.70 Set Power Output Limit – Command Code 0x82

**Table 134 – Command Word – Set Power Output Limit**

| **Field Name** | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x82 | 0x00 | Power Output Limit |

**Table 135 – Set Power Output Limit Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Power Output Limit Low Byte | Power Output Limit High Byte |

This command defines the maximum power output that the motor may provide.

The power output limit value is calculated as follows:
(Power Output Limit in watts * 2) = data payload value

## 6.71 Set Device Temperature Limit – Command Code 0x83

**Table 136 – Command Word – Set Device Temperature Limit**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x83 | 0x00 | Device Temperature Limit |

**Table 137 – Set Device Temperature Limit Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Device Temperature Limit Low Byte | Device Temperature Limit High Byte |

This command defines the maximum temperature that the motor may operate at.

The temperature limit value is calculated as follows:

Data payload value = twos complement of (Temperature in Celsius * 128).

## 6.72 STOP Motor by Braking – Command Code 0x85

**Table 138 – Command Word – STOP Motor by Braking**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes |
| **Value** | 0x85 | 0x00 | |

**Table 139 – STOP Motor by Braking Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Low Byte<br>0x00= disable motor braking<br>0x01 = enable motor braking | High Byte<br>time in seconds (0-255) to wait before applying brake to motor |

This command stops the motor by braking.

## 6.73 RUN/STOP Motor – Command Code 0x86

**Table 140 – Command Word – RUN/STOP Motor**

| Field Name | Command Code | | Command Data |
|---|---|---|---|
| **Field Length** | 2 Bytes | | 1 Byte |
| **Value** | 0x86 | 0x00 | 0x00 = STOP<br>0x01 = RUN |

This command starts or stops the motor.

## 6.74 Set Demand Ramp Time – Command Code 0x88

**Table 141 – Command Word – Set Demand Ramp Time**

| Field Name | Message Type | Packet Number | Payload Length | Command Code | | Command Data |
|---|---|---|---|---|---|---|
| **Field Length** | 1 Byte | 1 Byte | 1 Byte | 2 Bytes | | 1 Byte |
| **Value** | 0x03 | 0x00 | 3 | 0x88 | 0x00 | Demand Ramp Time in seconds |

This command sets the time in seconds the motor will take to change from current demand to the next demand.

Time in seconds = Payload

## 6.75 Set Inducer Ramp Rate – Command Code 0x89

**Table 142 – Command Word – Set Inducer Ramp Rate**

| Field Name | Message Type | Packet Number | Payload Length | Command Code | | Command Data |
|---|---|---|---|---|---|---|
| **Field Length** | 1 Byte | 1 Byte | 1 Byte | 2 Bytes | | 1 Byte |
| **Value** | 0x03 | 0x00 | 3 | 0x89 | 0x00 | Inducer Ramp Rate |

This command sets the ramp rate for the inducer motor to go from current demand to the next demand. The payload byte is specified as follows:

**Table 143 - Set Inducer Ramp Rate Payload**

| Payload Value | Definition |
|---|---|
| 0x00 | Slew the demand as fast as possible |
| 0x01-0x7F | Objective Ramp Rate = value RPM/Sec |
| 0x80 | Slew the demand as fast as possible |
| 0x81-0xFF | Objective speed slew rate = (10*value) RPM/sec |

## 6.76 Set Blower Coefficient 6-10 – Command Codes 0x8A - 0x8E

**Table 144 – Command Word – Set Blower Coefficient 6-10**

| Field Name | Message Type | Packet Number | Payload Length | Command Code | | Command Data |
|---|---|---|---|---|---|---|
| **Field Length** | 1 Byte | 1 Byte | 1 Byte | 2 Bytes | | 2 Bytes |
| **Value** | 0x03 | 0x00 | 4 | 0x8A-0x8E | 0x00 | Blower Coefficient |

**Table 145 – Set Blower Coefficient 6-10 Data Payload**

| Command Data | |
|---|---|
| 2 Bytes | |
| Blower Coefficient Low Byte | Blower Coefficient High Byte |

The Set Blower Coefficient commands set the coefficients for blowers 6, 7, 8, 9 and 10, where command code 0x8A corresponds to blower 6, 0x8B to blower 7, etc. Blower coefficients provide the motor with information on the fan, ducting and other specifications

of the HVAC system in order to allow internal calculation of required operational parameters.

## 6.77 Publish Price – Command Code 0xE0

This control command is being reserved for the future implementation of a publish price command.

**Table 146 – Command Word – Publish Price**

| Field Name | Command Code | | Command Data | | | |
|---|---|---|---|---|---|---|
| **Field Length** | 2 Bytes | | TBD | TBD | TBD | TBD |
| **Value** | 0xE0 | 0x00 | TBD | TBD | TBD | TBD |

## 6.78 Water Heater Modify – Command Code 0xF0

**Table 147 – Command Word – Water Heater Setting Change**

| Field Name | Command Code | | Water Heater Modify Command code | Water Heater Modify Command data |
|---|---|---|---|---|
| **Field Length** | 2 Bytes | | 2 Bytes | TBC |
| **Value** | 0xF0 | 0x00 | Refer to Water Heater Application profile for options | |

# 7.0 Message Data Interfaces

## 7.1 General

Subsystems shall support the following message data interfaces:

Identification
Configuration
Status
Sensors (Air Conditioners and Heat Pumps)

Subsystems may support other message data interfaces at the manufacturer's discretion.

### 7.1.1 MDI size requirements

The maximum MDI size is 240 bytes.

### 7.1.2 ASCII string requirements

ASCII strings shall end with a 0x00 (NULL) character. If a field with a data type of ASCII string does not contain any other ASCII characters, it shall contain one 0x00 character as a placeholder to indicate that no data is stored in this field.

### 7.1.3 Mandatory fields

Fields shaded in green in the MDI tables are mandatory. These fields shall contain valid data for the subsystem at all times. They are pre-programmed at the factory.

### 7.1.4 Optional fields

Fields that are un-shaded in the MDI tables are optional. Subsystems are not required to have valid data in these fields. If the field is unused, it shall contain a single 0x00 character as a placeholder.

### 7.1.5 Updating of MDIs

MDI data shall be updated as soon as a change occurs. MDIs must always have valid, up-to-date data at all times.

## 7.2 Identification Message Data Interface (MDI)

All subsystems shall support the identification message data interface. All identification information is required with the identified default value(s) being used as a placeholder if the information is not known.

Database Identification (DB ID) is not implemented for this MDI.

### 7.2.1 Get Identification Data

**Table 148 – Get Identification data**

| Byte | Bit | Description | Size | Notes |
|---|---|---|---|---|
| 0-1 | 15-0 | Manufacturer ID | 2 Bytes | This is ClimateTalk Alliance assigned Manufacturer ID |
| 2 | 7-4 | ClimateTalk Standard Version | 4 Bits | 2 (This is ClimateTalk Standard version number that the control might be referring to) |
| | 3-0 | ClimateTalk Standard Revision | 4 Bits | 0 (This is ClimateTalk Standard revision number that the control might be referring to) |
| 3 | 7-0 | Number of Micros | 1 Byte | Number of micros with unique software version numbers<br>If > 0, then SW Version/Revision shall repeat |
| 4-n | 7-0 | SW Software (SW) Version | variable | ALPHA-NUMERIC ASCII (NULL = END) |
| | 7-0 | SW Software (SW) Revision | variable | ALPHA-NUMERIC ASCII (NULL = END) |
| | 7-0 | Serial Number | variable | ALPHA-NUMERIC ASCII (NULL = END) |
| TBC | 7-0 | Date Code – Month | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| | 7-0 | Date Code – Day | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| | 7-0 | Date Code – Year | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| | 7-0 | Verification/Test Date – Month | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |

| Byte | Bit | Description | Size | Notes |
|---|---|---|---|---|
| | 7-0 | Verification/Test Date – Day | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| | 7-0 | Verification/Test Date - Year | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| TBC | 7-0 | Installation Date – Month | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| | 7-0 | Installation Date – Day | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| | 7-0 | Installation Date – Year | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| | 7-0 | Address | variable | ALPHA-NUMERIC ASCII (NULL = END = 0x00).<br>Default/Unused shall be NULL. |
| | 7-0 | Zip Code | variable | ALPHA-NUMERIC ASCII (NULL = END).<br>Default/Unused shall be NULL |
| | 7-0 | Manufacturer | variable | ALPHA-NUMERIC ASCII (NULL = END).<br>Default/Unused shall be NULL |
| | 7-0 | Control Name | variable | ALPHA-NUMERIC ASCII (NULL = END).<br>Default/Unused shall be NULL |
| | 7-0 | Model | variable | ALPHA-NUMERIC ASCII (NULL = END).<br>Default/Unused shall be NULL |
| | 7-0 | Model Version | variable | ALPHA-NUMERIC ASCII (NULL = END).<br>Default/Unused shall be NULL |
| | 7-0 | Model Revision | variable | ALPHA-NUMERIC ASCII (NULL = END).<br>Default/Unused shall be NULL |

### 7.2.2 Set Identification Data

**Table 149 – Set Identification Data**

| Byte | Bit | Description | Size | Notes |
|---|---|---|---|---|
| 1 | 7-0 | Date Code – Month | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| 2 | 7-0 | Date Code – Day | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| 3 | 7-0 | Date Code – Year | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| 4 | 7-0 | Verification/Test Date - Month | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| 5 | 7-0 | Verification/Test Date - Day | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| 6 | 7-0 | Verification/Test Date - Year | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| 7 | 7-0 | Installation Date – Month | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| 8 | 7-0 | Installation Date – Day | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| 9 | 7-0 | Installation Date – Year | 1 Byte | Default/Unused shall be 0xFF (mm/dd/yy) |
| 10-n | 7-0 | Address | variable | ALPHA-NUMERIC ASCII (NULL = END). Default/Unused shall be NULL. |
| | 7-0 | Zip Code | variable | ALPHA-NUMERIC ASCII (NULL = END). Default/Unused shall be NULL |
| | 7-0 | Manufacturer | variable | ALPHA-NUMERIC ASCII (NULL = END). Default/Unused shall be NULL |
| | 7-0 | Control Name | variable | ALPHA-NUMERIC ASCII (NULL = END). Default/Unused shall be NULL |
| | 7-0 | Model | variable | ALPHA-NUMERIC ASCII (NULL = END). Default/Unused shall be NULL |

| Byte | Bit | Description | Size | Notes |
|---|---|---|---|---|
| | 7-0 | Model Version | variable | ALPHA-NUMERIC ASCII (NULL = END). Default/Unused shall be NULL |
| | 7-0 | Model Revision | variable | ALPHA-NUMERIC ASCII (NULL = END). Default/Unused shall be NULL |

## 7.3 Configuration Message Data Interface

### 7.3.1 Thermostat Configuration Data

The abbreviations used in the schedule section are defined as follows:

MOR Morning

DAY Day

NHT Night

OCC1 Occupied, 1st Time Step

OCC2 Occupied, 2nd Time Step

UNO1 Unoccupied, 1st Time Step

UNO2 Unoccupied, 2nd Time Step

**Table 150 – Thermostat Configuration Data**

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 33 | 0 | 7-0 | Auto-Configured System Type | 1 Byte | | 0 = UNKNOWN OR TO BE DETERMINED (--)<br>1 = CONVENTIONAL SYSTEM ( )<br>2 = HEAT PUMP SYSTEM (HP) |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | | | | | 3 = DUAL FUEL SYSTEM (DF)<br>4 = COOLING SYSTEM (AC)<br>5 = GAS HEAT SYSTEM (GH)<br>6 = ELEC HEAT SYSTEM (EH)<br>7 = ELECTRIC ONLY SYSTEM (ES)<br>8 = FAN ONLY SYSTEM (FN)<br>9 = GEOTHERMAL HEAT PUMP SYSTEM (GH)<br>10 = GEOTHERMAL DUAL FUEL SYSTEM (GF)<br>11 = BOILER SYSTEM (BC)<br>12 = BOILER HEAT PUMP SYSTEM (BH)<br>127 = This shall be the default value or is Unused.<br>255 = Other i.e. if the system configuration is anything different than specified above. |
| | | 1 | 7-4 | Auto-Configured Number Heat Stages | 4 Bits | | Value = Number of Stages<br>15 = Variable/Modulating |
| | | | 3-0 | Auto-Configured Number Cool Stages | 4 Bits | | Value = Number of Stages<br>15 = Variable/Modulating |
| | | 2 | 7-0 | Balance Point Set Temperature | 1 Byte | 0 - 127 | 0x00 = Balance Point System is off<br>0xFF = Default value indicating that this is not being used. |
| | | 3-4 | 15-0 | Filter Time - Hrs | 2 Bytes | | 0 = Disabled<br>0xFFFF = Default value indicating that it is not being used. |
| | | 5 | 7-0 | Temperature Display adjustment factor | 1 Byte | | Signed Byte – TBC<br>If this is unused, the value should be reported as 0. So, if a value of 0 is seen it could imply that the offset is zero or unused. |
| | | 6-7 | 15-0 | Programmable Hold Time | 2 Bytes | | 0 = Disabled<br>0xFFFF = Default value indicating that it is not being used. |
| | | 8 | 7-0 | Limited Heat Limit Range | 1 Byte | | 0xFF = Unused/ Default value indicating that it is not being used |
| | | 9 | 7-0 | Limited Cool Limit Range | 1 Byte | | 0xFF = Unused/ Default value indicating that it is not being used |
| | | 10 | 7 | EMR Enable/Disable Flag | 1 bit | 0-1 | 1 = Enabled 0 = Disabled |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | 6 | Keypad lockout Flag | 1 bit | 0-1 | 1 = Enabled 0 = Disabled |
| | | | 5 | Degree F/C mode flag | 1 bit | 0-1 | 1 = °F 0 = °C |
| | | | 4 | Fast 2nd Stage Cool/Heat/Aux Flag | 1 bit | 0-1 | 1 = ON 0 = OFF |
| | | | 3 | Continuous Display Light Flag | 1 bit | 0-1 | 1 = Enabled 0 = Disabled |
| | | | 2 | Compressor Lockout Flag | 1 bit | 0-1 | 1 = Enabled 0 = Disabled |
| | | | 1 | Adjustable Heat Cycle Rate | 1 bit | 0-1 | 1 = Fast 0 = Slow - (2 Rates). If only one rate setting available, use 0 |
| | | | 0 | Adjustable Cool Cycle Rate | 1 bit | 0-1 | 1 = Fast 0 = Slow - (2 Rates). If only one rate setting available, use 0. |
| | | 11 | 7-6 | Remote Sensor D Weight | 2 Bits | 0 - 3 | 0 = None or NA (Default) 1 = Low 2 = Med 3 = High |
| | | | 5-4 | Remote Sensor C Weight | 2 Bits | 0 - 3 | 0 = None or NA (Default) 1 = Low 2 = Med 3 = High |
| | | | 3-2 | Remote Sensor B Weight | 2 Bits | 0 - 3 | 0 = None or NA (Default) 1 = Low 2 = Med 3 = High |
| | | | 1-0 | Remote Sensor A Weight | 2 Bits | 0 - 3 | 0 = None or NA (Default) 1 = Low 2 = Med 3 = High |
| | | 12 | 7-6 | Local Sensor Weight | 2 Bits | 0 - 3 | 0 = None , 1 = Low 2 = Med 3 = High (Default/ No Remote sensors) |
| | | | 5 | Spare | 1 Bit | 0-1 | 0 |
| | | | 4 | Commercial/ Residential Thermostat | 1 Bit | | 0 = Residential (Default) 1 = Commercial |
| | | | 3-2 | Program Profile Type | 2 bits | | Bits 00 = Non-Programmable / Bits 10 = 7 Day |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | Bits 01 = 5-1-1 | Bits 11 = 5-2 |
| | | | 1-0 | Programmable Interval Type | 2 bits | | Bits 00 = 4-Step | Bits 01 = 2-Step |
| | | | | | | | Bits 10 = Non-Programmable | Bits 11 = Reserved. Do Not Use |
| | | | | | | | Depending on the profile and Step type, DBID 1 shall be populated. | |
| | | 13 | 7-0 | Air Handler Lock-Out Temperature | 1 Byte | | 0xFF = Default /Unused value indicating that it is not being used | |
| | | 14-15 | 15-0 | UV Lamp - Days | 2 Bytes | | 0 = Disabled; 0xFFFF = Default/Unused value indicating that it is not being used | |
| | | 16-17 | 15-0 | Humidifier Pad - Hours | 2 Bytes | | 0 = Disabled; 0xFFFF = Default/Unused value indicating that it is not being used. | |
| | | 18 | 7-4 | Number Aux Heat Stages | 4 Bits | | Value = Stages - Except Variable System = 15 | |
| | | | 3-0 | Number of Fan Stages | 4 Bits | | Value = Stages - Except Variable System = 15 | |
| | | 19 | 7-0 | Adjustable Aux Heat Cycle Rates | 1 Byte | | Default/Unused is 0; Percentage - 0.5% Increments. (Refer to Control Command 0x48 for details). | |
| | | 20 | 7-0 | Adjustable Heat Cycle Rates | 1 Byte | | IF > 0, DB ID 0 Byte 10 Bit 1 Ignored; Percentage - 0.5% Increments. (Refer to Control Command 0x48 for details). | |
| | | 21 | 7-0 | Adjustable Cool Cycle Rates | 1 Byte | | IF > 0, DB ID 0 Byte 10 Bit 0 Ignored; Percentage - 0.5% Increments. (Refer to Control Command 0x48 for details). | |
| | | 22 | 7 | Real Time Clock Change Lockout | 1 Bit | | 0 = Lockout Disabled/Unused; 1 = Lockout Enabled; | |
| | | | 6 | O/B Mode | 1 Bit | | 0 = O Mode/Unavailable; 1 = B Mode | |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | 5 | Beeper Enable | 1 Bit | | 0 = Disabled/Unavailable; 1 = Enabled |
| | | | 4-3 | Spares | 3 Bits | | 0 |
| | | | 2 | Keypad Lockout - Partial | 1 Bit | | 0 = Lockout Disabled/Unavailable; 1 = Lockout Enabled |
| | | | 1 | Keypad Lockout - Total | 1 Bit | | 0 = Lockout Disabled/Unavailable; 1 = Lockout Enabled |
| | | | 0 | Daylight Savings Time | 1 Bit | | 0 = DST Disabled/Unavailable; 1 = Enabled |
| | | 23 | 7-0 | GMT Offset (Hours) | 1 Byte | | Signed Byte (Scale 4) |
| | | 24 | 7-0 | Display Contrast | 1 Byte | | 0% (Lowest) to 100% (Highest) |
| | | 25-26 | 15-0 | Communications Fault Timer Setting | 1 Bytes | | 30 Seconds to 900 Seconds (15Minutes). This indicates the time that the controls shall wait before reacting to a communication fault. |
| | | 27 | 7-1 | Spares | 7 Bits | | 0 |
| | | | 0 | Phone Number Display on Comm Fault | 1 Bit | | 0 = Disabled/Unused; 1 = Enabled |
| | | 28 | 7-0 | Node Type of Indoor Unit | 1 Byte | | 0xFF = Default/Unused value indicating that it is not being used |
| | | 29 | 7-0 | Node Type of Outdoor Unit | | | 0xFF = Default/Unused value indicating that it is not being used |
| | | 30 | 7-4 | Spares | 4 Bits | | 0 |
| | | | 3 | Humidification Capable | 1 bit | | 0 = Not Capable; 1 = Capable |
| | | | 2 | Dehumidification Capable | 1 bit | | 0 = Not Capable; 1 = Capable |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | 1 | Independent HUM capable | 1 bit | | 0 = Not Capable; 1 = Capable |
| | | | 0 | Independent DHM Capable | 1 bit | | 0 = Not Capable; 1 = Capable |
| | | 31 | 7-4 | Spares | 4 Bits | | 0 |
| | | | 3 | 5 – 2 Programmable Profile Capable | 1 Bit | | 0 = Not Capable; 1 = Capable |
| | | | 2 | 7 Day Programmable Profile Capable | 1 Bit | | 0 = Not Capable; 1 = Capable |
| | | | 1 | 5-1-1 Programmable Profile Capable | 1 Bit | | 0 = Not Capable; 1 = Capable |
| | | | 0 | Non-Programmable Profile Capable | 1 Bit | | 0 = Not Capable; 1 = Capable |
| | | 32 | 7-3 | Spares | 5 Bits | | 0 |
| | | | 2 | 2 Steps Capable | 1 Bit | | 0 = Not Capable; 1 = Capable |
| | | | 1 | Non-Programmable (1 Step) Capable | 1 Bit | | 0 = Not Capable; 1 = Capable |
| | | | 0 | 4 Steps Capable | 1 Bit | | 0 = Not Capable; 1 = Capable |
| 1 | 112 | 0 | 7-0 | Heat Profile - Mon - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 1 | 7-0 | Heat Profile - Mon - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 2 | 7-0 | Heat Profile - Mon - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 3 | 7-0 | Heat Profile - Mon - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 4 | 7-0 | Heat Profile - Mon - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 5 | 7-0 | Heat Profile - Mon - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 6 | 7-0 | Heat Profile - Mon - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 7 | 7-0 | Heat Profile - Mon - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 8 | 7-0 | Heat Profile - Tue - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 9 | 7-0 | Heat Profile - Tue - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 10 | 7-0 | Heat Profile - Tue - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 11 | 7-0 | Heat Profile - Tue - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 12 | 7-0 | Heat Profile - Tue - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 13 | 7-0 | Heat Profile - Tue - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 14 | 7-0 | Heat Profile - Tue - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 15 | 7-0 | Heat Profile - Tue - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 16 | 7-0 | Heat Profile - Wed - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 17 | 7-0 | Heat Profile - Wed - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 18 | 7-0 | Heat Profile - Wed - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 19 | 7-0 | Heat Profile - Wed - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 20 | 7-0 | Heat Profile - Wed - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 21 | 7-0 | Heat Profile - Wed - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 22 | 7-0 | Heat Profile - Wed - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 23 | 7-0 | Heat Profile - Wed - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 24 | 7-0 | Heat Profile - Thur - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 25 | 7-0 | Heat Profile - Thur - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 26 | 7-0 | Heat Profile - Thur - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 27 | 7-0 | Heat Profile - Thur - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 28 | 7-0 | Heat Profile - Thur - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 29 | 7-0 | Heat Profile - Thur - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 30 | 7-0 | Heat Profile - Thur - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 31 | 7-0 | Heat Profile - Thur - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 32 | 7-0 | Heat Profile - Fri - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 33 | 7-0 | Heat Profile - Fri - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 34 | 7-0 | Heat Profile - Fri - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 35 | 7-0 | Heat Profile - Fri - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 36 | 7-0 | Heat Profile - Fri - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 37 | 7-0 | Heat Profile - Fri - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 38 | 7-0 | Heat Profile - Fri - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 39 | 7-0 | Heat Profile - Fri - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 40 | 7-0 | Heat Profile - W/E 1 - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 41 | 7-0 | Heat Profile - W/E 1 - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 42 | 7-0 | Heat Profile - W/E 1 - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 43 | 7-0 | Heat Profile - W/E 1 - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 44 | 7-0 | Heat Profile - W/E 1 - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 45 | 7-0 | Heat Profile - W/E 1 - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 46 | 7-0 | Heat Profile - W/E 1 - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 47 | 7-0 | Heat Profile - W/E 1 - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 48 | 7-0 | Heat Profile - W/E 2 - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 49 | 7-0 | Heat Profile - W/E 2 - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 50 | 7-0 | Heat Profile - W/E 2 - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 51 | 7-0 | Heat Profile - W/E 2 - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 52 | 7-0 | Heat Profile - W/E 2 - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 53 | 7-0 | Heat Profile - W/E 2 - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 54 | 7-0 | Heat Profile - W/E 2 - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 55 | 7-0 | Heat Profile - W/E 2 - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 56 | 7-0 | Cool Profile - Mon - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 57 | 7-0 | Cool Profile - Mon - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 58 | 7-0 | Cool Profile - Mon - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 59 | 7-0 | Cool Profile - Mon - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 60 | 7-0 | Cool Profile - Mon - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 61 | 7-0 | Cool Profile - Mon - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 62 | 7-0 | Cool Profile - Mon - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 63 | 7-0 | Cool Profile - Mon - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 64 | 7-0 | Cool Profile - Tue - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 65 | 7-0 | Cool Profile - Tue - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 66 | 7-0 | Cool Profile - Tue - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 67 | 7-0 | Cool Profile - Tue - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 68 | 7-0 | Cool Profile - Tue - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 69 | 7-0 | Cool Profile - Tue - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 70 | 7-0 | Cool Profile - Tue - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 71 | 7-0 | Cool Profile - Tue - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 72 | 7-0 | Cool Profile - Wed - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 73 | 7-0 | Cool Profile - Wed - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 74 | 7-0 | Cool Profile - Wed - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 75 | 7-0 | Cool Profile - Wed - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 76 | 7-0 | Cool Profile - Wed - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 77 | 7-0 | Cool Profile - Wed - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 78 | 7-0 | Cool Profile - Wed - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 79 | 7-0 | Cool Profile - Wed - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 80 | 7-0 | Cool Profile - Thur - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 81 | 7-0 | Cool Profile - Thur - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 82 | 7-0 | Cool Profile - Thur - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 83 | 7-0 | Cool Profile - Thur - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 84 | 7-0 | Cool Profile - Thur - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 85 | 7-0 | Cool Profile - Thur - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 86 | 7-0 | Cool Profile - Thur - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 87 | 7-0 | Cool Profile - Thur - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 88 | 7-0 | Cool Profile - Fri - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 89 | 7-0 | Cool Profile - Fri - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 90 | 7-0 | Cool Profile - Fri - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 91 | 7-0 | Cool Profile - Fri - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 92 | 7-0 | Cool Profile - Fri - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 93 | 7-0 | Cool Profile - Fri - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 94 | 7-0 | Cool Profile - Fri - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 95 | 7-0 | Cool Profile - Fri - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 96 | 7-0 | Cool Profile - W/E 1 - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 97 | 7-0 | Cool Profile - W/E 1 - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 98 | 7-0 | Cool Profile - W/E 1 - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 99 | 7-0 | Cool Profile - W/E 1 - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 100 | 7-0 | Cool Profile - W/E 1 - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 101 | 7-0 | Cool Profile - W/E 1 - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 102 | 7-0 | Cool Profile - W/E 1 - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 103 | 7-0 | Cool Profile - W/E 1 - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 104 | 7-0 | Cool Profile - W/E 2 - MOR/OCC1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 105 | 7-0 | Cool Profile - W/E 2 - MOR/OCC1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 106 | 7-0 | Cool Profile - W/E 2 - DAY/UNO1 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 107 | 7-0 | Cool Profile - W/E 2 - DAY/UNO1 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 108 | 7-0 | Cool Profile - W/E 2 - EVE/OCC2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 109 | 7-0 | Cool Profile - W/E 2 - EVE/OCC2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |
| | | 110 | 7-0 | Cool Profile - W/E 2 - NHT/UNO2 Start Time | 1 Byte | | b7 Damper bit (Commercial Only), b6-b2 Hour b1,b0 Min |
| | | 111 | 7-0 | Cool Profile - W/E 2 - NHT/UNO2 Temp | 1 Byte | | b7 Prg Fan b6-b0 Set Point |

#### 7.3.1.1 Damper Bit

The damper bit is generally used only in commercial equipment. If the bit is unused, it shall be set to 0x00. The damper bit is defined as follows:

**Table 151 – Damper Bit Definition**

| Value | Meaning |
|---|---|
| 0 | Damper is open |
| 1 | Damper is closed |

#### 7.3.1.2 Programmable Fan Bit

The Programmable Fan bit specifies the fan mode. Note that it does not specify the current operating state of the fan. The Programmable Fan bit is defined as follows:

**Table 152 – Programmable Fan Bit Definition**

| Value | Meaning |
|---|---|
| 0 | Fan mode is Off/Auto |
| 1 | Fan mode is On |

### 7.3.2 Furnace Configuration Data

**Table 153 – Furnace Configuration Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes | |
|---|---|---|---|---|---|---|---|---|
| 0 | 9 | 0 | 7-4 | No of Fan Speeds | 4 bits | 0-4 | Value = Number of Stages<br>15 = Variable/Modulating | |
| | | | 3-0 | No of Inducer Stages | 4 bits | 0-4 | Value = Number of Stages<br>15 = Variable/Modulating | |
| | | 1 | 7-4 | No of Heat Stages | 4 bits | 0-4 | Value = Number of Stages<br>15= Variable/Modulating | |
| | | | 3-0 | No of Cool Stages | 4 bits | 0-4 | Value = Number of Stages<br>15 = Variable/Modulating | |
| | | 2 | 7-6 | Pressure Configuration | 2 bits | | 0x00 = Sensorless | 0x01 = PS |
| | | | | | | | 0x02 = Transducer | 0x03 = Spare |
| | | | 5-4 | Ignition Type | 2 bits | | 0x00 = Spark | 0x01 = Silicon Carbide |
| | | | | | | | 0x02 = Silicon Nitride | 0x03 = Spare |
| | | | 3-2 | Fuel Type | 2 bits | | 0x00 = Propane | 0x01 = Natural |
| | | | | | | | 0x02 = Spare | 0x03 = Spare |
| | | | 1-0 | HVAC Operation | 2 bits | | 0x00 = 24VAC | 0x01 = Serial |
| | | | | | | | 0x02 = Combo | 0x03 = Water |
| | | 3 | 7-3 | Spares | 5 bits | | 0 | |
| | | | 2 | Humidification Capable in Cool Mode | 1 bit | | 0 = Not Capable; 1= Capable | |
| | | | 1 | Humidification Capable | 1 bit | | 0 = Not Capable; 1= Capable | |
| | | | 0 | Dehumidification Capable | 1 bit | | 0 = Not Capable; 1= Capable | |
| | | 4 | 7-0 | Furnace Size | 1 byte | | 0 = Unavailable; Units are in KBTU | |
| | | 5 | 7-0 | Circulator/Blower Motor Manufacturer | 1 byte | | 0 = Unavailable; Serial Motor Manufacturer Code | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | 6 | 7-0 | Circulator/Blower Motor Size, HP | 1 byte | | 0 = Unavailable;<br>Units are in encoded HP.<br>3 = 1/3 HP; 6 = ½ HP; 9 = ¾ HP; 12 = 1 HP; 24 = 2 HP |
| | | 7-8 | 15-0 | Circulator/Blower Maximum Airflow | 2 Bytes | | CFM Units |
| 1[3] | 5 | 0 | 7-0 | CFM Per Ton | 1 Byte | 0-5 | TBC |
| | | 1 | 7-4 | Selected Tonnage (manual) | 4 bits | Whole Digits (0-9) | 0x00 = Unavailable/Unused |
| | | | 3-0 | | 4 bits | Tenths (0-9) | |
| | | 2 | 7-0 | HEAT CFM | 1 byte | divide by 10 | 0x00 = Unavailable/Unused. The reported value should be multiplied by 10. |
| | | 3 | 7-0 | Cool Speed Trim Adjustment | 1 byte | -100% to 100% | 0 = Unused or Trim Adjustment is zero.<br>Resolution is in 1% increments.<br>Mixed Mode Operation Only |
| | | 4 | 7-0 | Heat Speed Trim Adjustment | 1 byte | -100% to 100% | 0 = Unused or Trim Adjustment is zero.<br>Resolution is in 1% increments.<br>Mixed Mode Operation Only |

[3] DB-ID 0 is mandatory and DB-ID 1 is optional for Furnace controls. But if DB-ID 1 is sent out, it shall have to have a minimum number of bytes identified above and should report the default value for each byte.

### 7.3.3 Air Handler Configuration Data

**Table 154 – Air Handler Configuration Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | EU Range | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 8 | 0 | 7-4 | No of Fan Speeds | 4 bits | 0-4 | Value = Number of Stages<br>15 = Variable/Modulating |
| | | | 3-0 | Reserved | 4 bits | 0-4 | 0 |
| | | 1 | 7-4 | No of Heat Stages | 4 bits | 0-4 | Value = Number of Stages<br>15 = Variable/Modulating |
| | | | 3-0 | Reserved | 4 bits | 0-4 | 0 |
| | | 2 | 2-7 | Reserved | 6 bits | | 0 |
| | | | 1-0 | HVAC Operation | 2 bits | | 0x00 = 24VAC; 0x01 = Serial<br>0x02 = Combo; 0x03 = Reserved |
| | | 3 | 7-3 | Reserved | 7 bits | | 0 |
| | | | 2 | Humidification Capable in Cool Mode | 1 bit | | 0 = Not Capable; 1= Capable |
| | | | 1 | Humidification Capable | 1 bit | | 0 = Not Capable; 1= Capable |
| | | | 0 | Dehumidification Capable | 1 byte | | 0 = Not Capable; 1= Capable |
| | | 4 | 7-0 | Circulator/Blower Motor HP | 1 byte | | 0 = Unavailable; Units are in encoded HP.<br>3 = 1/3 HP; 6 = ½ HP;<br>9 = ¾ HP; 12 = 1 HP; 24 = 2 HP |
| | | 5 | 7-0 | Circulator/Blower Motor Manufacturer | 1 byte | | 0 = Unavailable; Serial Motor Manufacturer Code |
| | | 6-7 | 15-0 | Circulator/Blower Maximum Airflow | 2 Bytes | | CFM Units |
| 1[4] | 6 | 0 | 7-0 | Air Handler Size | 1 byte | | 0 = Unavailable; Units are in KW |
| | | 1 | 7-0 | CFM Per Ton | 1 Byte | 0-5 | (TBC) |
| | | 2 | 7-4 | Selected Tonnage (manual) | 4 bits | Whole Digits (0-9) | 0x00 = Unavailable/Unused |
| | | | 3-0 | | 4 bits | Tenths (0-9) | |

[4] DB-ID 1 is optional for Air Handler. If DB-ID 1 is sent out, it shall have to have a minimum number of bytes identified above and should report a value for each byte as mentioned above.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | 3 | 7-0 | HEAT CFM | 1 byte | divide by 10 | 0x00 = Unavailable/Unused. The reported value should be multiplied by 10. |
| | | 4 | 7-0 | Cool Speed Trim Adjustment | 1 byte | -100% to 100% | 0 = Unused or Trim Adjustment is zero.<br>Resolution is in 1% increments.<br>Mixed Mode Operation Only |
| | | 5 | 7-0 | Heat Speed Trim Adjustment | 1 byte | -100% to 100% | 0 = Unused or Trim Adjustment is zero.<br>Resolution is in 1% increments.<br>Mixed Mode Operation Only |

### 7.3.4 Air Conditioner Configuration Data

**Table 155 – Air Conditioner Configuration Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes | |
|---|---|---|---|---|---|---|---|---|
| 0 | 5 | 0 | 7-4 | No of Outdoor Fan Speeds | 4 bits | 0-4 | Value = Number of Stages<br>15 = Variable/Modulating | |
| | | | 3-0 | Reserved | 4 bits | | 0 | |
| | | 1 | 7-4 | Reserved | 4 bits | | 0 | |
| | | | 3-0 | No of Cool Stages | 4 bits | 0-4 | Value = Number of Stages<br>15 = Variable/Modulating | |
| | | 2 | 7-2 | Reserved | 6 bits | | 0 | |
| | | | 0-1 | HVAC Operation | 2 bits | | 0x00 = 24VAC | 0x01 = Serial |
| | | | | | | | 0x02 = Combo | 0x03 = Reserved |
| | | 3 | 7-1 | Reserved | 7 bits | | 0 | |
| | | | 0 | Dehumidification Capable | 1 byte | | 0 = Not Capable; 1 = Capable. | |
| | | 4 | 7-0 | Tonnage | 1 byte | 1.5 - 5 | 0 = Unavailable.<br>The reported value is in 0.5 ton increments | |
| 1[5] | 2 | 0 | 7-0 | Cool Speed Trim Adjustment | 1 byte | | 0 = Unused or Trim Adjustment is zero.<br>Resolution is in 1% increments.<br>Mixed Mode Operation Only | |

[5] DB-ID 1 is optional for Air Conditioner. This is typical for Mixed Mode Operation. If DB-ID 1 is sent out, it shall have to have a minimum number of bytes identified above and should report a value for each byte as mentioned above.

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 1 | 7-0 | Reserved | 1 byte | | 0 |
| 2[6] | 4 | 0 | 7-0 | Outdoor Motor Manufacturer | 1 byte | | 0 = Unavailable; Serial Motor Manufacturer Code |
| | | 1 | 7-0 | Outdoor Fan Motor Size, HP | 1 byte | | 0 = Unavailable;<br>Units are in encoded HP.<br>3 = 1/3 HP; 6 = ½ HP; 9 = ¾ HP; 12 = 1 HP; 24 = 2 HP |
| | | 2-3 | 15-0 | Outdoor Maximum Airflow | 2 Bytes | | CFM Units |

### 7.3.5 Heat Pump Configuration Data

**Table 156 – Heat Pump Configuration Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 5 | 0 | 7-4 | No of Outdoor Fan Speeds | 4 bits | 0-4 | Value = Number of Stages<br>15 = Variable/Modulating |
| | | | 3-0 | Reserved | 4 bits | | 0 |
| | | 1 | 7-4 | No of Heat Stages | 4 bits | 0-4 | Value = Number of Stages<br>15 = Variable/Modulating |

[6] DB-ID 2 is optional for Air Conditioner. This is typical when Air Conditioner is communicating with a ClimateTalk blower motor. If DB-ID 2 is sent out, it shall have to have a minimum number of bytes identified above and should report a value for each byte as mentioned above.

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes | |
|---|---|---|---|---|---|---|---|---|
| | | | 3-0 | No of Cool Stages | 4 bits | 0-4 | Value = Number of Stages<br>15 = Variable/Modulating | |
| | | 2 | 7-2 | Reserved | 6 bits | | 0 | |
| | | | 0-1 | HVAC Operation | 2 bits | | 0x00 = 24VAC | 0x01 = Serial |
| | | | | | | | 0x02 = Combo | 0x03 = Reserved |
| | | 3 | 7-1 | Reserved | 7 bits | | 0 | |
| | | | 0 | Dehumidification Capable | 1 byte | | 0 = Not Capable; 1 = Capable | |
| | | 4 | 7-0 | Tonnage | 1 byte | 1.5 - 5 | 0 = Unavailable.<br>The reported value is in 0.5 ton increments | |
| 1[7] | 2 | 0 | 7-0 | Cool Speed Trim Adjustment | 1 byte | | 0 = Unused or Trim Adjustment is zero.<br>Resolution is in 1% increments.<br>Mixed Mode Operation Only | |
| | | 1 | 7-0 | Heat Speed Trim Adjustment | 1 byte | | 0 = Unused or Trim Adjustment is zero.<br>Resolution is in 1% increments.<br>Mixed Mode Operation Only | |
| 2[8] | 4 | 0 | 7-0 | Outdoor Motor Manufacturer | 1 byte | | 0 = Unavailable;<br>Serial Motor Manufacturer Code | |
| | | 1 | 7-0 | Outdoor Fan Motor Size, HP | 1 byte | | 0 = Unavailable;<br>Units are in encoded HP.<br>3 = 1/3 HP; 6 = ½ HP; 9 = ¾ HP; 12 = | |

[7] DB-ID 1 is optional for Heat Pump. This is typical for Mixed Mode Operation. If DB-ID 1 is sent out, it shall have to have a minimum number of bytes identified above and should report a value for each byte as mentioned above.

[8] DB-ID 2 is optional for Heat Pump. This is typical when Heat Pump is communicating with a ClimateTalk blower motor. If DB-ID 2 is sent out, it shall have to have a minimum number of bytes identified above and should report a value for each byte as mentioned above.

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | | | | | 1 HP; 24 = 2 HP |
| | | 2-3 | 15-0 | Outdoor Maximum Airflow | 2 Bytes | | CFM Units |

### 7.3.6 Crossover Configuration Data

**Table 157 – Crossover Configuration Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Raw Range | Notes | |
|---|---|---|---|---|---|---|---|---|
| 0 | 6 | 0 | 7-4 | No of Fan Speeds Enabled | 4 bits | 0-4 | Value = # Stage - 15 = Variable | |
| | | | 3-0 | Reserved | 4 bits | 0-4 | 0 | |
| | | 1 | 7-4 | No of Heat Stages Enabled | 4 bits | 0-4 | Value = # Stage - 15 = Variable | |
| | | | 3-0 | No of Cool Stages Enabled | 4 bits | 0-4 | Value = # Stage - 15 = Variable | |
| | | 2 | 7-4 | No of Aux Stages Enabled | 4 bits | | Value = # Stage - 15 = Variable | |
| | | | 3 | Indoor Heat Type configured | 1 bit | | 0 = Gas Heat | 1 = Electric |
| | | | 2 | Outdoor Equipment Type configured | 1 bit | | 0 = Air conditioner | 1 = Heat Pump |
| | | | 1 | Equipment Type of Crossover configured | 1 bit | | 0 = Indoor | 1 = Outdoor |
| | | | 0 | Mode of Operation | 1 bit | | 0 = 24VAC Output | 1 = Mini-Tstat |
| | | 3 | 7-2 | Spares | 6 bits | | 0 | |
| | | | 1 | Humidification Enabled | 1 bit | | 0 = Disabled or Unavailable; 1 = Enabled | |
| | | | 0 | Dehumidification Enabled | 1 bit | | 0 = Disabled or Unavailable; 1 = Enabled | |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Raw Range | Notes | |
|---|---|---|---|---|---|---|---|---|
| | | 4 | 7 | Independent HUM enabled | 1 bit | | 0 = Disabled or Unavailable; 1 = Enabled | |
| | | | 6 | Independent DHM enabled | 1 bit | | 0 = Disabled or Unavailable; 1 = Enabled | |
| | | | 5-0 | Spares | 6 bits | | 0 | |
| | | 5 | 7-6 | Spares | 2 Bits | | 0 | |
| | | | 5-4 | Speed Trim Adjustment | 2 Bits | | 0 = Unused or Trim adjustment is zero. | 1 = -10% |
| | | | | | | | 2 = 10% | 3 = 0% |
| | | | 3-2 | Air Flow (CFM/Ton) | 2 Bits | | 0 = 350<br>2 = 400 | 1 = 375<br>3 = 400 |
| | | | 1-0 | Tonnage | 2 bits | | 0 = 2 Ton<br>2 = 4 Ton | 1 = 3 Ton<br>3 = 5 Ton |
| 1 | 6 | 0 | 7-4 | No of Fan Speeds Capable | 4 bits | 0-4 | Value = # Stage - 15 = Variable | |
| | | | 3-0 | Spares | 4 bits | 0-4 | 0 | |
| | | 1 | 7-4 | No of Heat Stages Capable | 4 bits | 0-4 | Value = # Stage - 15 = Variable | |
| | | | 3-0 | No of Cool Stages Capable | 4 bits | 0-4 | Value = # Stage - 15 = Variable | |
| | | 2 | 7-4 | No of Aux Stages Capable | 4 bits | | Value = # Stage - 15 = Variable | |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Raw Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | 3-0 | Spares | 8 bits | | 0 |
| | | 3 | 7-2 | Spares | 2 bits | | 0 |
| | | | 1 | Humidification Capable | 1 bit | | 0 = Not Capable; 1 = Capable |
| | | | 0 | Dehumidification Capable | 1 bit | | 0 = Not Capable; 1 = Capable |
| | | 4 | 7 | Independent HUM capable | 1 bit | | 0 = Not Capable; 1 = Capable |
| | | | 6 | Independent DHM Capable | 1 bit | | 0 = Not Capable; 1 = Capable |
| | | | 5-0 | Spares | 6 bits | | 0 |
| | | 5 | 7-0 | Reserved | 8 bits | | 0 |

### 7.3.7 Zone Controller Configuration Data

**Table 158 – Zone Controller Configuration Data**

| DB ID TAG | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| 0 | 10 | 0 | 7-0 | Auto-Configured System Type | | 0 = UNKNOWN OR TO BE DETERMINED (--)<br>1 = CONVENTIONAL SYSTEM ( )<br>2 = HEAT PUMP SYSTEM (HP)<br>3 = DUAL FUEL SYSTEM (DF)<br>4 = COOLING SYSTEM (AC)<br>5 = GAS HEAT SYSTEM (GH)<br>6 = ELEC HEAT SYSTEM (EH)<br>7 = ELECTRIC ONLY SYSTEM (ES)<br>8 = FAN ONLY SYSTEM (FN)<br>9 = GEOTHERMAL HEAT PUMP SYSTEM (GH)<br>10 = GEOTHERMAL DUAL FUEL SYSTEM (GF)<br>11 = BOILER SYSTEM (BC)<br>12 = BOILER HEAT PUMP SYSTEM (BH)<br>127 = This shall be the default value or is Unused.<br>255 = Other i.e. if the system configuration is anything different than specified above. |
| | | 1 | 7-4 | Auto-Configured Number Heat Stages | | Value = Number of Stages<br>15 = Variable/Modulating |
| | | | 3-0 | Auto-Configured Number Cool Stages | | Value = Number of Stages<br>15 = Variable/Modulating |
| | | 2 | 7-0 | Balance Point Set Temperature | | 0x00 = Balance Point System is off<br>0xFF = Default value indicating that this is not being used. |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| | | 3-4 | 15-0 | Filter Time - Hrs | | 0 = Disabled<br>0xFFFF = Default value indicating that it is not being used. |
| | | 5-6 | 15-0 | UV Time - Hrs | | 0 = Disabled<br>0xFFFF = Default value indicating that it is not being used. |
| | | 7-8 | 15-0 | Humidifier Pad Time - Hrs | | 0xFFFF = Default value indicating that it is not being used. |
| | | 9 | 7-3 | Reserved | | 0 |
| | | | 2 | Humidification Capable in Cool Mode | | 0 = Not Capable; 1= Capable |
| | | | 1 | Humidification Capable | | 0 = Not Capable; 1= Capable |
| | | | 0 | Dehumidification Capable | | 0 = Not Capable; 1= Capable |

### 7.3.8 Zone User Interface Configuration Data

The abbreviations used in the schedule section are defined as follows:

MOR Morning
DAY Day
NHT Night
OCC1 Occupied, 1st Time Step
OCC2 Occupied, 2nd Time Step
UNO1 Unoccupied, 1st Time Step

UNO2 Unoccupied, 2nd Time Step

**Table 159 – Zone Controller User Interface Configuration Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes | |
|---|---|---|---|---|---|---|---|---|
| 0 | 3 | 0 | 7-5 | Spare | 3 Bit | | 0 | |
| | | | 4 | Reserved | 1 Bit | | 0 | |
| | | | 3-2 | Program Profile Type | 2 bits | | Bits 00 = Non-Programmable | Bits 10 = 7 Day |
| | | | | | | | Bits 01 = 5-1-1 | Bits 11 = 5-2 |
| | | | 1-0 | Programmable Interval Type | 2 bits | | Bits 00 = 4-Step | Bits 01 = 2-Step |
| | | | | | | | Bits 10 = Non-Programmable | Bits 11 = Reserved. Do Not Use |
| | | 1 | 7-4 | Spares | 4 Bits | | 0 | |
| | | | 3 | 5 – 2 Programmable Profile Capable | 1 Bit | | 0 = Not Capable; 1 = Capable | |
| | | | 2 | 7 Day Programmable Profile Capable | 1 Bit | | 0 = Not Capable; 1 = Capable | |
| | | | 1 | 5-1-1 Programmable Profile Capable | 1 Bit | | 0 = Not Capable; 1 = Capable | |
| | | | 0 | Non-Programmable Profile Capable | 1 Bit | | 0 = Not Capable; 1 = Capable | |
| | | 2 | 7-3 | Spares | 5 Bits | | 0 | |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | 2 | 2 Steps Capable | 1 Bit | | 0 = Not Capable; 1 = Capable |
| | | | 1 | Non-Programmable (1 Step) Capable | 1 Bit | | 0 = Not Capable; 1 = Capable |
| | | | 0 | 4 Steps Capable | 1 Bit | | 0 = Not Capable; 1 = Capable |
| 1 | 112 | 0 | 7-0 | Heat Profile - Mon - MOR/OCC1 Start Time | | | Depending on the profile and Step type, DBID 1 shall be populated. |
| | | 1 | 7-0 | Heat Profile - Mon - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 2 | 7-0 | Heat Profile - Mon - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 3 | 7-0 | Heat Profile - Mon - DAY/UNO1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 4 | 7-0 | Heat Profile - Mon - EVE/OCC2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 5 | 7-0 | Heat Profile - Mon - EVE/OCC2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 6 | 7-0 | Heat Profile - Mon - NHT/UNO2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 7 | 7-0 | Heat Profile - Mon - NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 8 | 7-0 | Heat Profile - Tue - MOR/OCC1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 9 | 7-0 | Heat Profile - Tue - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 10 | 7-0 | Heat Profile - Tue - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 11 | 7-0 | Heat Profile - Tue - DAY/UNO1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 12 | 7-0 | Heat Profile - Tue - EVE/OCC2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 13 | 7-0 | Heat Profile - Tue - EVE/OCC2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 14 | 7-0 | Heat Profile - Tue - NHT/UNO2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 15 | 7-0 | Heat Profile - Tue - NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 16 | 7-0 | Heat Profile - Wed - MOR/OCC1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 17 | 7-0 | Heat Profile - Wed - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 18 | 7-0 | Heat Profile - Wed - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 19 | 7-0 | Heat Profile - Wed - DAY/UNO1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 20 | 7-0 | Heat Profile - Wed - EVE/OCC2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 21 | 7-0 | Heat Profile - Wed - EVE/OCC2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 22 | 7-0 | Heat Profile - Wed - NHT/UNO2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 23 | 7-0 | Heat Profile - Wed - NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 24 | 7-0 | Heat Profile - Thur - MOR/OCC1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 25 | 7-0 | Heat Profile - Thur - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 26 | 7-0 | Heat Profile - Thur - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 27 | 7-0 | Heat Profile - Thur - DAY/UNO1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 28 | 7-0 | Heat Profile - Thur - EVE/OCC2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 29 | 7-0 | Heat Profile - Thur - EVE/OCC2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 30 | 7-0 | Heat Profile - Thur - NHT/UNO2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 31 | 7-0 | Heat Profile - Thur - NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 32 | 7-0 | Heat Profile - Fri - MOR/OCC1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 33 | 7-0 | Heat Profile - Fri - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 34 | 7-0 | Heat Profile - Fri - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 35 | 7-0 | Heat Profile - Fri - DAY/UNO1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 36 | 7-0 | Heat Profile - Fri - EVE/OCC2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 37 | 7-0 | Heat Profile - Fri - EVE/OCC2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 38 | 7-0 | Heat Profile - Fri - NHT/UNO2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 39 | 7-0 | Heat Profile - Fri - NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 40 | 7-0 | Heat Profile - W/E 1 - MOR/OCC1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 41 | 7-0 | Heat Profile - W/E 1 - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 42 | 7-0 | Heat Profile - W/E 1 - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 43 | 7-0 | Heat Profile - W/E 1 - DAY/UNO1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 44 | 7-0 | Heat Profile - W/E 1 - EVE/OCC2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 45 | 7-0 | Heat Profile - W/E 1 - EVE/OCC2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 46 | 7-0 | Heat Profile - W/E 1 - NHT/UNO2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 47 | 7-0 | Heat Profile - W/E 1 - NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 48 | 7-0 | Heat Profile - W/E 2 - MOR/OCC1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 49 | 7-0 | Heat Profile - W/E 2 - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 50 | 7-0 | Heat Profile - W/E 2 - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 51 | 7-0 | Heat Profile - W/E 2 - DAY/UNO1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 52 | 7-0 | Heat Profile - W/E 2 - EVE/OCC2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 53 | 7-0 | Heat Profile - W/E 2 - EVE/OCC2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 54 | 7-0 | Heat Profile - W/E 2 - NHT/UNO2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 55 | 7-0 | Heat Profile - W/E 2 -NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 56 | 7-0 | Cool Profile - Mon - MOR/OCC1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 57 | 7-0 | Cool Profile - Mon - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 58 | 7-0 | Cool Profile - Mon - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 59 | 7-0 | Cool Profile - Mon - DAY/UNO1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 60 | 7-0 | Cool Profile - Mon - EVE/OCC2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 61 | 7-0 | Cool Profile - Mon - EVE/OCC2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 62 | 7-0 | Cool Profile - Mon - NHT/UNO2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 63 | 7-0 | Cool Profile - Mon - NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 64 | 7-0 | Cool Profile - Tue - MOR/OCC1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 65 | 7-0 | Cool Profile - Tue - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 66 | 7-0 | Cool Profile - Tue - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 67 | 7-0 | Cool Profile - Tue - DAY/UNO1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 68 | 7-0 | Cool Profile - Tue - EVE/OCC2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 69 | 7-0 | Cool Profile - Tue - EVE/OCC2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 70 | 7-0 | Cool Profile - Tue - NHT/UNO2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 71 | 7-0 | Cool Profile - Tue - NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 72 | 7-0 | Cool Profile - Wed - MOR/OCC1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 73 | 7-0 | Cool Profile - Wed - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 74 | 7-0 | Cool Profile - Wed - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 75 | 7-0 | Cool Profile - Wed - DAY/UNO1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 76 | 7-0 | Cool Profile - Wed - EVE/OCC2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 77 | 7-0 | Cool Profile - Wed - EVE/OCC2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 78 | 7-0 | Cool Profile - Wed - NHT/UNO2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 79 | 7-0 | Cool Profile - Wed - NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 80 | 7-0 | Cool Profile - Thur - MOR/OCC1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 81 | 7-0 | Cool Profile - Thur - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 82 | 7-0 | Cool Profile - Thur - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 83 | 7-0 | Cool Profile - Thur - DAY/UNO1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 84 | 7-0 | Cool Profile - Thur - EVE/OCC2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 85 | 7-0 | Cool Profile - Thur - EVE/OCC2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 86 | 7-0 | Cool Profile - Thur - NHT/UNO2 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 87 | 7-0 | Cool Profile - Thur - NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 88 | 7-0 | Cool Profile - Fri - MOR/OCC1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 89 | 7-0 | Cool Profile - Fri - MOR/OCC1 Temp | | | b7 Prg Fan b6-b0 Set Point |
| | | 90 | 7-0 | Cool Profile - Fri - DAY/UNO1 Start Time | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 91 | 7-0 | Cool Profile - Fri - DAY/UNO1 Temp | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 92 | 7-0 | Cool Profile - Fri - EVE/OCC2 Start Time | | | b7 Prg Fan b6-b0 Set Point |
| | | 93 | 7-0 | Cool Profile - Fri - EVE/OCC2 Temp | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 94 | 7-0 | Cool Profile - Fri - NHT/UNO2 Start Time | | | b7 Prg Fan b6-b0 Set Point |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 95 | 7-0 | Cool Profile - Fri - NHT/UNO2 Temp | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 96 | 7-0 | Cool Profile - W/E 1 - MOR/OCC1 Start Time | | | b7 Prg Fan b6-b0 Set Point |
| | | 97 | 7-0 | Cool Profile - W/E 1 - MOR/OCC1 Temp | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 98 | 7-0 | Cool Profile - W/E 1 - DAY/UNO1 Start Time | | | b7 Prg Fan b6-b0 Set Point |
| | | 99 | 7-0 | Cool Profile - W/E 1 - DAY/UNO1 Temp | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 100 | 7-0 | Cool Profile - W/E 1 - EVE/OCC2 Start Time | | | b7 Prg Fan b6-b0 Set Point |
| | | 101 | 7-0 | Cool Profile - W/E 1 - EVE/OCC2 Temp | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 102 | 7-0 | Cool Profile - W/E 1 - NHT/UNO2 Start Time | | | b7 Prg Fan b6-b0 Set Point |
| | | 103 | 7-0 | Cool Profile - W/E 1 - NHT/UNO2 Temp | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 104 | 7-0 | Cool Profile - W/E 2 - MOR/OCC1 Start Time | | | b7 Prg Fan b6-b0 Set Point |
| | | 105 | 7-0 | Cool Profile - W/E 2 - MOR/OCC1 Temp | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 106 | 7-0 | Cool Profile - W/E 2 - DAY/UNO1 Start Time | | | b7 Prg Fan b6-b0 Set Point |
| | | 107 | 7-0 | Cool Profile - W/E 2 - DAY/UNO1 Temp | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 108 | 7-0 | Cool Profile - W/E 2 - EVE/OCC2 Start Time | | | b7 Prg Fan b6-b0 Set Point |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 109 | 7-0 | Cool Profile - W/E 2 - EVE/OCC2 Temp | | | b7 Damper bit, b6-b2 Hour b1,b0 Min |
| | | 110 | 7-0 | Cool Profile - W/E 2 - NHT/UNO2 Start Time | | | b7 Prg Fan b6-b0 Set Point |
| | | 111 | 7-0 | Cool Profile - W/E 2 - NHT/UNO2 Temp | | | b7 Prg Fan b6-b0 Set Point |

### 7.3.9 Water Heater Configuration Data

Refer to Water Heater Application Profile for the Water Heater configuration data.

## 7.4 Status Message Data Interface (MDI)

### 7.4.1 Thermostat Status Data

**Table 160 – Thermostat Status Data**

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 31 | 0 | 7-0 | Critical Fault | 1 Byte | | 0x00 = No Critical Fault.<br>0x01 to 0xFF = Critical Fault Code (This is OEM Specific) |
| | | 1 | 7-0 | Minor Fault | 1 Byte | | 0x00 = No Minor Fault.<br>0x01 to 0xFF = Minor Fault Code |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | | | | | (This is OEM Specific) |
| | | 2 | 7-0 | System Active Control Status | 1 Byte | | 0 = Off<br>1 = Cool State<br>2 = Auto-Cool State<br>3 = Heat State<br>4 = Auto-Heat State<br>5 = Back-Up (Emergency) Heat State<br>Other = Reserved. |
| | | 3 | 7-0 | Curtailment Active Control Status | 1 Byte | | 0 = No Curtailment<br>1 = DLC Curtailment<br>2 = Tiered Price Protection<br>3 = RTP Price Protection<br>4 = Real Time Pricing<br>Other = Reserved. |
| | | 4 | 7-0 | Humidification Setpoint | 1 Byte | | 0 = Not Enabled or Unavailable;<br>1-100 = %RH. |
| | | 5 | 7-0 | De-humidification Setpoint | 1 Byte | | 0 = Not Enabled or Unavailable;<br>1-100 = %RH. |
| | | 6 | 7-0 | Working Set Point Temperature | 1 Byte | | This is current active setpoint in $^{o}$F. |
| | | 7-8 | 15-12 | Display Temperature | 4 bits | | 0 |
| | | | 11-4 | | 8 bits | | Whole Temperature |
| | | | 3-0 | | 4 bits | | Fractional Part (x/16ths) |
| | | 9 | 7-0 | Heat Set Point Temperature | 1 Byte | | This would be the working setpoint in $^{o}$F if the system mode were set to |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | | | | | heat. |
| | | 10 | 7-0 | Cool Set Point Temperature | 1 Byte | | This would be the working setpoint in °F if the system mode were set to cool. |
| | | 11 | 7-0 | Current Day of Week | 1 Byte | | 0xFF = Unknown or Unavailable<br>Monday = 0x00<br>Tuesday = 0x01<br>Wednesday = 0x02<br>Thursday = 0x03<br>Friday = 0x04<br>Saturday = 0x05<br>Sunday = 0x06. |
| | | 12 | 7-0 | Current Time - Hours | 1 Byte | | 0xFF = Unknown or Unavailable.<br>24 hour clock, where 0x00 = midnight |
| | | 13 | 7-0 | Current Time - Min | 1 Byte | | 0xFF = Unknown or Unavailable.<br>0-59 = Minutes past the hour. |
| | | 14 | 7-0 | Current Time - Sec | 1 Byte | | 0xFF = Unknown or Unavailable.<br>0-59 = Seconds past the minute. |
| | | 15 | 7-4 | Spares | 4 Bits | | 0 |
| | | | 3 | Programmable Hold | 1 Bit | | 0 = Disabled or Unused; 1 = Enabled |
| | | | 2 | Startup Hold | 1 Bit | | 0 = Disabled or Unused; 1 = Enabled |
| | | | 1 | Temporary Hold | 1 Bit | | 0 = Disabled or Unused; 1 = Enabled |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | 0 | Permanent Hold | 1 Bit | | 0 = Disabled or Unused; 1 = Enabled |
| | | 16-17 | 15-0 | Timed Temporary Hold Remaining | 2 Bytes | Minutes | 0xFFFF = Disabled or Unused. 0x0000 to 0xFFFE = Time remaining in minutes. |
| | | 18 | 7-0 | Dehumidification Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 19 | 7-0 | Humidification Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 20 | 7-0 | Heat Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 21 | 7-0 | Cool Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 22 | 7-0 | Fan Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 23 | 7-0 | Emergency Heat Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 24 | 7-0 | Aux Heat Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 25 | 7-0 | Current Time - Year | 1 Byte | | 0xFF= Unknown or Unavailable. 0x00 to 0xFE = Years Offset from 2000 to value in byte. |
| | | 26 | 7-0 | Current Time - Month | 1 Byte | | 0xFF = Unknown or Unavailable. 0x00 = Jan to 0x0B = Dec |
| | | 27 | 7-0 | Current Time - Date | 1 Byte | | 0xFF = Unknown or Unavailable. 1 to 31 (Month Dependent). |
| | | 28 | 7-0 | Relative Humidity Reading | 1 Byte | | 0 = Not Enabled or Unavailable; 1- |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | | | | | 100 = %RH. |
| | | 29 | 7-0 | Away Mode Status | 1 Byte | 0-1 | 0 = Away Mode Disabled or Unavailable;<br>1 = Away Mode Enabled. |
| | | 30 | 7-0 | Fan Mode Setting | 1 Byte | 0-1 | 0 = Auto;<br>1 = Always ON;<br>2 = ON when occupied (Smart Fan Feature) |

### 7.4.2 Furnace Status Data

**Table 161 – Furnace Status Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 22 | 0 | 7-0 | Critical Fault | 1 Byte | | 0x00 = No Critical Fault.<br>0x01 to 0xFF = Critical Fault Code (This is OEM Specific) |
| | | 1 | 7-0 | Minor Fault | 1 Byte | | 0x00 = No Minor Fault.<br>0x01 to 0xFF = Minor Fault Code (This is OEM Specific) |
| | | 2 | 7-0 | Heat Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 3 | 7-0 | Cool Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 4 | 7-0 | Fan Requested Mode | 1 Byte | 0 - 5 | Refer to Fan Demand Control Command for values. |
| | | 5 | 7-0 | Fan Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 6 | 7-0 | Fan Requested Rate/Slew | 1 Byte | | 0x00 = Slew as fast as possible or Unavailable.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |
| | | 7 | 7-0 | Fan Requested Delay | 1 Byte | | 0x00 = Zero delay or Unavailable.<br>0x01 to 0xFF = Number of Seconds to wait before turning ON the Fan. |
| | | 8 | 7-0 | Defrost Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 9 | 7-0 | Emergency Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 10 | 7-0 | Aux Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 11 | 7-0 | Humidification Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 12 | 7-0 | Dehumidification Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 13-14 | 15-0 | Current Airflow | 2 Bytes | | 0x0000 = Zero CFM or Unavailable<br>0x0001 to 0xFFFF = Motor CFM Reported or Estimated. |
| | | 15 | 7-0 | Current Heat Actual Status | 1 Byte | 0 - 100% | % Based on Configuration Capabilities<br>(Relays On/Total # Relays) Percentage - 0.5% Increments |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 16 | 7-0 | Current Cool Actual Status | 1 Byte | 0 - 100% | % Based on Configuration Capabilities<br>(Relays On/Total # Relays)<br>Percentage - 0.5% Increments |
| | | 17 | 7-0 | Current Fan Actual Status | 1 Byte | 0 - 100% | % Based on Configuration Capabilities<br>(Relays On/Total # Relays)<br>Percentage - 0.5% Increments |
| | | 18 | 7-0 | Fan Current Rate Status | 1 Byte | | 0x00 = Slew as fast as possible or Unavailable.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |
| | | 19 | 7-0 | Fan Current Delay Remaining Status | 1 Byte | | 0x00 = Zero delay remaining or Unavailable.<br>0x01 to 0xFF = Number of Seconds remaining to wait before turning ON the Fan. |
| | | 20 | 7-0 | Current Humidification Actual Status | 1 Byte | 0 - 100% | Percentage - 0.5% Increments |
| | | 21 | 7-0 | Current Dehumidification Actual Status | 1 Byte | 0 - 100% | Percentage - 0.5% Increments |

### 7.4.3 Air Handler Status Data

**Table 162 – Air Handler Status data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 20 | 0 | 7-0 | Critical Fault | 1 Byte | | 0x00 = No Critical Fault.<br>0x01 to 0xFF = Critical Fault Code (This is OEM Specific) |
| | | 1 | 7-0 | Minor Fault | 1 Byte | | 0x00 = No Minor Fault.<br>0x01 to 0xFF = Minor Fault Code (This is OEM Specific) |
| | | 2 | 7-0 | Heat Requested Demand | 1 Byte | | Demand Percentage - 0.5% Increments |
| | | 3 | 7-0 | Fan Requested Mode | 1 Byte | | Refer to Fan Demand Control Command for values. |
| | | 4 | 7-0 | Fan Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 5 | 7-0 | Fan Requested Rate/Slew | 1 Byte | | 0x00 = Slew as fast as possible or Unavailable.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |
| | | 6 | 7-0 | Fan Requested Delay | 1 Byte | | 0x00 = Zero delay or Unavailable.<br>0x01 to 0xFF = Number of Seconds to wait before turning ON the Fan. |
| | | 7 | 7-0 | Defrost Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 8 | 7-0 | Emergency Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 9 | 7-0 | Aux Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 10 | 7-0 | Humidification Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 11 | 7-0 | Dehumidification Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 12-13 | 15-0 | Current Airflow | 2 Bytes | | 0x0000 = Zero CFM or Unavailable<br>0x0001 to 0xFFFF = Motor CFM Reported or Estimated. |
| | | 14 | 7-0 | Current Heat Actual Status | 1 Byte | 0 - 100% | % Based on Configuration Capabilities (Relays On/Total # Relays)<br>Demand Percentage - 0.5% Increments |
| | | 15 | 7-0 | Current Fan Actual Status | 1 Byte | 0 - 100% | % Based on Configuration Capabilities (Relays On/Total # Relays)<br>Demand Percentage - 0.5% Increments |
| | | 16 | 7-0 | Fan Current Rate Status | 1 Byte | | 0x00 = Slew as fast as possible or Unavailable.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |
| | | 17 | 7-0 | Fan Current Delay Remaining Status | 1 Byte | | 0x00 = Zero delay remaining or Unavailable.<br>0x01 to 0xFF = Number of Seconds remaining to wait before turning ON the Fan. |
| | | 18 | 7-0 | Current Humidification Actual Status | 1 Byte | | Percentage - 0.5% Increments |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 19 | 7-0 | Current Dehumidification Actual Status | 1 Byte | | Percentage - 0.5% Increments |

### 7.4.4 Air Conditioner Status Data

**Table 163 – Air Conditioner Status Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 9 | 0 | 7-0 | Critical Fault | 1 Byte | | 0x00 = No Critical Fault.<br>0x01 to 0xFF = Critical Fault Code (This is OEM Specific) |
| | | 1 | 7-0 | Minor Fault | 1 Byte | | 0x00 = No Minor Fault.<br>0x01 to 0xFF = Minor Fault Code (This is OEM Specific) |
| | | 2 | 7-0 | Cool Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 3 | 7-0 | Dehumidification Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 4 | 7-0 | Current Cool Actual Status | 1 Byte | 0 - 100% | % Based on Configuration Capabilities (Relays On/Total # Relays)<br>Percentage - 0.5% Increments |
| | | 5 | 7-0 | Fan Requested Demand | 1 Byte | 0 - 100% | Current Fan Demand to Indoor Unit.<br>Demand Percentage - 0.5% Increments |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | 6 | 7-0 | Fan Requested Rate | 1 Byte | | Current Fan Request Rate to Indoor Unit.<br>0x00 = Slew as fast as possible or Unused.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |
| | | 7 | 7-0 | Fan Requested Delay | 1 Byte | | Current Fan Request Delay to Indoor Unit.<br>0x00 = Zero delay or Unused.<br>0x01 to 0xFF = Number of Seconds to wait before turning ON the Fan. |
| | | 8 | 7-0 | Current Dehumidification Actual Status | 1 Byte | 0 - 100% | Percentage - 0.5% Increments |

### 7.4.5 Heat Pump Status Data

**Table 164 – Heat Pump Status Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 12 | 0 | 7-0 | Critical Fault | 1 Byte | | 0x00 = No Critical Fault.<br>0x01 to 0xFF = Critical Fault Code (This is OEM Specific) |
| | | 1 | 7-0 | Minor Fault | 1 Byte | | 0x00 = No Minor Fault.<br>0x01 to 0xFF = Minor Fault Code (This is OEM Specific) |
| | | 2 | 7-0 | Heat Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 3 | 7-0 | Cool Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 4 | 7-0 | Dehumidification Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 5 | 7-0 | Heat Current Demand Status | 1 Byte | 0 - 100% | % Based on Configuration Capabilities (Relays On/Total # Relays) Percentage - 0.5% Increments |
| | | 6 | 7-0 | Cool Current Demand Status | 1 Byte | 0 - 100% | % Based on Configuration Capabilities (Relays On/Total # Relays) Percentage - 0.5% Increments |
| | | 7 | 7-0 | Defrost Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 8 | 7-0 | Fan Requested Demand | 1 Byte | 0 - 100% | Current Fan Demand to Indoor Unit. Demand Percentage - 0.5% Increments |
| | | 9 | 7-0 | Fan Requested Rate | 1 Byte | | Current Fan Request Rate to Indoor Unit.<br>0x00 = Slew as fast as possible or Unused.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |
| | | 10 | 7-0 | Fan Requested Delay | 1 Byte | | Current Fan Request Delay to Indoor Unit.<br>0x00 = Zero delay or Unused.<br>0x01 to 0xFF = Number of Seconds to wait before turning ON the Fan. |
| | | 11 | 7-0 | Current Dehumidification Actual Status | 1 Byte | 0 - 100% | Percentage - 0.5% Increments |

### 7.4.6 Crossover Status Data

**Table 165 – Crossover Status Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 23 | 0 | 7-0 | Critical Fault | 1 Byte | | 0x00 = No Critical Fault.<br>0x01 to 0xFF = Critical Fault Code (This is OEM Specific) |
| | | 1 | 7-0 | Minor Fault | 1 Byte | | 0x00 = No Minor Fault.<br>0x01 to 0xFF = Minor Fault Code (This is OEM Specific) |
| | | 2 | 7-0 | Heat Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 3 | 7-0 | Cool Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 4 | 7-0 | Fan Requested Mode | 1 Byte | 0-5 | Current Fan Request Mode accepted if emulating Indoor Unit; If emulating Outdoor Unit, this value is 0x00.<br>Refer to Fan Demand Control Command for values. |
| | | 5 | 7-0 | Fan Requested Demand | 1 Byte | 0 - 100% | Current Fan Request Demand accepted if emulating Indoor Unit; If emulating Outdoor Unit, this value is 0x00.<br>Demand Percentage - 0.5% Increments |
| | | 6 | 7-0 | Fan Requested Rate | 1 Byte | 0 - 100% | Current Fan Request Rate accepted if emulating Indoor Unit; If emulating Outdoor Unit, this value is 0x00.<br>0x00 = Slew as fast as possible or Unused.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | 7 | 7-0 | Defrost Requested Demand | 1 Byte | 0 - 100% | Output Mode - Demand Percentage - 0.5% Increments<br>Input Mode - % Based on Configuration Capabilities<br>Relays On/Total # Stages) |
| | | 8 | 7-0 | Dehumidification Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 9 | 7-0 | Humidification Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 10 | 7-0 | Emergency Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 11 | 7-0 | Aux Requested Demand | 1 Byte | 0 - 100% | Demand Percentage - 0.5% Increments |
| | | 12 | 7-0 | Current Indoor Heat Actual Status | 1 Byte | 0 - 100% | % Based Relays On/Total # Relays Percentage - 0.5% Increments |
| | | 13 | 7-0 | Current Outdoor Heat Actual Status | 1 Byte | 0 - 100% | % Based Relays On/Total # Relays Percentage - 0.5% Increments |
| | | 14 | 7-0 | Current Cool Actual Status | 1 Byte | 0 - 100% | % Based Relays On/Total # Relays Percentage - 0.5% Increments |
| | | 15 | 7-0 | Current Fan Actual Status | 1 Byte | 0 - 100% | % Based Relays On/Total # Relays Percentage - 0.5% Increments |
| | | 16 | 7-0 | Current Fan Rate Actual Status | 1 Byte | | 0x00 = Slew as fast as possible or Unavailable.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |
| | | 17 | 7-0 | Current Fan Delay Actual Status | 1 Byte | | 0x00 = Zero delay remaining or Unavailable.<br>0x01 to 0xFF = Number of Seconds |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| | | | | | | | remaining to wait before turning ON the Fan. |
| | | 18 | 7-0 | Current Humidification Actual Status | 1 Byte | | % Based Relays On/Total # Relays Percentage - 0.5% Increments |
| | | 19 | 7-0 | Current Dehumidification Actual Status | 1 Byte | | % Based Relays On/Total # Relays Percentage - 0.5% Increments |
| | | 20 | 7-0 | Defrost Requested Demand | 1 Byte | 0 - 100% | The % defrost heat demand sent to the Indoor Unit when emulating an Outdoor Unit.<br>If emulating an Indoor Unit, this is 0x00<br>Demand Percentage - 0.5% Increments |
| | | 21 | 7-0 | Fan Requested Demand | 1 Byte | 0 - 100% | The % Fan Demand sent to the Indoor Unit when emulating an Outdoor Unit.<br>If emulating an Indoor Unit, this is 0x00<br>Demand Percentage - 0.5% Increments |
| | | 22 | 7-0 | Fan Requested Rate | 1 Byte | | Current Fan Request Rate to Indoor Unit when emulating Outdoor Unit.<br>0x00 = Slew as fast as possible or Unused or when emulating Indoor Unit.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |

### 7.4.7 Zone Controller Status Data

**Table 166 – Zone Controller Status Data**

| DB ID TAG | DB ID LENGTH | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| 0 | 20 | 0 | 7-0 | Critical Fault | | 0x00 = No Critical Fault.<br>0x01 to 0xFF = Critical Fault Code (This is OEM Specific) |
| | | 1 | 7-0 | Minor Fault | | 0x00 = No Minor Fault.<br>0x01 to 0xFF = Minor Fault Code (This is OEM Specific) |
| | | 7 | 7-0 | Heat Requested Demand | | Demand Percentage - 0.5% Increments |
| | | 9 | 7-0 | Cool Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 10 | 7-0 | Fan Requested Mode | | 0 = Manual Fan Mode;<br>Refer to Fan Demand Control Command for values. |
| | | 11 | 7-0 | Fan Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 12 | 7-0 | Fan Requested Rate/Slew | | Current Fan Request Rate to Indoor Unit.<br>0x00 = Slew as fast as possible or Unused.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |
| | | 13 | 7-0 | Fan Requested Delay | | Current Fan Request Delay to Indoor Unit.<br>0x00 = Zero delay or Unused.<br>0x01 to 0xFF = Number of Seconds to wait before turning ON the Fan. |
| | | 14 | 7-0 | Emergency Heat Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 15 | 7-0 | Aux Heat Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 16 | 7-0 | Humidification Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |

| DB ID TAG | DB ID LENGTH | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| | | 17 | 7-0 | Dehumidification Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 18 | | System Active Control Status | | 0 = Off<br>1 = Cool State<br>2 = Auto-Cool State<br>3 = Heat State<br>4 = Auto-Heat State<br>5 = Back-Up (Emergency) Heat State<br>Other = Reserved. |
| | | 19 | 7 | Freeze Fault Active | | 0 = No Fault; 1 = Frozen Detected |
| | | | 6 | Over Heat Fault Active | | 0 = No Fault; 1 = Over Heat Detected |
| | | | 5-0 | Spares | | 0 |

### 7.4.8 Zone User Interface Status Data

**Table 167 – Zone User Interface Status Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| 0 | 26 | 0 | 7-0 | Critical Fault | | 0x00 = No Critical Fault.<br>0x01 to 0xFF = Critical Fault Code (This is OEM Specific) |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| | | 1 | 7-0 | Minor Fault | | 0x00 = No Minor Fault.<br>0x01 to 0xFF = Minor Fault Code (This is OEM Specific) |
| | | 2 | 7-0 | Working Set Point Temperature | | This is current active setpoint in °F. |
| | | 3 | 7-0 | Heat Set Point Temperature | | This would be the working setpoint in °F if the system mode were set to heat. |
| | | 4 | 7-0 | Cool Set Point Temperature | | This would be the working setpoint in °F if the system mode were set to cool. |
| | | 5 | 7-0 | Humidification Setpoint | | 0 = Not Enabled or Unavailable; 1-100 = %RH. |
| | | 6 | 7-0 | De-humidification Setpoint | | 0 = Not Enabled or Unavailable; 1-100 = %RH. |
| | | 7 | 7-0 | Heat Requested Demand | | Demand Percentage - 0.5% Increments |
| | | 9 | 7-0 | Cool Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 10 | 7-0 | Fan Requested Mode | | Fan Mode of Operation = Manual Mode (0) |
| | | 11 | 7-0 | Fan Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 12 | 7-0 | Fan Requested Rate/Slew | | Current Fan Request Rate to Zone Controller.<br>0x00 = Slew as fast as possible or Unused.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |
| | | 13 | 7-0 | Fan Requested Delay | | Current Fan Request Delay to Zone Controller.<br>0x00 = Zero delay or Unused.<br>0x01 to 0xFF = Number of Seconds to wait |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| | | | | | | before turning ON the Fan. |
| | | 14 | 7-0 | Emergency Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 16 | 7-0 | Humidification Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 17 | 7-0 | Dehumidification Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 18 | 7-0 | System Active Control Status | | 0 = Off<br>1 = Cool State<br>2 = Auto-Cool State<br>3 = Heat State<br>4 = Auto-Heat State<br>5 = Back-Up (Emergency) Heat State<br>Other = Reserved. |
| | | 19 | 7-0 | Current Day of Week | | 0xFF = Unknown or Unavailable<br>Monday = 0x00<br>Tuesday = 0x01<br>Wednesday = 0x02<br>Thursday = 0x03<br>Friday = 0x04<br>Saturday = 0x05<br>Sunday = 0x06. |
| | | 20 | 7-0 | Current Time - Hours | | 0xFF = Unknown or Unavailable.<br>24 hour clock, where 0x00 = midnight |
| | | 21 | 7-0 | Current Time - Min | | 0xFF = Unknown or Unavailable.<br>0-59 = Minutes past the hour. |
| | | 22 | 7-0 | Current Time - Sec | | 0xFF = Unknown or Unavailable.<br>0-59 = Seconds past the minute. |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| | | 23 | 7-0 | Current Time - Year | | 0xFF= Unknown or Unavailable.<br>0x00 to 0xFE = Years Offset from 2000 to value in byte. |
| | | 24 | 7-0 | Current Time - Month | | 0xFF = Unknown or Unavailable.<br>0x00 = Jan to 0x0B = Dec |
| | | 25 | 7-0 | Current Time - Date | | 0xFF = Unknown or Unavailable.<br>1 to 31 (Month Dependent). |

### 7.4.9 Zone Temperature Controller Status Data

**Table 168 – Zone Temperature Controller Status Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| 0 | 26 | 0 | 7-0 | Critical Fault | | 0x00 = No Critical Fault.<br>0x01 to 0xFF = Critical Fault Code (This is OEM Specific) |
| | | 1 | 7-0 | Minor Fault | | 0x00 = No Minor Fault.<br>0x01 to 0xFF = Minor Fault Code (This is OEM Specific) |
| | | 2 | 7-0 | Working Set Point Temperature | | This is current active setpoint in °F |
| | | 3 | 7-0 | Heat Set Point Temperature | | This would be the working setpoint in °F if the system mode were set to heat. |
| | | 4 | 7-0 | Cool Set Point Temperature | | This would be the working setpoint in °F if the system mode were set to cool. |
| | | 5 | 7-0 | Humidification Setpoint | | 0 = Not Enabled or Unavailable; 1-100 = %RH. |
| | | 6 | 7-0 | De-humidification Setpoint | | 0 = Not Enabled or Unavailable; 1-100 = %RH. |
| | | 7 | 7-0 | Heat Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 9 | 7-0 | Cool Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 10 | 7-0 | Fan Requested Mode | | Fan Mode of Operation = Manual Mode (0) |
| | | 11 | 7-0 | Fan Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| | | 12 | 7-0 | Fan Requested Rate/Slew | | Current Fan Request Rate to Zone Controller.<br>0x00 = Slew as fast as possible or Unused.<br>0x01 to 0xFF = Number of Seconds to Slew from 0 to maximum demand. |
| | | 13 | 7-0 | Fan Requested Delay | | Current Fan Request Delay to Zone Controller.<br>0x00 = Zero delay or Unused.<br>0x01 to 0xFF = Number of Seconds to wait before turning ON the Fan. |
| | | 14 | 7-0 | Emergency Heat Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 16 | 7-0 | Humidification Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 17 | 7-0 | Dehumidification Requested Demand | 0 - 200 | Demand Percentage - 0.5% Increments |
| | | 18 | 7-0 | System Active Control Status | | 0 = Off<br>1 = Cool State<br>2 = Auto-Cool State<br>3 = Heat State<br>4 = Auto-Heat State<br>5 = Back-Up (Emergency) Heat State<br>Other = Reserved |
| | | 19 | 7-0 | Current Day of Week | | 0xFF = Unknown or Unavailable<br>Monday = 0x00<br>Tuesday = 0x01<br>Wednesday = 0x02<br>Thursday = 0x03<br>Friday = 0x04<br>Saturday = 0x05 |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| | | | | | | Sunday = 0x06 |
| | | 20 | 7-0 | Current Time - Hours | | 0xFF = Unknown or Unavailable.<br>24 hour clock, where 0x00 = midnight |
| | | 21 | 7-0 | Current Time - Min | | 0xFF = Unknown or Unavailable.<br>0-59 = Minutes past the hour. |
| | | 22 | 7-0 | Current Time - Sec | | 0xFF = Unknown or Unavailable.<br>0-59 = Seconds past the minute. |
| | | 23 | 7-0 | Current Time - Year | | 0xFF= Unknown or Unavailable.<br>0x00 to 0xFE = Years Offset from 2000 to value in byte. |
| | | 24 | 7-0 | Current Time - Month | | 0xFF = Unknown or Unavailable.<br>0x00 = Jan to 0x0B = Dec |
| | | 25 | 7-0 | Current Time - Date | | 0xFF = Unknown or Unavailable.<br>1 to 31 (Month Dependent). |

### 7.4.10Zone Damper Status Data

**Table 169 – Zone Damper Status Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| 0 | 4 | 0 | 7-0 | Critical Fault | | 0x00 = No Critical Fault.<br>0x01 to 0xFF = Critical Fault Code (This is OEM Specific) |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| | | 1 | 7-0 | Minor Fault | | 0x00 = No Minor Fault.<br>0x01 to 0xFF = Minor Fault Code (This is OEM Specific) |
| | | 2 | 7-0 | Requested Damper Opening Position | 0-200 | 0% = Closed to 100% = Open Fully<br>Demand Percentage - 0.5% Steps |
| | | 3 | 7-0 | Actual Damper Opening Position | 0-200 | 0% = Closed to 100% = Open Fully<br>Demand Percentage - 0.5% Steps |

### 7.4.11 Occupancy Sensor Status Data

**Table 170 – Occupancy Sensor Status Data**

| DB ID Tag | DB ID Length | Byte | Bit | Description | Range | Notes |
|---|---|---|---|---|---|---|
| 0 | 3 | 0 | 7-0 | Critical Fault | | 0x00 = No Critical Fault.<br>0x01 to 0xFF = Critical Fault Code (This is OEM Specific) |
| | | 1 | 7-0 | Minor Fault | | 0x00 = No Minor Fault.<br>0x01 to 0xFF = Minor Fault Code (This is OEM Specific) |
| | | 2 | 7-0 | Occupancy Detected | 0-1 | 0 = No Occupancy<br>1 = Actively Occupied<br>Other values are Reserved. |

### 7.4.12 Water Heater Status Data

Refer to *Water Heater Application Profile* for Water Heater Status Data.

## 7.5 Sensor Message Data Interface (MDI)

### 7.5.1 Furnace Sensor Data

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0-1 | 15 | Return Air Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |
| 1 | 2 | 2-3 | 15 | Supply Air Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| | | | | | | | 0 to 15 in 1/16th increments. |

### 7.5.2 Air Handler Sensor Data

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0-1 | 15 | Return Air Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |
| 1 | 2 | 2-3 | 15 | Supply Air Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |

### 7.5.3 Air Conditioner Sensor Data

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0-1 | 15 | Outdoor Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 13-4 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 3-0 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |

### 7.5.4 Heat Pump Sensor Data

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0-1 | 15 | Outdoor Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 13-4 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 3-0 | | 4 bits | Fractional Part | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |

### 7.5.5 Crossover Sensor Data

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0-1 | 15 | Outdoor Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |
| 1 | 2 | 0-1 | 15 | Return Air Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |
| 2 | 2 | 0-1 | 15 | Supply Air Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F |

| DB ID TAG | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| | | | | | | | 0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |

### 7.5.6 Zone User Interface Sensor Data

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0-1 | 15 | Local Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |
| 1 | 2 | 0-1 | 15 | Relative Humidity Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Reserved | 0 = Reserved |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the RH in %<br>0 to 100 |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| | | | | | | | Other Value are Reserved. |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the RH in %<br>0 to 15 in 1/16th increments. |

### 7.5.7 Zone Temperature Controller Sensor Data

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Range | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0-1 | 15 | Local Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |
| 1 | 2 | 0-1 | 15 | Relative Humidity Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Reserved | 0 = Reserved |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the RH in %<br>0 to 100<br>Other Value are Reserved. |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the RH in % |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | 0 to 15 in 1/16th increments. |

### 7.5.8 Supply Air Temperature (SAT) Sensor Data

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0-1 | 15 | Supply Air Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |

### 7.5.9 Return Air Temperature (RAT) Sensor Data

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0-1 | 15 | Return Air Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |

### 7.5.10 Outside Air Temperature (OAT) Sensor Data

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0-1 | 15 | Outside Air Temperature | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |

### 7.5.11 Remote Temperature Sensor Data

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0-1 | 15 | Remote Temperature Sensor | 1 bit | Validity Flag | 1 = Valid 0 = Invalid or Not Installed. |
| | | | 14 | | 1 bit | Signed Bit | 1 = Negative 0 = Positive |

| DB ID Tag | DB ID Length | Byte | Bit | Description | Size | Element | Notes |
|---|---|---|---|---|---|---|---|
| | | | 4-13 | | 10 bits | Whole Value | Absolute value of the integer part of the temperature in °F<br>0 to 1023 |
| | | | 0-3 | | 4 bits | Fractional Part (x/16ths) | Absolute value of the fractional part of the temperature in °F<br>0 to 15 in 1/16th increments. |

# 8.0 Annex A – Message Utilization

The following sections shall help identify the CIM implementation via the many different profiles developed. Each profile that implements the CIM would have an appropriate section to help define how the CIM is utilized.

## 8.1 Command Reference Utilization

Not all command reference messages are required by every profile. Profiles define which message types are required.

### 8.1.1 Generic Application Message Utilization

The following table identifies the Generic Application messages utilized as either optional/mandatory message to be used by CIM subsystems.

**Table 171 – Generic Application CMD Reference Utilization**

| Message Name |
|---|
| Set Identification |
| Get Identification |
| Set Network Node List |

#### 8.1.1.1 Generic Application Received Messages

The tables below define the message types each subsystem shall support. Subsystems shall support the messages shaded in green; subsystems may support additional messages provided the subsystem is capable of receiving or transmitting that message type.

An "X" means that there are requirements defined for that message type in the specification. A shaded green cell means that supporting that message type is mandatory. An empty cell means that there are no requirements currently defined.

**Table 172 – Generic Profile Application Received Messages**

| Message Name | Subsystem |
|---|---|
| | Generic |
| Set Identification | X |
| Get Identification | X |
| Set Network Node List | X |

#### 8.1.1.2 Generic Application Transmitted Messages

Currently, there are no messages defined to be optional/mandatory transmitted messages by a generic device.

### 8.1.2 HVAC Profile Message Utilization

The following table identifies the HVAC profile messages utilized as either optional/mandatory message to be used by CIM subsystems.

**Table 173 – HVAC Profile Message Utilization**

| Message Name | Message Name |
|---|---|
| Get Configuration | Set Application Shared Data to Network |
| Get Status | Get Application Shared Data from Network |
| Set Control Command | Set Manufacturer Device Data |
| Set Display Message | Get Manufacturer Device Data |
| Set Diagnostics | Set Manufacturer Generic Data |
| Get Diagnostics | Get Manufacturer Generic Data |
| Get Sensor Data | Get User Menu |
| Set Network Node List | Set User Menu Update |
| Set Identification | Set Factory Shared Data to Application |
| Get Identification | Get Shared Data from Application |
| | Set Echo Data |

#### 8.1.2.1 HVAC Profile Received Message Types

The tables below define the message types each subsystem shall support. Subsystems shall support the messages shaded in green; subsystems may support additional messages provided the subsystem is capable of receiving or transmitting that message type.

An "X" means that there are requirements defined for that message type in the specification. A shaded green cell means that supporting that message type is mandatory. An empty cell means that there are no requirements currently defined.

**Table 174 – HVAC Profile Application Received Messages**

| Message Name | Subsystem | | | | | |
|---|---|---|---|---|---|---|
| | TSTAT | IFC | AH | AC | HP | XOVER |
| Get Configuration | X | X | X | X | X | X |
| Get Status | X | X | X | X | X | X |
| Set Control Command | X | X | X | X | X | X |
| Set Display Message | X | | | | | |
| Set Diagnostics | X | | | | | |
| Get Sensor Data | X | X | X | X | X | X |
| Set Identification | X | X | X | X | X | X |
| Get Identification | X | X | X | X | X | X |
| Set Manufacturer Device Data | X | X | X | X | X | X |
| Get Manufacturer Device Data | X | X | X | X | X | X |
| Set Network Node List | X | X | X | X | X | X |
| Get User Menu | X[9] | X | X | X | X | X |
| Set User Menu Update | X[9] | X | X | X | X | X |
| Set Factory Shared Data to Application | X | X | X | X | X | X |
| Get Shared Data from Application | X | X | X | X | X | X |
| Set Echo Data | X | X | X | X | X | X |

[9] Non-graphical thermostats could be capable of displaying user menu information.

#### 8.1.2.2 HVAC Profile Transmitted Message Types

The following table will define the additions or deletions to subsystems that exist in the CIM HVAC Profile.

An "X" means that there are requirements defined for that message type in the specification. An empty cell means that there are no requirements currently defined.

**Table 175 – HVAC Profile Application Transmitted Message Types**

| Message Name | Subsystem | | | | | |
|---|---|---|---|---|---|---|
| | TSTAT | IFC | AH | AC | HP | XOVER |
| Get Configuration | X | | | X | X | X |
| Get Status | X | | | X | X | X |
| Set Control Command | X | | | X | X | X |
| Set Display Message | | X | X | X | X | X |
| Set Diagnostics | | X | X | X | X | X |
| Get Diagnostics | X | | | | | |
| Get Sensor Data | X | | | | | |
| Set Identification | X | X | X | X | X | X |
| Set Application Shared Data to Network | X | X | X | X | X | X |
| Get Application Shared Data from Network | X | X | X | X | X | X |
| Get User Menu | X | | | | | |
| Set User Menu Update | X | | | | | |
| Set Echo Data | X | X | X | X | X | X |

### 8.1.3 Zoning Profile Message Utilization

The following table identifies the Zoning profile messages utilized as either optional/ mandatory message to be used by CIM subsystems.

**Table 176 – Zoning Profile Message Utilization**

| Message Name |
|---|
| Get Configuration |
| Get Status |
| Set Control Command |
| Set Display Message |
| Set Diagnostics |
| Get Diagnostics |
| Get Sensor Data |
| Set Network Node List |
| Set Identification |
| Get Identification |
| Set Application Shared Data to Network |
| Get Application Shared Data from Network |
| Set Manufacturer Device Data |
| Get Manufacturer Device Data |
| Set Manufacturer Generic Data |
| Get Manufacturer Generic Data |
| Get User Menu |
| Set User Menu Update |
| Set Factory Shared Data to Application |
| Get Shared Data from Application |
| Set Echo Data |

#### 8.1.3.1 Zoning Profile Received Message Types

The tables below define the message types each subsystem shall support. Subsystems shall support the messages shaded in green; subsystems may support additional messages provided the subsystem is capable of receiving or transmitting that message type.

An "X" means that there are requirements defined for that message type in the specification. A shaded green cell means that supporting that message type is mandatory. An empty cell means that there are no requirements currently defined.

**Table 177 – Zoning Profile Application Received Message Types**

| Message Name | Subsystem | | | | | |
|---|---|---|---|---|---|---|
| | ZCTRL | ZUI | ZTC | ZDAMP | Occupancy Sensor | Temp Sensor |
| Get Configuration | X | X | | | | |
| Get Status | X | X | X | X | | |
| Set Control Command | X | | | X | | |
| Set Display Message | X | X | X | | | |
| Set Diagnostics | X | | | | | |
| Get Diagnostics | | | | | | |
| Get Sensor Data | | X | X | | X | X |
| Get Event Data | | | | | | |
| Set Identification | X | X | X | X | X | X |
| Get Identification | X | X | X | X | X | X |
| Set Manufacturer Device Data | | | | | | |
| Get Manufacturer Device Data | | | | | | |
| Set Network Node List | X | X | X | X | X | X |
| Get User Menu | | X | X | X | X | X |
| Set User Menu Update | | X | X | X | X | X |
| Set Factory Shared Data to Application | X | | | | | |
| Get Shared Data from Application | X | | | | | |
| Set Echo Data | X | | | | | |

#### 8.1.3.2 Zoning Profile Transmitted Message Types

The following table will define the additions or deletions to subsystems that exist in the CIM HVAC Profile.

An "X" means that there are requirements defined for that message type in the specification. An empty cell means that there are no requirements currently defined.

**Table 178 – Zoning Profile Application Transmitted Message Types**

| Message Name | Subsystem | | | | | |
|---|---|---|---|---|---|---|
| | ZCTRL | ZUI | ZTC | ZDAMP | Occupancy Sensor | Temp Sensor |
| Get Configuration | X | | | | | |
| Get Status | X | | | | | |
| Set Control Command | X | | | | | |
| Set Display Message | | | | | | |
| Set Diagnostics | | X | X | | | |
| Get Diagnostics | | | | | | |
| Get Sensor Data | | | | | | |
| Get Event Data | | | | | | |
| Set Identification | X | | | | | |
| Get Identification | X | | | | | |
| Set Application Shared Data to Network | | | | | | |
| Get Application Shared Data from Network | | | | | | |
| Set Manufacturer Device Data | | | | | | |
| Get Manufacturer Device Data | | | | | | |
| Set Network Node List | | | | | | |
| Get User Menu | X[9] | X | | | | |
| Set User Menu Update | X[9] | X | | | | |

### 8.1.4 Motor Profile Message Utilization

The following table identifies the Motor profile messages utilized as either optional/mandatory message to be used by CIM subsystems.

**Table 179 – Motor Profile Message Utilization**

| Message Name |
|---|
| Get Status |
| Control Command |
| Get Sensor Data |
| Get Identification Data |
| Direct Memory Access Read |
| Direct Memory Access Write |
| Set Manufacturer Generic Data |
| Get Manufacturer Generic Data |
| Manufacturer Generic Reply |

### 8.1.5 Water Heater Profile Message Utilization

Refer to the Water Heater Profile for utilized commands.

# 9.0 Annex B – Node Type Table

**Table 180 – Node Types**

| Node Type Code (Hex) | Node Type Code (Dec) | Node Type |
|---|---|---|
| 0x01 | 1 | Thermostat |
| 0x02 | 2 | Gas Furnace |
| 0x03 | 3 | Air Handler |
| 0x04 | 4 | Air Conditioner |
| 0x05 | 5 | Heat Pump |
| 0x06 | 6 | Electric Furnace |
| 0x07 | 7 | Package System - Gas |
| 0x08 | 8 | Package System - Electric |
| 0x09 | 9 | Crossover (aka OBBI) |
| 0x0A | 10 | Secondary Compressor |
| 0x0B | 11 | Air Exchanger |
| 0x0C | 12 | Unitary Control |
| 0x0D | 13 | Dehumidifier |
| 0x0E | 14 | Electronic Air Cleaner |
| 0x0F | 15 | ERV |
| 0x10 | 16 | Humidifier (Evaporative) |
| 0x11 | 17 | Humidifier (Steam) |
| 0x12 | 18 | HRV |
| 0x13 | 19 | IAQ Analyzer |
| 0x14 | 20 | Media Air Cleaner |
| 0x15 | 21 | Zone Control |
| 0x16 | 22 | Zone User Interface |
| 0x17 | 23 | Boiler |
| 0x18 | 24 | Water Heater – Gas |

| Node Type Code (Hex) | Node Type Code (Dec) | Node Type |
|---|---|---|
| 0x19 | 25 | Water Heater – Electric |
| 0x1A | 26 | Water Heater - Commercial |
| 0x1B | 27 | Pool Heater |
| 0x1C | 28 | Ceiling Fan |
| 0x1D | 29 | Gateway |
| 0x1E | 30 | Diagnostic Device |
| 0x1F | 31 | Lighting Control |
| 0x20 | 32 | Security System |
| 0x21 | 33 | UV Light |
| 0x22 | 34 | Weather Data Device |
| 0x23 | 35 | Whole House Fan |
| 0x24 | 36 | Solar Inverter |
| 0x25 | 37 | Zone Damper |
| 0x26 | 38 | Zone Temperature Control (ZTC) |
| 0x27 | 39 | Temperature Sensor |
| 0x28 | 40 | Occupancy Sensor |
| *0x29 - 0xA4* | *41-164* | *Reserved* |
| 0xA5 | 165 | Network Coordinator |
| *0xA6-0xEF* | *166-239* | *Reserved* |
| *0xF0-0xFF* | *240-255* | *Reserved* |

# 10.0 Annex C – Bibliography

"TIA-485 (Revision A), Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems" *Telecommunications Industry Association*, 1998.

Zimmermann, Hubert (April 1980). "OSI Reference Model — The ISO Model of Architecture for Open Systems Interconnection". *IEEE Transactions on Communications*