Domain "ule.hanfun.interfaces"

ModuleClasses

0x100 - Alert Server Interface

The *Alert Server* interface can be used by any device that requires sending an alert. The alert type will be specified by the unit type where the interface is implemented, for example for a smoke detector the alert will indicate the presence of smoke, but for a motion detector the same alert will indicate that movement exists on the area covered by it. This interface can support multiple alerts, from 1 up to 32.

Actions

Return Type	Name	Arguments	Optional	Documentation
None	Status	Unit Type: integer State Attribute: Array: boolean {Constraint: maxStates(integer)="32"; The maximum number of independent states is fixed at 32.}	No	This command, sent to a client implementation of the Alert interface, indicates the current state of all alerts.

Data

Name	Type	Optional	Writable	Readable	Eventable	Documentation
State	Array: boolean {Constraint: maxStates(integer)="32"; The maximum number of independent states is fixed at 32.}	Yes	Yes	Yes	Yes	*State* indicates the state of an alert. A bit set to "true" indicates an active alert, a bit set to "false" indicates idle.
Enable	Array: boolean {Constraint: maxEnables(integer)="32"; The maximum number of independent enables is fixed at 32.}	Yes	Yes	Yes	Yes	*Enable* indicates if an alert is enabled or disabled. A disabled alert will never trigger. A bit set to "true" indicates alert is enabled, a bit set to "false" indicates alert is disabled.

0x100 - Alert Client Interface

The *Alert Client* interface can be used by any device that requires receiving an alert.

0x200 - On/Off Server Interface

The *On/Off Server* interface enables a device to have some feature (light, relay, LED, etc) locally turned on/off.

Data

Name	Type	Optional	Writable	Readable	Eventable	Documentation
State	boolean	Yes	Yes	Yes	Yes	*State* indicates the current On (when true)/Off (when false) state.

0x200 - On/Off Client Interface

The *On/Off Client* interface enables a device to remotely turn on/off have some feature. It allows, for example, turning on or off a siren, an LED or a relay.

Actions

Return Type	Name	Arguments	Optional	Documentation
None	On	None	No	This command, sent to a server implementation of the On-Off interface, turns some device feature on.
None	Off	None	No	This command, sent to a server implementation of the On-Off interface, turns some device feature off.
None	Toggle	None	No	This command, sent to a server implementation of the On-Off interface, toggles the state of some device feature. If the feature was set to On, it will be turned Off and vice-versa.

0x300 - Simple Power Metering Server Interface

The *Simple Power Metering Server* interface enables a device to realize measurements of electric quantities that are made available for other devices to read. All attributes are optional so is up to the device definition to define which ones are implemneted.

Actions

Return Type	Name	Arguments	Optional	Documentation
None	Report	Number of Attributes: integer {Constraint: minValue(integer)="0x00"} {Constraint: maxValue(integer)="0xFF"} Pair ID-Value: Struct - integer ***Attribute ID*** Identifier of an attribute whose value is sent. {Constraint: minValue(integer)="0x00"} {Constraint: maxValue(integer)="0xFF"} - integer ***Attribute Value*** Value currently store in the specified attribute.	Yes	This optional command, sent to a client implementation of the Simple Power Metering interface, will send the value of all implemented attributes with the periodicity defined by ***Report Interval*** attribute.

Data

Name	Type	Optional	Writable	Readable	Eventable	Documentation
Energy	Struct {UnitOfMeasure: "Watts/Hour"} - integer *Precision Code* *Precision Code* indicates a metric prefix that affects the basic unit of measurement. {Constraint: Possible Values; *Precision Code* can only take a value from the following: 0x00, 0x10, 0x11, 0x12, 0x13, 0x20, 0x21, 0x22 and 0x23.} - integer *Energy Value* {Constraint: minValue(integer)="0x00000000"} {Constraint: maxValue(integer)="0xFFFFFFFF"}	Yes	Yes	Yes	Yes	*Energy* attribute stores energy consumption from device power up or from a measurement reset
Energy at Last Reset	Struct {UnitOfMeasure: "Watts/Hour"} - integer *Precision Code* *Precision Code* indicates a metric prefix that affects the basic unit of measurement. {Constraint: Possible Values; *Precision Code* can only take a value from the following: 0x00, 0x10, 0x11, 0x12, 0x13, 0x20, 0x21, 0x22 and 0x23.} - integer *Energy Value* {Constraint: minValue(integer)="0x00000000"} {Constraint: maxValue(integer)="0xFFFFFFFF"}	Yes	Yes	Yes	Yes	*Energy at Last Reset* stores the *Energy* value at the instant a measurement reset occurred (see *Measurement Reset* action). It is mandatory if the referred action is implemented.
Time at Last Reset	Struct {UnitOfMeasure: "Seconds"} - boolean *Time Reference* *Time Reference* indicates the original source from which time is measured/referenced. - integer *Energy Value* {Constraint: minValue(integer)="0x00000000"} {Constraint: maxValue(integer)="0xFFFFFFFF"}	Yes	Yes	Yes	Yes	*Time at Last Reset* stores the time value (from device uptime or UTC) at the instant a measurement reset occurred (see *Measurement Reset* action). It is mandatory if the referred action is implemented.

Name	Type	Optional	Writable	Readable	Eventable	Documentation
Instantaneous Power	Struct {UnitOfMeasure: "Watts"} - integer ***Precision Code*** ***Precision Code*** indicates a metric prefix that affects the basic unit of measurement. {Constraint: Possible Values; ***Precision Code*** can only take a value from the following: 0x00, 0x10, 0x11, 0x12, 0x13, 0x20, 0x21, 0x22 and 0x23.} - integer ***Power Value*** {Constraint: minValue(integer)="0x00000000"} {Constraint: maxValue(integer)="0xFFFFFFFF"}	Yes	Yes	Yes	Yes	***Instantaneous Power*** stores the presently instantaneous power value.
Average Power	Struct {UnitOfMeasure: "Watts"} - integer ***Precision Code*** ***Precision Code*** indicates a metric prefix that affects the basic unit of measurement. {Constraint: Possible Values; ***Precision Code*** can only take a value from the following: 0x00, 0x10, 0x11, 0x12, 0x13, 0x20, 0x21, 0x22 and 0x23.} - integer ***Power Value*** {Constraint: minValue(integer)="0x00000000"} {Constraint: maxValue(integer)="0xFFFFFFFF"}	Yes	Yes	Yes	Yes	***Average Power*** stores the power measured over a period of time specified by ***Average Power Interval*** .
Average Power Interval	integer {UnitOfMeasure: "Seconds"} {Constraint: minValue(integer)="0x0000"} {Constraint: maxValue(integer)="0xFFFF"}	Yes	Yes	Yes	Yes	***Average Power Interval*** specifies the time period over which power should be averaged and stored into ***Average Power*** .
Voltage	Struct {UnitOfMeasure: "Volts"} - integer ***Precision Code*** ***Precision Code*** indicates a metric prefix that affects the basic unit of measurement. {Constraint: Possible Values; ***Precision Code*** can only take a value from the following: 0x00, 0x10, 0x11, 0x12, 0x13, 0x20, 0x21, 0x22 and 0x23.} - integer ***Voltage Value*** {Constraint: minValue(integer)="0x00000000"} {Constraint: maxValue(integer)="0xFFFFFFFF"}	Yes	Yes	Yes	Yes	***Voltage*** stores the presently instantaneous voltage value.

Name	Type	Optional	Writable	Readable	Eventable	Documentation
Current	Struct {UnitOfMeasure: "Amperes"} - integer ***Precision Code*** ***Precision Code*** indicates a metric prefix that affects the basic unit of measurement. {Constraint: Possible Values; ***Precision Code*** can only take a value from the following: 0x00, 0x10, 0x11, 0x12, 0x13, 0x20, 0x21, 0x22 and 0x23.} - integer ***Current Value*** {Constraint: minValue(integer)="0x00000000"} {Constraint: maxValue(integer)="0xFFFFFFFF"}	Yes	Yes	Yes	Yes	***Current*** stores the presently instantaneous current value.
Frequency	Struct {UnitOfMeasure: "Hertz"} - integer ***Precision Code*** ***Precision Code*** indicates a metric prefix that affects the basic unit of measurement. {Constraint: Possible Values; ***Precision Code*** can only take a value from the following: 0x00, 0x10, 0x11, 0x12, 0x13, 0x20, 0x21, 0x22 and 0x23.} - integer ***Frequency Value*** {Constraint: minValue(integer)="0x00000000"} {Constraint: maxValue(integer)="0xFFFFFFFF"}	Yes	Yes	Yes	Yes	***Frequency*** stores the presently instantaneous frequency value.
Power Factor	integer {Constraint: minValue(integer)="0x00"} {Constraint: maxValue(integer)="0xFF"}	Yes	Yes	Yes	Yes	***Power Factor*** stores the ratio between real and apparent power.
Report Interval	integer {UnitOfMeasure: "Seconds"} {Constraint: minValue(integer)="0x0000"} {Constraint: maxValue(integer)="0xFFFF"}	Yes	Yes	Yes	Yes	***Report Interval*** stores the periodic time interval, in seconds, at which the ***Report*** action should be sent. If set to 0 (zero) the action will never be. It is mandatory if the referred action is implemented.

0x300 - Simple Power Metering Client Interface

The ***Simple Power Metering Client*** interface enables a device to receive measurements of electric quantities.

Actions

Return Type	Name	Arguments	Optional	Documentation
None	Measurement Reset	None	Yes	This optional command sent to a server implementation of the Simple Power Metering interface, performs the following operations: ***Copy Energy attribute present value into Energy at Last Reset attribute;*** Set Energy attribute to 0 (zero); * Store device time into Time at Last Reset attribute. Due to the nature of these operations, if this action is implemented then ***Energy at Last Reset*** and ***Time at Last Reset*** must also be.

0x301 - Simple Temperature Server Interface

The ***Simple Temperature Server*** interface enables a device to be able to provide temperature readings.

Data

Name	Type	Optional	Writable	Readable	Eventable	Documentation
Measured Temperature	integer {UnitOfMeasure: "One hundredth Celsius"} {Constraint: minValue(integer)="-32768"} {Constraint: maxValue(integer)="+32767"}	Yes	Yes	Yes	Yes	*Measured Temperature* holds the current measured temperature, in one hundredth (1/100) of Celsius degrees.
Minimum Measureable Temperature	integer {UnitOfMeasure: "One hundredth Celsius"} {Constraint: minValue(integer)="-32768"} {Constraint: maxValue(integer)="+32767"}	Yes	Yes	Yes	Yes	*Minimum Measureable Temperature* holds the minimum temperature, in one hundredth (1/100) of Celsius degrees, that can be measured by the device.
Maximum Measureable Temperature	integer {UnitOfMeasure: "One hundredth Celsius"} {Constraint: minValue(integer)="-32768"} {Constraint: maxValue(integer)="+32767"}	Yes	Yes	Yes	Yes	*Maximum Measureable Temperature* holds the maximum temperature, in one hundredth (1/100) of Celsius degrees, that can be measured by the device.

Name	Type	Optional	Writable	Readable	Eventable	Documentation
Tolerance	integer {UnitOfMeasure: "One hundredth Celsius"} {Constraint: minValue(integer)="0x0000"} {Constraint: maxValue(integer)="0xFFFF"}	Yes	Yes	Yes	Yes	*Tolerance* holds the magnitude, in one hundredth (1/100) of Celsius degrees of the error associated with *Measured Temperature* , which means the actual value is between (*Measured Temperature* – *Tolerance*) and (*Measured Temperature* + *Tolerance*).

0x301 - Simple Temperature Client Interface

The *Simple Temperature Client* interface enables a device to be able to receive temperature readings.