

e-Room Plus[®]

CONTROLADOR DE CLIMA PARA REDES LONWORKS[®]
Ref: RP.626601-000

Perfil Funcional

Versión 0.x.x

Este documento describe las variables de red y los parámetros de configuración del producto que forman su interface de red Lon. La aplicación está formada por objetos lógicos (perfiles funcionales) de acuerdo con las Directrices de Interoperabilidad de LONMARK[™].

Resource Files versión 1.0

Perfiles Funcionales del producto

Cantidad	Código	Perfil Funcional	Versión
1	1060	Occupancy Sensor	2.1
1	3040	Lamp Actuator SR5	1.0
1	3040	Lamp Actuator SR6	1.0
1	3071	Occupancy Controller	2.0
1	3200	Switch	1.0
1	3200	Switch Auxiliar 2	1.0
1	3200	Switch Dimmer	1.0
1	3200	Switch Sunblind	1.1
1	6110	Sunblind Actuator	1.0
1	6111	Sunblind Controller	1.0
1	8020	Fan Coil Unit	2.2
1	21234	Room Medic Alarm	1.0

Perfil Funcional:

Occupancy Sensor

08504 v2.1

Contenido

1. Descripción.....	3
1.1. Funcionamiento	3
1.1.1. Tarjetero (Hotel 2T-T/4T-T).....	3
1.1.2. Detector Presencia-Puerta (Hotel 2T-D/4T-D).....	3
1.1.3. Cama Ocupada (Residencia)	4
1.1.4. Detector Presencia (Oficina 2T-I/4T-I).....	4
2. Interfaz de red.....	5
3. Variables de red.....	6
3.1. Variables de salida	6
3.1.1. nvoOccup.....	6
3.2. Variables de configuración.....	7
3.2.1. nciPresenceTime	7

1. Descripción

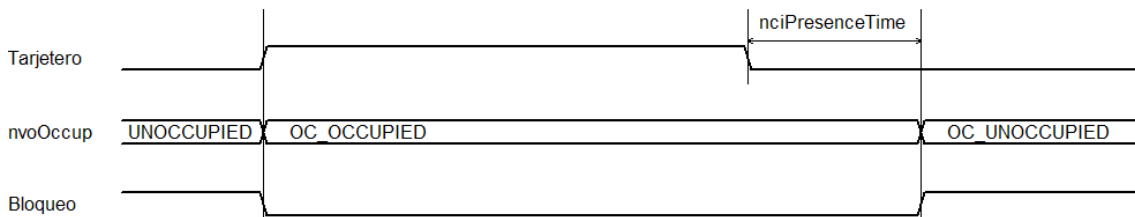
El objeto *Occupancy Sensor* se usa en dispositivos con sensores cuya salida indica el estado de ocupación/desocupación de una estancia. El paso al estado ocupado se produce por un flanco de activación del sensor, mientras que el paso al estado desocupado se produce al expirar el tiempo establecido en `nciPresenceTime`.

Su salida se suele conectar a un controlador, que se encarga de realizar una acción en función de la ocupación, como puede ser encender una luz o ajustar la consigna del climatizador a un nivel predeterminado.

1.1. Funcionamiento

1.1.1. Tarjetero (Hotel 2T-T/4T-T)

En esta configuración el estado de ocupación viene determinado por la presencia/ausencia de la tarjeta en el tarjetero. El teclado del equipo se bloquea en estado desocupado.



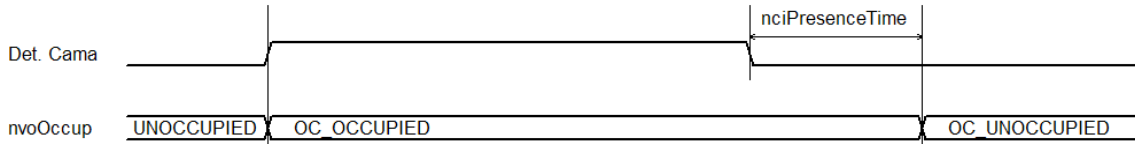
1.1.2. Detector Presencia-Puerta (Hotel 2T-D/4T-D)

En este caso el estado de ocupación se obtiene a partir de una combinación de dos sensores, uno tipo PIR y un contacto instalado en la puerta. Un pulso del sensor PIR provoca el paso a Ocupado, mientras que la no detección de pulsos durante un determinado tiempo después de cerrar la puerta provoca el paso a Desocupado. Este tiempo está compuesto por un tiempo de guarda (por si el PIR emite justo después de cerrar la puerta) y el valor configurado en `nciPresenceTime`.

El teclado del equipo se bloquea en estado desocupado.

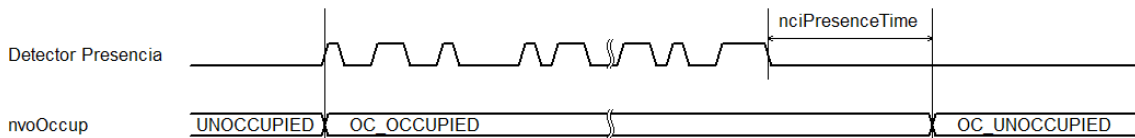
1.1.3. Cama Ocupada (Residencia)

En esta configuración el estado de ocupación sigue la evolución del sensor.

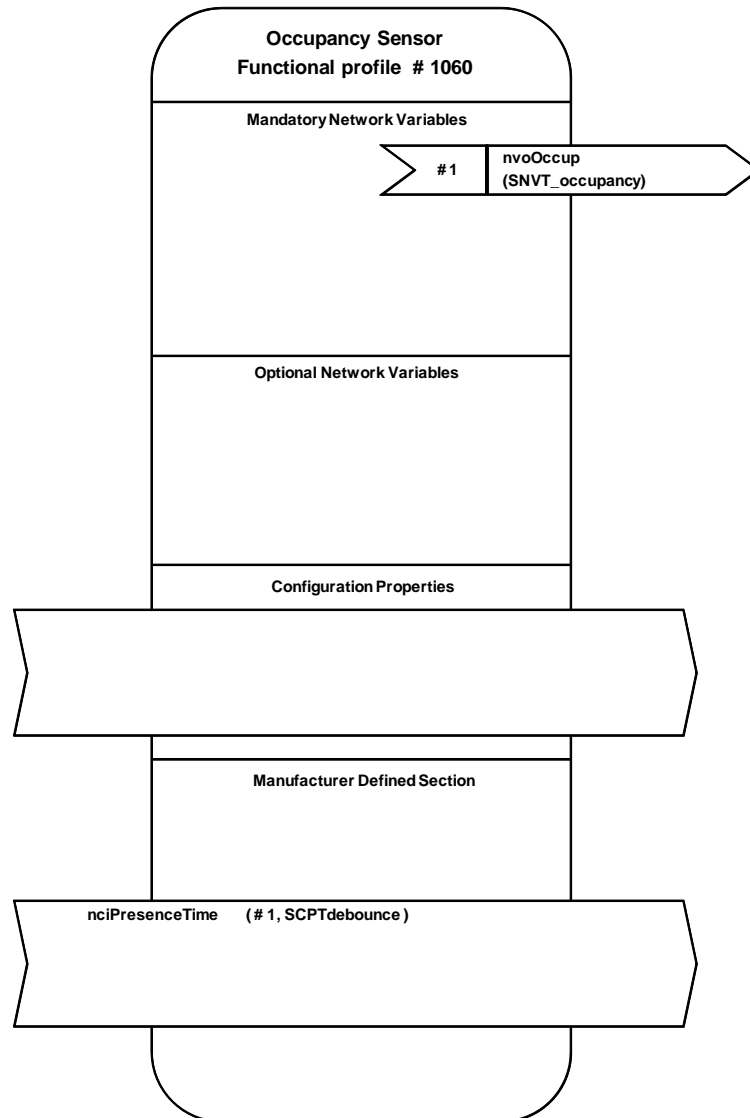


1.1.4. Detector Presencia (Oficina 2T-I/4T-I)

Un flanco de subida del sensor provoca el paso a Ocupado, mientras que un flanco de bajada arranca un temporizador que, al expirar, provoca el paso a Desocupado. Dicho temporizador se recarga con cada flanco de subida.



2. Interfaz de red



3. Variables de red

3.1. Variables de salida

3.1.1. nvoOccup

```
network output SNVT_occupancy nvoOccup;
```

Esta variable proporciona el estado de ocupación de la zona donde se halla el controlador.

Tipo

SNVT_occupancy

```
typedef occup_t SNVT_occupancy;
```

Margen de valores

```
typedef enum occup_t
{
    /* 0 */ OC_OCCUPIED,
    /* 1 */ OC_UNOCCUPIED,
    ...
    /* -1 */ OC_NUL = -1
} occup_t;
```

Valor por defecto

```
{-1}          OC_NUL, valor inválido
```

3.2. Variables de configuración

3.2.1. nciPresenceTime

```
network input cp SCPTdebounce nciPresenceTime;
```

Esta propiedad de configuración establece el periodo de tiempo que puede pasar después de que el sensor no detecte presencia para que la variable `nvoOccup` indique la desocupación de una estancia.

Tipo

SCPTdebounce, derivado de SNVT_time_sec.

```
typedef unsigned long SNVT_time_sec;
```

Margen de valores

0...65530 (0...6553 segundos, con resolución 1 segundo)

Valor por defecto

{ 100 } 10 segundos

Perfil Funcional:

Lamp Actuator SR5

08504 v1.0

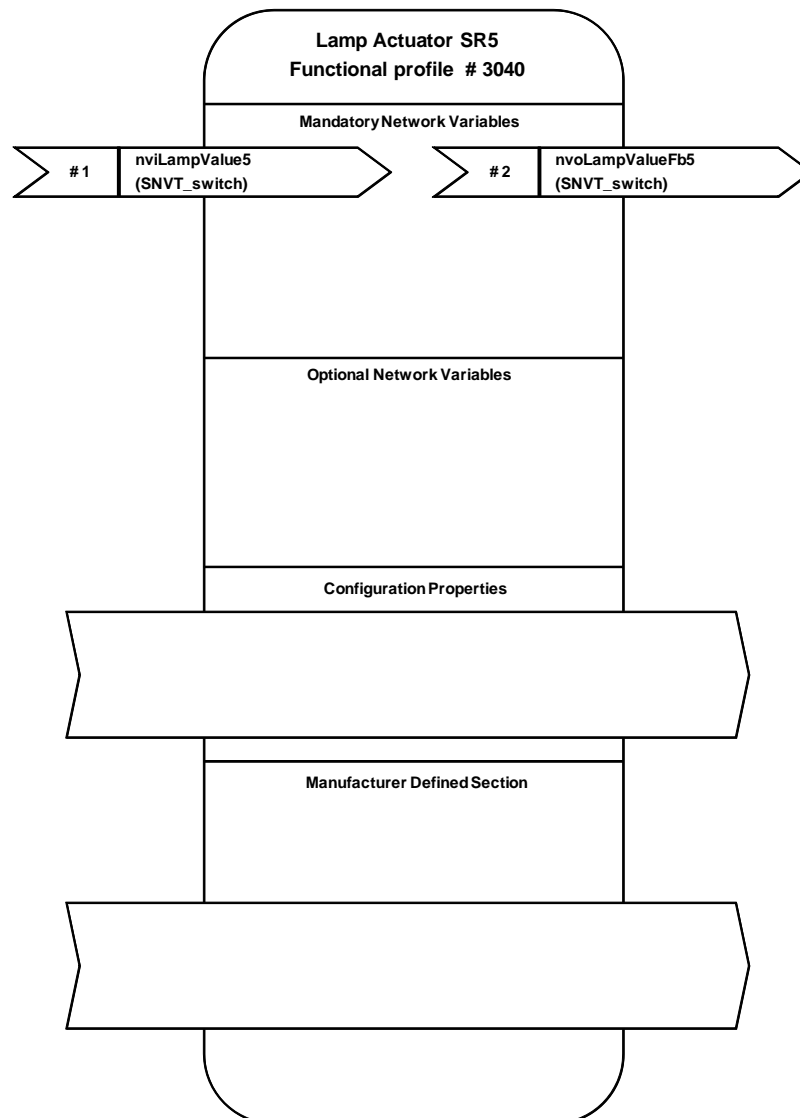
Contenido

1. Descripción.....	3
2. Interfaz de red.....	3
3. Variables de red.....	4
3.1. Variables de entrada.....	4
3.1.1. nviLampValue5	4
3.2. Variables de salida	5
3.2.1. nvoLampValueFb5.....	5

1. Descripción

El objeto *Lamp Actuator SR5* se usa para controlar la Salida Relé 5 del e-Room Plus en las instalaciones a 2 Tubos. En instalaciones a 4 Tubos la Salida Relé se utiliza para el control de una electroválvula y el perfil permanece desactivado, aunque se cambie el valor de la variable de entrada.

2. Interfaz de red



3. Variables de red

3.1. Variables de entrada

3.1.1. nviLampValue5

```
network input SNVT_switch nviLampValue5;
```

Esta variable proporciona un mecanismo mediante el cual otro dispositivo puede actuar sobre el estado de la salida.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Apagado

1 Encendido (**Nota:** Si value = 0, se interpreta como Apagado)

-1 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

3.2. Variables de salida

3.2.1. nvoLampValueFb5

```
network output SNVT_switch nvoLampValueFb5;
```

Esta variable indica el estado del actuador (Apagado/Encendido) y el nivel porcentual de intensidad.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Apagado

1 Encendido (**Nota:** Si value = 0, se interpreta como Apagado)

-1 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

Perfil Funcional:

Lamp Actuator SR6

08504 v1.0

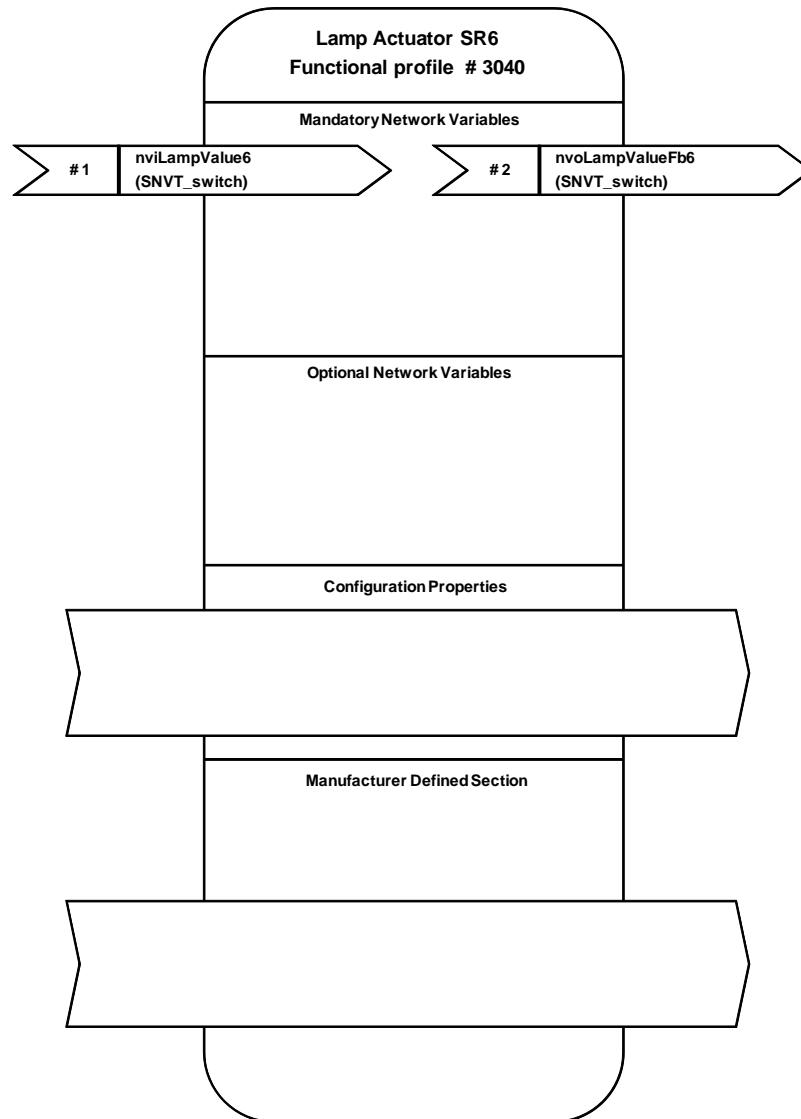
Contenido

1. Descripción	3
2. Interfaz de red	3
3. Variables de red	4
3.1. Variables de entrada.....	4
3.1.1. nviLampValue6	4
3.2. Variables de salida	5
3.2.1. nvoLampValueFb6.....	5

1. Descripción

El objeto *Lamp Actuator SR6* se usa para controlar la Salida Relé 6 del e-Room Plus, que normalmente gobernará el sistema de iluminación de la estancia.

2. Interfaz de red



3. Variables de red

3.1. Variables de entrada

3.1.1. nviLampValue6

```
network input SNVT_switch nviLampValue6;
```

Esta variable proporciona un mecanismo mediante el cual otro dispositivo puede actuar sobre el estado de la salida.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0	Apagado
1	Encendido (Nota: Si value = 0, se interpreta como Apagado)
-1	Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

3.2. Variables de salida

3.2.1. nvoLampValueFb6

```
network output SNVT_switch nvoLampValueFb6;
```

Esta variable indica el estado del actuador (Apagado/Encendido) y el nivel porcentual de intensidad.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Apagado

1 Encendido (**Nota:** Si value = 0, se interpreta como Apagado)

-1 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

Perfil Funcional:

Occupancy Controller

08504 v2.0

Contenido

1. Descripción	3
1.1. Funcionamiento	3
2. Interfaz de red	4
3. Variables de red	5
3.1. Variables de entrada.....	5
3.1.1. nviOccupancy	5
3.1.2. nviManOverride.....	6
3.2. Variables de salida	7
3.2.1. nvoLampValue.....	7
3.2.2. nvoCourtesyLamp.....	8
4. Variables de configuración	9
4.1. nciHoldTime.....	9
4.2. nciCourtesyTime.....	10

1. Descripción

El objeto *Occupancy Controller* permite controlar un actuador (normalmente una lámpara) en función del estado de ocupación de una estancia. Cuando la estancia está ocupada, se activa la salida, mientras que si la estancia pasa a estar desocupada la salida se desactiva.

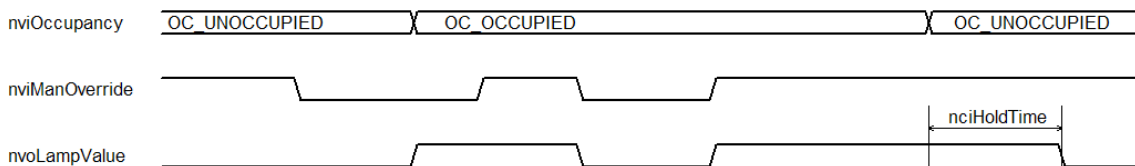
Dispone de una entrada para controlar manualmente la salida cuando la estancia está ocupada.

Al objeto se le ha añadido la posibilidad de controlar una luz de cortesía, cuyo estado varía en función de las entradas (sólo en las configuraciones para Hotel).

1.1. Funcionamiento

El valor de la variable de salida `nvoLampValue` viene determinado por el valor de `nviOccupancy`:

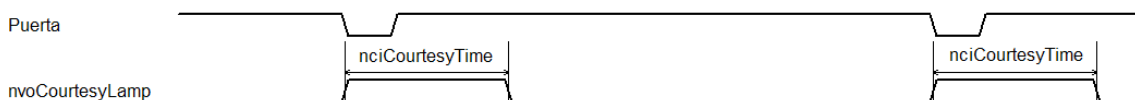
- cuando pasa de Desocupado a Ocupado, la salida se activa automáticamente.
- cuando pasa de Ocupado a Desocupado la salida se desactiva automáticamente, pudiéndose establecer un retardo en la desactivación.
- mientras esté en Ocupado, se puede cambiar el valor de la salida mediante la variable `nviManOverride`.



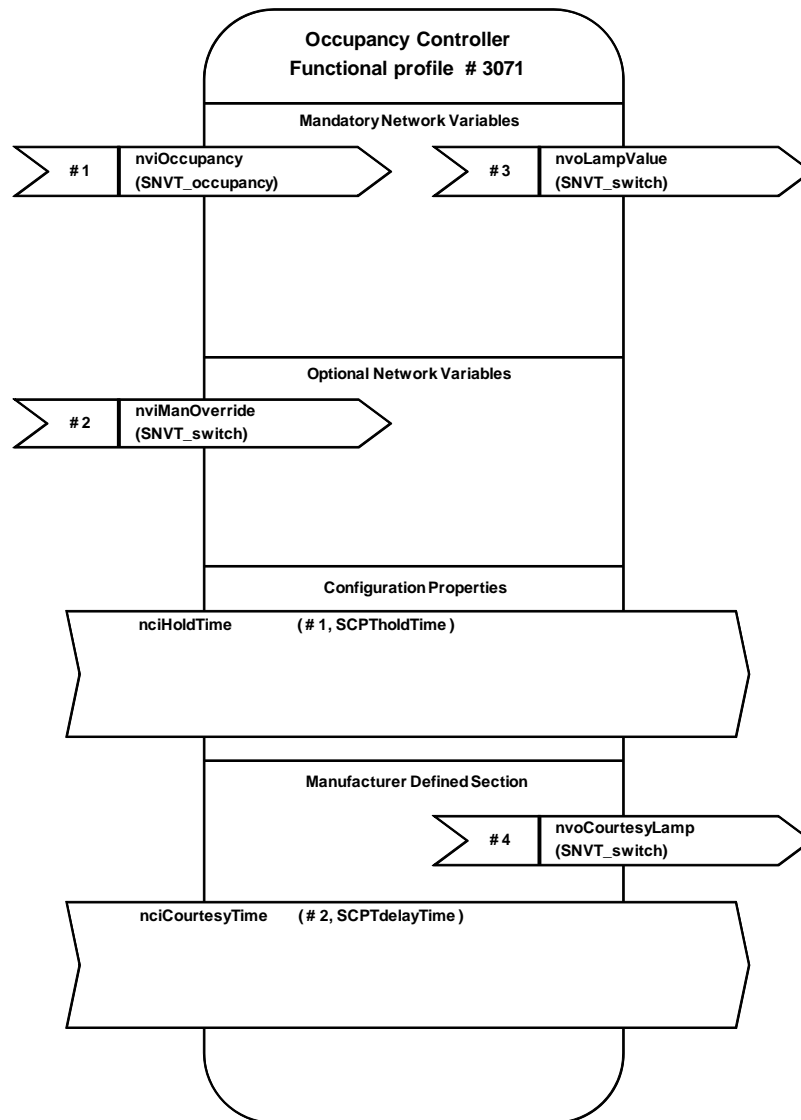
La salida Cortesía se activa al insertar o extraer la tarjeta del tarjetero (configuración Hotel 2T-T/4T-T),



o cada vez que se abre la puerta de la habitación (configuración Hotel 2T-D/4T-D).



2. Interfaz de red



3. Variables de red

3.1. Variables de entrada

3.1.1. nviOccupancy

```
network input SNVT_occupancy nviOccupancy;
```

Esta variable proporciona al controlador el estado de ocupación de la zona donde se halla.

Tipo

SNVT_occupancy

```
typedef occup_t SNVT_occupancy;
```

Margen de valores

```
typedef enum occup_t
{
    /* 0 */ OC_OCCUPIED,
    /* 1 */ OC_UNOCCUPIED,
    /* 2 */ OC_BYPASS,
    /* 3 */ OC_STANDBY,
    /* -1 */ OC_NUL = -1
} occup_t;
```

Valor por defecto

```
{ 1 }          OC_OCCUPIED, ocupado
```

Notas

El controlador sólo responderá cuando reciba alguno de estos valores: OC_OCCUPIED y OC_UNOCCUPIED. Para el resto de valores de la enumeración, el controlador no realizará acción alguna.

3.1.2. nviManOverride

```
network input SNVT_switch nviManOverride;
```

Esta variable proporciona la posibilidad de controlar local y manualmente la salida del actuador.

Cuando la variable `nviOccupancy` tiene el valor `OC_OCCUPIED`, cualquier cambio en la variable `nviManOverride` provoca que el controlador traslade el valor de esta variable a la salida.

Cuando la variable `nviOccupancy` recibe el valor `OC_UNOCCUPIED`, el controlador fuerza la salida al estado Apagado y los cambios en esta variable se ignoran.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0	Apagado
1	Encendido (Nota: Si <code>value = 0</code> , se interpreta como Apagado)
-1	Inválido: El valor leído no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

3.2. Variables de salida

3.2.1. nvoLampValue

```
network output SNVT_switch nvoLampValue;
```

Esta variable proporciona el estado que debe adoptar el actuador conectado al controlador.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Apagado

1 Encendido (Nota: Si value = 0, se interpreta como Apagado)

-1 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

3.2.2. nvoCourtesyLamp

```
network output SNVT_switch nvoCourtesyLamp;
```

Esta variable proporciona el estado que debe adoptar el actuador de cortesía conectado al controlador. Se activa automáticamente al insertar/extraer la tarjeta en las configuraciones Hotel 2T-T/4T-T y al abrir la puerta en las configuraciones Hotel 2T-D/4T-D. Se desactiva automáticamente al expirar el tiempo establecido en `nciCourtesyTime`.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0	Apagado
1	Encendido (Nota: Si value = 0, se interpreta como Apagado)
-1	Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

4. Variables de configuración

4.1. *nciHoldTime*

```
network input SCPTholdTime cp nciHoldTime;
```

Esta propiedad de configuración establece el periodo de tiempo que puede pasar antes de que el controlador apague automáticamente la salida `nvoLampValue` al recibir `nviOccupancy` el valor `OC_UNOCCUPIED`.

Tipo

SCPTholdTime, derivado de SNVT_time_sec.

```
typedef unsigned long SNVT_time_sec;
```

Margen de valores

0...65530 (1...6553 segundos, con resolución 1 segundo)

Valor por defecto

{ 0 } 0 segundos

4.2. *nciCourtesyTime*

```
network input SCPTdelayTime cp nciCourtesyTime;
```

Esta propiedad de configuración establece el periodo de tiempo que puede pasar antes de que el controlador desactive automáticamente la salida `nvoCourtesyLamp`.

Tipo

SCPTdelayTime, derivado de SNVT_time_sec.

```
typedef unsigned long SNVT_time_sec;
```

Margen de valores

0...65530 (0...6553 segundos, con resolución 1 segundo)

Valor por defecto

{ 200 } 20 segundos.

Notas

Un valor de 0 segundos implica que la variable `nvoCourtesyLamp` no se activa nunca.

Perfil Funcional:

Switch

08504 v1.0

Contenido

1. Descripción	3
1.1. Funcionamiento	3
1.1.1. Pulsador Luz (Hotel 2T-T/4T-T, Hotel 2T-D/4T-D, Oficina 2T-I/4T-I)	3
1.1.2. Persianas (Oficina 2T-P).....	3
2. Interfaz de red	4
3. Variables de red	5
3.1. Variables de entrada.....	5
3.1.1. nviSwitchFb	5
3.2. Variables de salida	6
3.2.1. nvoSwitch	6
3.2.2. nvoSetting.....	7

1. Descripción

El objeto *Switch* es un objeto sensor que se usa para todo tipo de interruptores con o sin hardware específico. Este objeto se puede usar tanto en aplicaciones en lazo abierto como en lazo cerrado.

1.1. Funcionamiento

1.1.1. Pulsador Luz (Hotel 2T-T/4T-T, Hotel 2T-D/4T-D, Oficina 2T-I/4T-I)

	nviSwitchFb.state	nvoSwitch	nvoSetting
Pulsación	0	{ 200, 1 }	SET_ON
	1	{ 0, 0 }	SET_OFF
	-1	Conmuta entre los dos valores anteriores	Conmuta entre los dos valores anteriores
Liberación	-	-	-

1.1.2. Persianas (Oficina 2T-P)

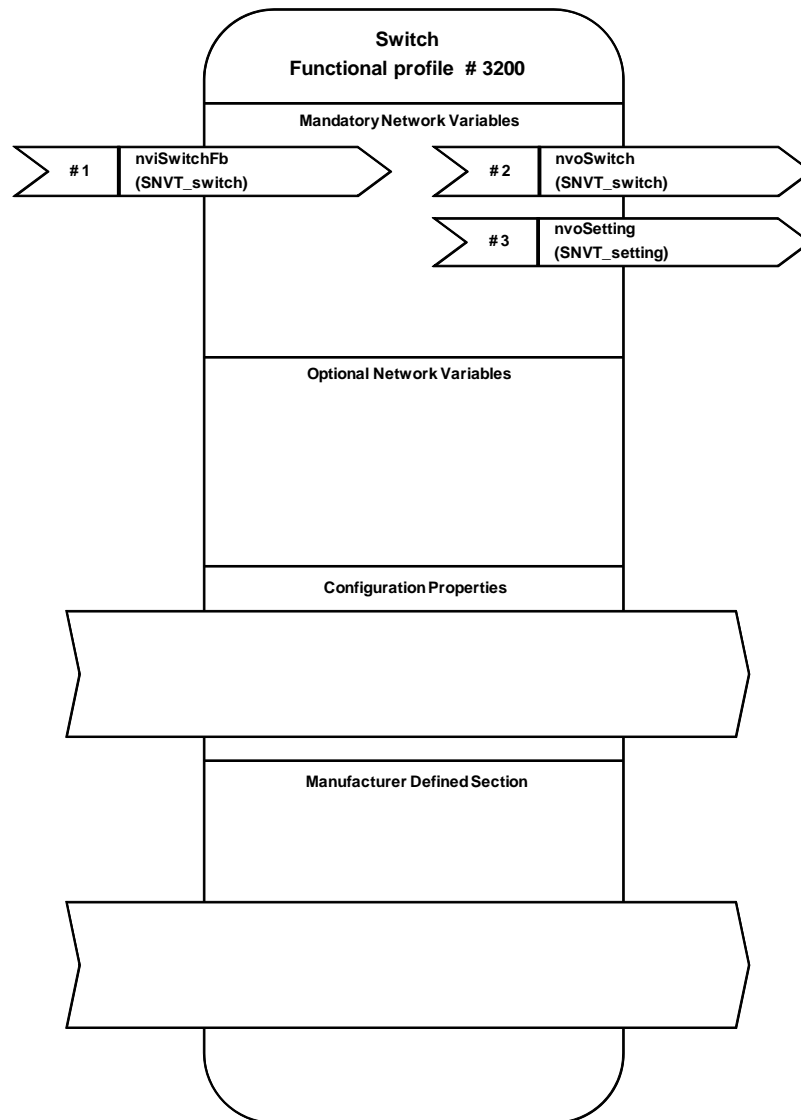
Pulsador de Subida

	nvoSwitch	nvoSetting
Pulsación corta	{ 200, 1 }	SET_ON
Liberación	{ 0, 0 }	-
Pulsación larga	{ 200, 1 }	SET_UP
Liberación	{ 0, 0 }	SET_STOP

Pulsador de Bajada

	nvoSwitch	nvoSetting
Pulsación corta	{ 200, 1 }	SET_OFF
Liberación	{ 0, 0 }	-
Pulsación larga	{ 200, 1 }	SET_DOWN
Liberación	{ 0, 0 }	SET_STOP

2. Interfaz de red



3. Variables de red

3.1. Variables de entrada

3.1.1. nviSwitchFb

```
network input SNVT_switch nviSwitchFb;
```

Esta variable proporciona la información del estado de otros dispositivos, permitiendo al objeto *switch* realizar acciones tales como conmutados.

Tipo

SNVT_Switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Apagado

1 Encendido (**Nota:** Si `value = 0`, se interpreta como Apagado)

-1 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

3.2. Variables de salida

3.2.1. nvoSwitch

```
network output SNVT_switch nvoSwitch;
```

Esta variable proporciona la salida del objeto. Se usa para el control directo de los dispositivos.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Apagado

1 Encendido (**Nota:** Si value = 0, se interpreta como Apagado)

-1 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

3.2.2. nvoSetting

```
network output SNVT_setting nvoSetting;
```

Esta variable establece el estado (ON/OFF) y la regulación porcentual del nivel (0%...100%) del controlador.

Tipo

SNVT_setting

```
typedef struct
{
    setting_t function;
    unsigned short setting;
    signed long rotation;
} SNVT_setting;
```

Margen de valores

```
function
```

Valor	Nemónico	Valor setting	Descripción
-1	SET_NUL	Se ignora	Inválido
0	SET_OFF	Se ignora	Apagar / Bajar Persiana a final carrera
1	SET_ON	Se ignora	Encender / Subir Persiana a final carrera
2	SET_DOWN	Se ignora	Disminuir nivel / Bajar Persiana
3	SET_UP	Se ignora	Aumentar nivel / Subir Persiana
4	SET_STOP	Se ignora	Parar acción

```
setting
```

0...200 (0%...100%, con resolución 0,5%)
 255 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

```
rotation
```

-17999...18000 (-359,98°...360°)
 32767 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

```
{ SET_STOP, 255, 32767 } Parar, inválido, inválido
```

Perfil Funcional:

Switch Auxiliar 2

08504 v1.0

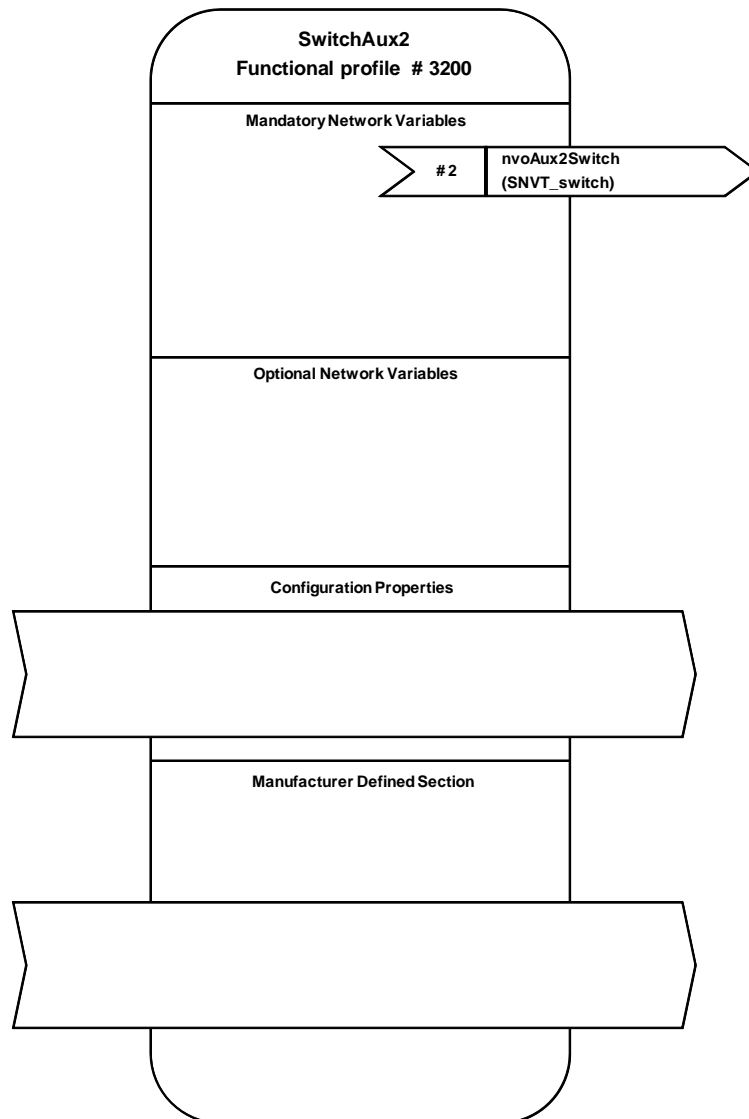
Contenido

1. Descripción.....	3
2. Interfaz de red.....	3
3. Variables de red.....	4
3.1. Variables de salida	4
3.1.1. nvoAux2Switch	4

1. Descripción

El objeto *Switch Auxiliar 2* es un objeto sensor que refleja el estado de la Entrada Analógica 1 del equipo cuando está configurado para ello.

2. Interfaz de red



3. Variables de red

3.1. Variables de salida

3.1.1. nvoAux2Switch

```
network output SNVT_switch nvoAux2Switch;
```

Esta variable proporciona la salida del objeto. Se usa para el control directo de los dispositivos.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Apagado

1 Encendido (**Nota:** Si value = 0, se interpreta como Apagado)

-1 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

Perfil Funcional:

Switch Dimmer

08504 v1.0

Contenido

1. Descripción	3
1.1. Funcionamiento	3
2. Interfaz de red	4
3. Variables de red	5
3.1. Variables de entrada.....	5
3.1.1. nviDimSwitchFb	5
3.2. Variables de salida	6
3.2.1. nvoDimSwitch	6
3.2.2. nvoDimSetting	7

1. Descripción

El objeto *Switch Dimmer* permite controlar un actuador con capacidad de regulación. Este objeto sólo se puede utilizar mediante el mando IR del equipo.

1.1. Funcionamiento

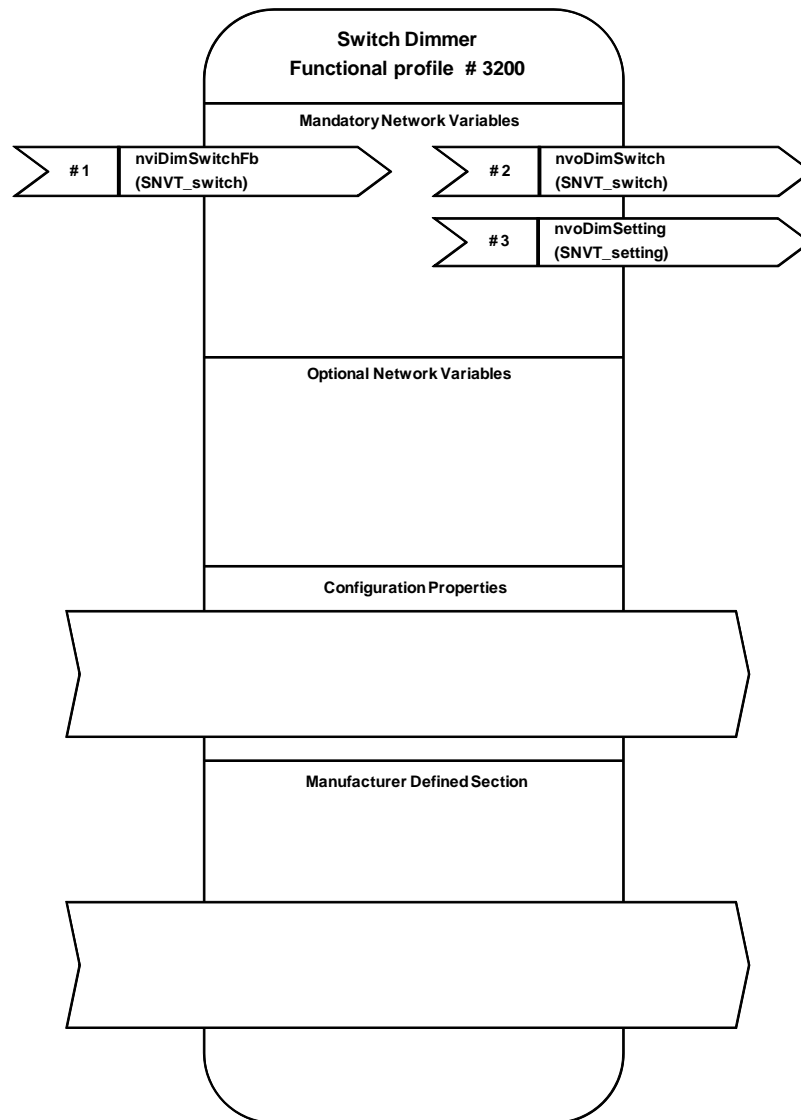
Pulsador de Subida

	nviDim-SwitchFb	nvoDimSwitch		nvoDimSetting	
	state	state	value	function	setting
Pulsación corta	1	0	-	SET_OFF	-1
	0	1	-	SET_ON	-1
	sin feedback	Conmuta entre 0 y 1	-	Conmuta entre SET_OFF y SET_ON	-1
Liberación		-	-	-	-
Pulsación larga (cada 100ms)		1	value + 3	SET_UP	3
Liberación		-	-	SET_STOP	-1

Pulsador de Bajada

	nviDim-SwitchFb	nvoDimSwitch		nvoDimSetting	
	state	state	value	function	setting
Pulsación corta	1	0	-	SET_OFF	-1
	0	1	-	SET_ON	-1
	sin feedback	Conmuta entre 0 y 1	-	Conmuta entre SET_OFF y SET_ON	-1
Liberación		-	-	-	-
Pulsación larga (cada 100ms)		1	value - 3	SET_DOWN	3
Liberación		-	-	SET_STOP	-1

2. Interfaz de red



3. Variables de red

3.1. Variables de entrada

3.1.1. nviDimSwitchFb

```
network input SNVT_switch nviDimSwitchFb;
```

Esta variable proporciona la información del estado de otros dispositivos, permitiendo al objeto *Switch Dimmer* realizar acciones tales como conmutados.

Tipo

SNVT_Switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Apagado

1 Encendido (**Nota:** Si value = 0, se interpreta como Apagado)

-1 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

3.2. Variables de salida

3.2.1. nvoDimSwitch

```
network output SNVT_switch nvoDimSwitch;
```

Esta variable proporciona la salida del objeto. Se usa para el control directo de los dispositivos.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Apagado

1 Encendido (**Nota:** Si value = 0, se interpreta como Apagado)

-1 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

3.2.2. nvoDimSetting

```
network output SNVT_setting nvoDimSetting;
```

Esta variable establece el estado (ON/OFF) y la regulación porcentual del nivel (0%...100%) del controlador.

Tipo

SNVT_setting

```
typedef struct
{
    setting_t function;
    unsigned short setting;
    signed long rotation;
} SNVT_setting;
```

Margen de valores

```
function
```

Valor	Nemónico	Valor setting	Descripción
-1	SET_NUL	Se ignora	Inválido
0	SET_OFF	Se ignora	Apagar
1	SET_ON	Se ignora	Encender
2	SET_DOWN	Relativo	Disminuir nivel
3	SET_UP	Relativo	Aumentar nivel
4	SET_STOP	Se ignora	Parar acción

```
setting
```

0...200 (0%...100%, con resolución 0,5%)
 255 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

```
rotation
```

-17999...18000 (-359,98°...360°)
 32767 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

```
{ SET_STOP, 255, 32767 } Parar
```

Perfil Funcional:

Switch Sunblind

08504 v1.1

Contenido

1. Descripción	3
1.1. Funcionamiento	3
2. Interfaz de red	4
3. Variables de red	5
3.1. Variables de entrada.....	5
3.1.1. nviSbSwitchFb	5
3.2. Variables de salida	6
3.2.1. nvoSbSwitch	6
3.2.2. nvoSbSetting	7

1. Descripción

El objeto *Switch Sunblind* permite controlar un actuador de persianas. Este objeto sólo se puede utilizar mediante el mando IR del equipo.

1.1. Funcionamiento

Pulsador de Subida

	feedback¹		nvoSbSwitch		nvoSbSetting
	state	value	state	value	function
Pulsación	1	0	0	0	SET_STOP
	otro	otro	1	200	SET_UP
Liberación			0 ²	0 ²	SET_STOP ²

¹ nviSbSwitchFb, si está enlazada, o nvoSbSwitch.

² Los valores se cambian al cabo del tiempo establecido en SunblindActuator::ncisblndUpTime.

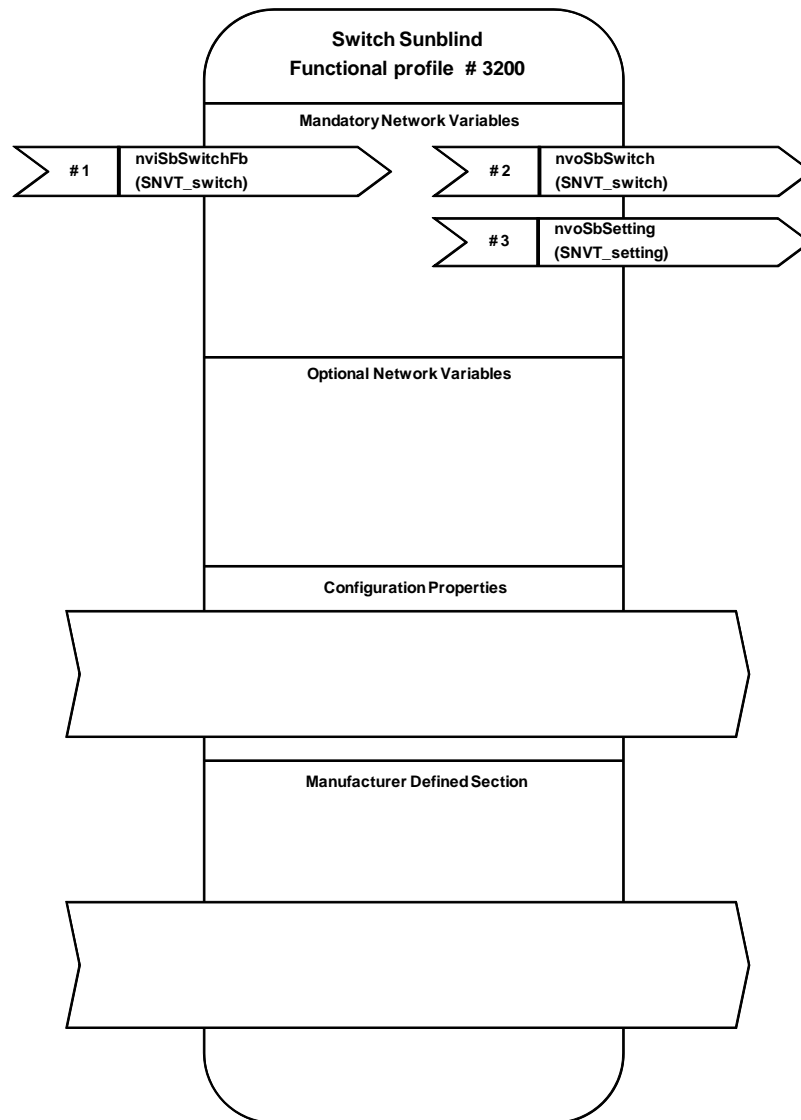
Pulsador de Bajada

	feedback¹		nvoSbSwitch		nvoSbSetting
	state	value	state	value	function
Pulsación	1	200	0	0	SET_STOP
	otro	otro	1	0	SET_DOWN
Liberación			0 ²	0 ²	SET_STOP ²

¹ nviSbSwitchFb, si está enlazada, o nvoSbSwitch.

² Los valores se cambian al cabo del tiempo establecido en SunblindActuator::ncisblndDownTime.

2. Interfaz de red



3. Variables de red

3.1. Variables de entrada

3.1.1. nviSbSwitchFb

```
network input SNVT_switch nviSbSwitchFb;
```

Esta variable proporciona la información del estado de otros dispositivos, permitiendo al objeto *Switch Sunblind* realizar acciones tales como parar cuando se está en movimiento.

Tipo

SNVT_Switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0	Apagado
1	Encendido (Nota: Si <code>value = 0</code> , se interpreta como Apagado)
-1	Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

3.2. Variables de salida

3.2.1. nvoSbSwitch

```
network output SNVT_switch nvoSbSwitch;
```

Esta variable proporciona la salida del objeto. Se usa para el control directo de los dispositivos.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0 .. 200 (0% .. 100%, con resolución 0,5%)

state

0 Apagado

1 Encendido (**Nota:** Si value = 0, se interpreta como Apagado)

-1 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Apagado

3.2.2. nvoSbSetting

```
network output SNVT_setting nvoSbSetting;
```

Esta variable establece el estado (ON/OFF) y la regulación porcentual del nivel (0%...100%) del controlador.

Tipo

SNVT_setting

```
typedef struct
{
    setting_t function;
    unsigned short setting;
    signed long rotation;
} SNVT_setting;
```

Margen de valores

```
function
```

Valor	Nemónico	Valor setting	Descripción
-1	SET_NUL	Se ignora	Inválido
2	SET_DOWN	Se ignora	Bajar Persiana
3	SET_UP	Se ignora	Subir Persiana
4	SET_STOP	Se ignora	Parar acción

```
setting
```

0...200 (0%...100%, con resolución 0,5%)
 255 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

```
rotation
```

-17999...18000 (-359,98°...360°)
 32767 Inválido: El valor no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ SET_STOP, 255, 32767 } Parar, inválido, inválido

Perfil Funcional:

Sunblind Actuator

08504 v1.0

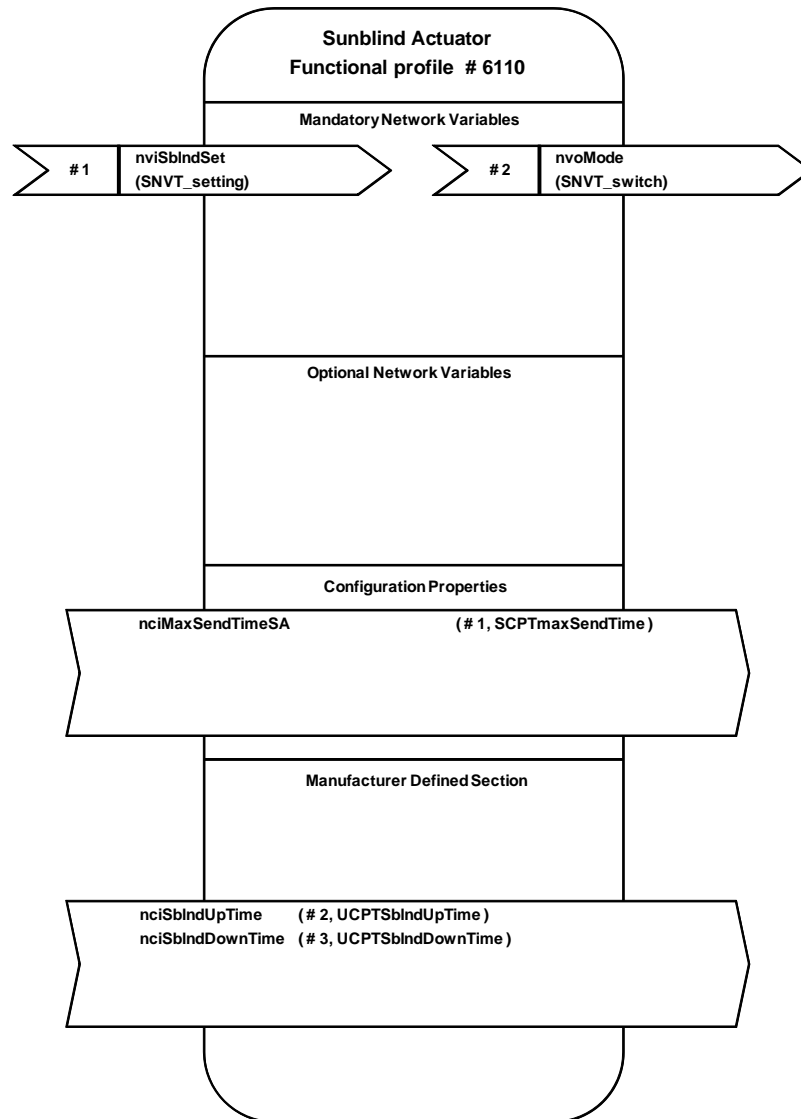
Contenido

1. Descripción	3
2. Interfaz de red	3
3. Variables de red	4
3.1. Variables de entrada.....	4
3.1.1. nviSblndSet.....	4
3.2. Variables de salida	6
3.2.1. nvoMode	6
4. Variables de configuración	7
4.1. nciMaxSendTimeSA	7
4.2. nciSblndUpTime	8
4.3. nciSblndDownTime.....	9

1. Descripción

El objeto *Sunblind Actuator* permite controlar una persiana motorizada y moverla a una posición específica.

2. Interfaz de red



3. Variables de red

3.1. Variables de entrada

3.1.1. nviSblndSet

```
network input SNVT_setting nviSblndSet;
```

Esta variable se utiliza para mover la persiana a la posición deseada. Su valor varía entre 0% y 100%, donde 100% indica que la persiana está completamente cerrada (bajada al extremo inferior)

Tipo

SNVT_setting

```
typedef struct
{
    setting_t function;
    unsigned short setting;
    signed long rotation;
} SNVT_setting;
```

Margen de valores

```
function
```

Valor	Nemónico	Valor setting	Descripción
-1	SET_NUL	Se ignora	No realiza acción alguna
0	SET_OFF	Se ignora	No realiza acción alguna
1	SET_ON	Se ignora	No realiza acción alguna
2	SET_DOWN	INVALID ¹	Ir al extremo inferior
3	SET_UP	INVALID ¹	Ir al extremo superior
4	SET_STOP	Se ignora	Parar movimiento
5	SET_STATE	Absoluto	Ir a posición

¹ El valor setting se basa en el formato de *SNVT_lev_cont*. Por lo tanto, INVALID equivale al valor 255 (0xFF).

```
setting
```

```
SET_STATE    0...200    ( 0%...100%, con resolución 0,5% )
SET_UP      255
```

SET_DOWN 255

rotation

-17999...18000 (-359,98°...360°)

Notas

- La parte decimal del campo `setting`, si la hubiera, se ignora.
- El campo `rotation` se ignora.

3.2. Variables de salida

3.2.1. nvoMode

```
network output SNVT_switch nvoMode;
```

Esta variable proporciona una salida de realimentación para LEDs o dispositivos de monitorización. Su valor varía entre 0% y 100%, donde 100% indica que la persiana está completamente cerrada (bajada al extremo inferior).

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200	(0%...100%, con resolución 0,5%)
255	(127,5%, posición indeterminada después de reset)

state

0	Motor parado
1	Motor en movimiento

Valor por defecto

{ 255, 0 }	(127,5%, la persiana ignora la posición actual)
------------	---

4. Variables de configuración

4.1. *nciMaxSendTimeSA*

```
network input cp SCPTmaxSendTime nciMaxSendTimeSA;
```

Esta propiedad de configuración establece el máximo periodo de tiempo que puede pasar antes de que el perfil funcional propague automáticamente el valor de la variable `nvoMode`.

Tipo

SCPTmaxSendTime, derivado de SNVT_time_sec.

```
typedef unsigned long SNVT_time_sec
```

Margen de valores

0...65530 (0...6553 segundos, con resolución 1 segundo)

Valor por defecto

{ 0 } no se propagan automáticamente las variables.

4.2. *nciSblndUpTime*

```
network input cp UCPTsblndUpTime nciSblndUpTime;
```

Esta propiedad de configuración establece el tiempo que se necesita para mover la persiana desde el extremo inferior hasta el extremo superior.

Tipo

UCPTsblndUpTime, derivado de SNVT_time_sec.

```
typedef unsigned long SNVT_time_sec
```

Margen de valores

0...65534 (0...6553,4 segundos, con resolución 0,1 segundos)

Valor por defecto

{ 300 } 30 segundos

4.3. *nciSblndDownTime*

```
network input cp UCPTsblndDownTime nciSblndDownTime;
```

Esta propiedad de configuración establece el tiempo que se necesita para mover la persiana desde el extremo superior hasta el extremo inferior.

Tipo

UCPTsblndUpTime, derivado de SNVT_time_sec.

```
typedef unsigned long SNVT_time_sec
```

Margen de valores

0...65534 (0...6553,4 segundos, con resolución 0,1 segundos)

Valor por defecto

{ 300 } 30 segundos

Perfil Funcional:

Sunblind Controller

08504 v1.0

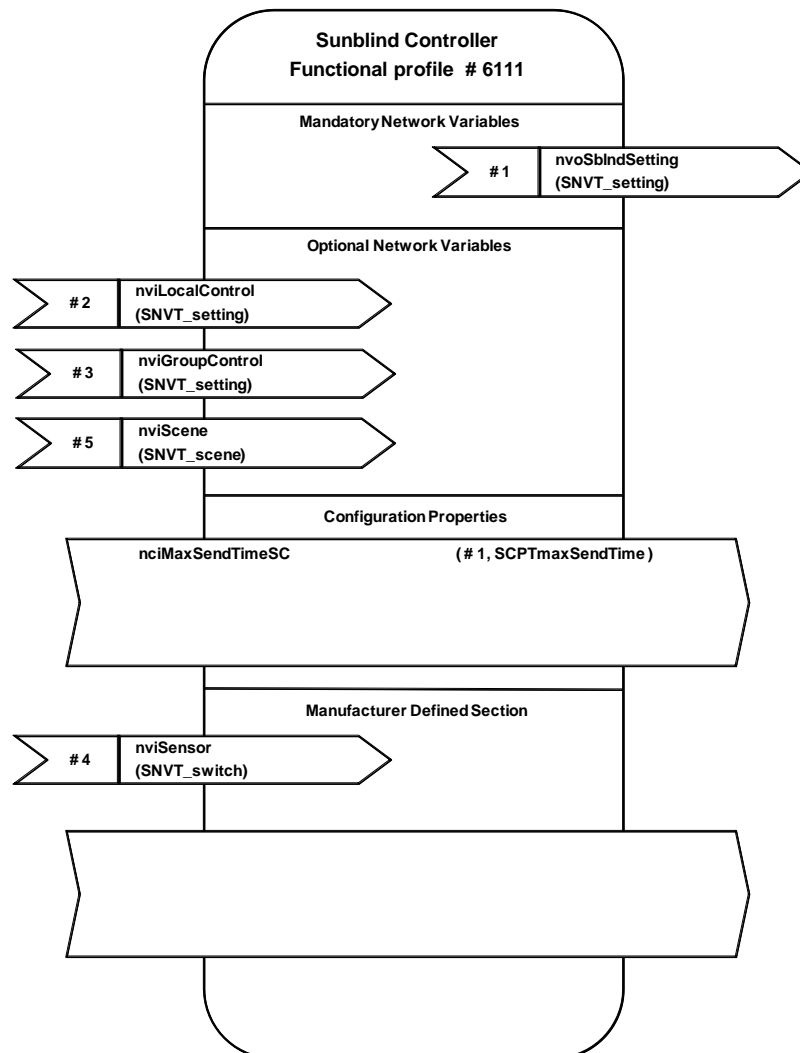
Contenido

1. Descripción	3
2. Interfaz de red	3
3. Variables de red	4
3.1. Variables de entrada.....	4
3.1.1. nviLocalControl	4
3.1.2. nviGroupControl.....	6
3.1.3. nviSensor.....	8
3.1.4. nviScene	9
3.2. Variables de salida	11
3.2.1. nvoSblndSetting.....	11
4. Variables de configuración	13
4.1. nciMaxSendTimeSC	13

1. Descripción

El objeto *Sunblind Controller* genera una salida para objetos *Sunblind Actuator* a partir de diferentes entradas, como pueden ser pulsadores locales, sensores climáticos (lluvia, viento, etc.) y objetos con capacidad de gestionar escenas.

2. Interfaz de red



3. Variables de red

3.1. Variables de entrada

3.1.1. nviLocalControl

```
network input SNVT_setting nviLocalControl;
```

Esta variable establece la posición de la persiana según las órdenes dadas desde un control local. Se transmite automáticamente a nvoSblndSetting.

Tipo

SNVT_setting

```
typedef struct
{
    setting_t function;
    unsigned short setting;
    signed long rotation;
} SNVT_setting;
```

Margen de valores

```
function
```

Valor	Nemónico	Valor setting	Descripción
-1	SET_NUL	Se ignora	No realiza acción alguna
0	SET_OFF	Se ignora	Ir al extremo inferior/Parar ²
1	SET_ON	Se ignora	Ir al extremo superior/Parar ²
2	SET_DOWN	INVALID ¹	Ir al extremo inferior
3	SET_UP	INVALID ¹	Ir al extremo superior
4	SET_STOP	Se ignora	Parar movimiento
5	SET_STATE	Absoluto	Ir a posición

¹ El valor setting se basa en el formato de SNVT_lev_cont. Por lo tanto, INVALID significa el valor 255 (0xFF).

² Los valores SET_OFF y SET_ON se han utilizado para proporcionar una funcionalidad adicional de marcha/paro. En la tabla se muestran las órdenes que envía el objeto *Sunblind Controller* al recibir estos valores por esta variable de red:

function	nvoSblndSetting actual	nvoSblndSetting que se envía
SET_ON	<> SET_STATE, 0%	SET_STATE, 0%
SET_ON	== SET_STATE, 0%	SET_STOP
SET_OFF	<> SET_STATE, 100%	SET_STATE, 100%
SET_OFF	== SET_STATE, 100%	SET_STOP

setting

0...200 (0%...100%, con resolución 0,5%)

rotation

-17999...18000 (-359,98°...360°)

Notas

- La parte decimal del campo `setting`, si la hubiera, se ignora.
- El campo `rotation` se ignora.
- La funcionalidad adicional de marcha/paro que proporcionan los valores `SET_ON` y `SET_OFF` difiere de la especificada por LonMark.

3.1.2. nviGroupControl

```
network input SNVT_setting nviGroupControl;
```

Esta variable establece la posición de la persiana en base a las órdenes procedentes de dispositivos orientados a controlar conjuntos de actuadores. Se transmite automáticamente a nvoSblndSetting.

Tipo

SNVT_setting

```
typedef struct
{
    setting_t function;
    unsigned short setting;
    signed long rotation;
} SNVT_setting;
```

Margen de valores

```
function
```

Valor	Nemónico	Valor setting	Descripción
-1	SET_NUL	Se ignora	No realiza acción alguna
0	SET_OFF	Se ignora	Ir al extremo inferior/Parar ²
1	SET_ON	Se ignora	Ir al extremo superior/Parar ²
2	SET_DOWN	INVALID ¹	Ir al extremo inferior
3	SET_UP	INVALID ¹	Ir al extremo superior
4	SET_STOP	Se ignora	Parar movimiento
5	SET_STATE	Absoluto	Ir a posición

¹ El valor setting se basa en el formato de SNVT_lev_cont. Por lo tanto, INVALID equivale al valor 255 (0xFF).

² Los valores SET_OFF y SET_ON se han utilizado para proporcionar una funcionalidad adicional de marcha/paro. En la tabla se muestran las órdenes que envía el objeto *Sunblind Controller* al recibir estos valores por esta variable de red:

function	nvoSblndSetting actual	nvoSblndSetting que se envía
SET_ON	<> SET_STATE, 0%	SET_STATE, 0%
SET_ON	== SET_STATE, 0%	SET_STOP
SET_OFF	<> SET_STATE, 100%	SET_STATE, 100%
SET_OFF	== SET_STATE, 100%	SET_STOP

setting

0...200 (0%...100%, con resolución 0,5%)

rotation

-17999...18000 (-359,98°...360°)

Notas

- La parte decimal del campo `setting`, si la hubiera, se ignora.
- El campo `rotation` se ignora.
- La funcionalidad adicional de marcha/paro que proporcionan los valores `SET_ON` y `SET_OFF` difiere de la especificada por LonMark.

3.1.3. nviSensor

```
network input SNVT_switch nviSensor;
```

Esta variable permite establecer la posición de la persiana en base a las órdenes recibidas desde sensores (de lluvia, viento, congelación, luminosidad, etc.). Proporciona una entrada estándar para todo tipo de sensor, cuya salida se ajustaría para producir la acción deseada en la persiana. Se transmite automáticamente a `nvoSblndSetting`.

Tipo

SNVT_Switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Parar movimiento

1 Ir a la posición indicada en el campo value

Notas

La parte decimal del campo `value`, si la hubiera, se ignora.

3.1.4. nviScene

```
network input SNVT_scene nviScene;
```

Esta variable carga una escena o almacena el nivel actual en la escena seleccionada. La función `SC_LEARN` almacena de manera permanente el nivel actual en la escena seleccionada. Si la escena a cargar/almacenar no se encuentra en el equipo, éste no realiza acción alguna.

Tipo

SNVT_scene

```
typedef struct
{
    scene_t function;
    unsigned scene_number;
} SNVT_scene;
```

Margen de valores

```
function
```

```
typedef enum scene_t
{
    /* 0 */      SC_RECALL,
    /* 1 */      SC_LEARN,
    ...
    /* -1 */     SC_NUL
} scene_t;
```

SC_RECALL: El equipo adopta inmediatamente el nivel almacenado en la escena establecida en el campo `scene_number`. Si la escena estuviera libre, el equipo no realizaría acción alguna.

SC_LEARN: El equipo almacena el nivel actual en la escena establecida en el campo `scene_number`.

SC_NUL: El equipo no realizará ninguna acción y seguirá actuando igual que lo hacía antes del cambio.

scene_number

1...10 La escena 0 no se usa.

Valor por defecto

{ 0, 0 } Ninguna escena seleccionada.

3.2. Variables de salida

3.2.1. nvoSblndSetting

```
network output SNVT_setting nvoSblndSetting;
```

Esta variable establece la posición de la persiana en base al estado de las entradas del objeto. Normalmente se enlazaré con la entrada de un objeto *Sunblind Actuator*.

Tipo

SNVT_setting

```
typedef struct
{
    setting_t function;
    unsigned short setting;
    signed long rotation;
} SNVT_setting;
```

Margen de valores

```
function
```

Valor	Nemónico	Valor setting	Descripción
-1	SET_NUL	Se ignora	No realiza acción alguna
0	SET_OFF	Se ignora	No realiza acción alguna
1	SET_ON	Se ignora	No realiza acción alguna
2	SET_DOWN	INVALID ¹	Ir al extremo inferior
3	SET_UP	INVALID ¹	Ir al extremo superior
4	SET_STOP	Se ignora	Parar movimiento
5	SET_STATE	Absoluto	Ir a posición

¹ El valor *setting* se basa en el formato de *SNVT_lev_cont*. Por lo tanto, INVALID significa el valor 255 (0xFF).

```
setting
```

0...200 (0%...100%, con resolución 0,5%)

```
rotation
```

-17999...18000 (-359,98°...360°)

Notas

- La parte decimal del campo `setting`, si la hubiera, se ignora.
- El campo `rotation` se ignora.

4. Variables de configuración

4.1. *nciMaxSendTimeSC*

```
network input cp SCPTmaxSendTime nciMaxSendTimeNV;
```

Esta propiedad de configuración establece el máximo periodo de tiempo que puede pasar antes de que el perfil funcional propague automáticamente el valor de la variable `nvoSblndSetting`.

Tipo

SCPTmaxSendTime, derivado de SNVT_time_sec.

```
typedef unsigned long SNVT_time_sec.
```

Margen de valores

0...65530 (0...6553 segundos, con resolución 1 segundo)

Valor por defecto

{ 0 } no se propagan automáticamente las variables

Perfil Funcional:

Fan Coil Unit

08504 v2.2

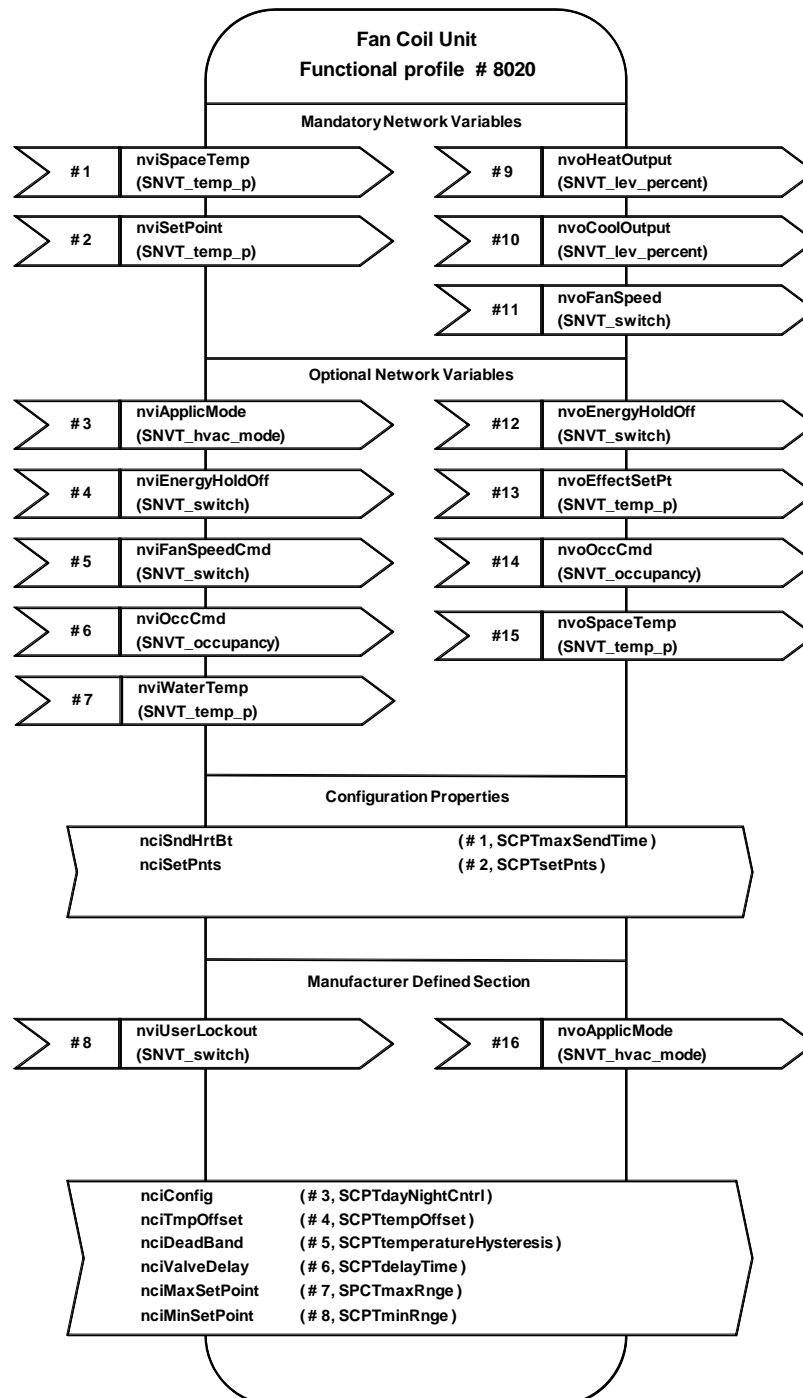
Contenido

1. Descripción.....	3
2. Interfaz de red.....	3
3. Variables de red.....	4
3.1. Variables de entrada.....	4
3.1.1. nviSpaceTemp.....	4
3.1.2. nviSetPoint.....	5
3.1.3. nviApplicMode	6
3.1.4. nviEnergyHoldOff.....	8
3.1.5. nviFanSpeedCmd	9
3.1.6. nviOccCmd	10
3.1.7. nviWaterTemp	12
3.1.8. nviUserLockout	13
3.2. Variables de salida	14
3.2.1. nvoHeatOutput.....	14
3.2.1. nvoCoolOutput.....	15
3.2.2. nvoFanSpeed	16
3.2.3. nvoEnergyHoldOff.....	17
3.2.4. nvoEffectSetPt.....	18
3.2.5. nvoOccCmd.....	19
3.2.6. nvoSpaceTemp.....	20
3.2.7. nvoApplicMode	21
3.3. Variables de configuración.....	22
3.3.1. nciSndHrtBt.....	22
3.3.2. nciSetPnts.....	23
3.3.3. nciConfig.....	25
3.3.4. nciDeadBand	27
3.3.5. nciTmpOffset	28
3.3.6. nciValveDelay	29
3.3.7. nciMaxSetPoint.....	30
3.3.8. nciMinSetPoint.....	31

1. Descripción

El objeto *Fan Coil Unit* se usa para controlar la temperatura de una estancia mediante la activación/desactivación de una salida para válvula de calor y una salida para válvula de frío. También puede controlar las salidas de un ventilador con varias velocidades.

2. Interfaz de red



3. Variables de red

3.1. Variables de entrada

3.1.1. nviSpaceTemp

```
network input SNVT_temp_p nviSpaceTemp;
```

Esta variable permite que otro equipo pueda proporcionar la temperatura ambiente. Si el valor de esta variable no es inválido, su información tiene prioridad sobre la proporcionada por el sensor local del equipo.

Tipo

SNVT_temp_p

```
typedef signed long SNVT_temp_p;
```

Margen de valores

Si `nviConfig.bit0` es 0 (temperatura en grados Celsius)
500...4500 (5,00 °C...45,00 °C, con resolución 0,5°)

Si `nviConfig.bit0` es 1 (temperatura en grados Fahrenheit)
4100...11300 (41,00 °F...113,00 °F, con resolución 1°)

32767 (+327,67° valor inválido), lectura del sensor local.

Valor por defecto

{ 32767 } +327,67°, lectura del sensor local.

3.1.2. nviSetPoint

```
network input SNVT_temp_p nviSetPoint;
```

Esta variable permite cambiar la consigna de temperatura a través de la red. Si su valor no es inválido, su información tiene prioridad sobre las consignas guardadas en `nciSetPnts` si el equipo se halla en los modos de funcionamiento HVAC_AUTO, HVAC_HEAT y/o HVAC_COOL.

Tipo

SNVT_temp_p

```
typedef signed long SNVT_temp_p;
```

Margen de valores

Si `nciConfig.bit0` es 0 (temperatura en grados Celsius)
500...4500 (5,00 °C...45,00 °C, con resolución 0,5°)

Si `nciConfig.bit0` es 1 (temperatura en grados Fahrenheit)
4100...11300 (41,00 °F...113,00 °F, con resolución 1°)

32767 (+327,67° valor inválido)

Valor por defecto

{ 32767 } +327,67°, valor inválido

Notas

El valor efectivo de la variable queda limitado por el valor presente en las variables de configuración `nciMaxSetPoint` y `nciMinSetPoint`.

3.1.3. nviApplicMode

```
network input SNVT_hvac_mode nviApplicMode;
```

Esta variable permite coordinar el objeto Fan Coil Unit con un objeto equipo que proporcione la energía, por ejemplo el agua caliente o fría. Se usa para cambiar entre los modos de frío/calor. Esta variable tiene preferencia sobre el modo establecido localmente en el equipo.

Tipo

SNVT_hvac_mode

```
typedef hvac_t SNVT_hvac_mode;
```

Margen de valores

```
typedef enum hvac_t  
{  
    /* 0 */ HVAC_AUTO = 0,  
    /* 1 */ HVAC_HEAT = 1,  
    /* 3 */ HVAC_COOL = 3,  
    /* 6 */ HVAC_OFF = 6,  
    /* 13 */ HVAC_ECONOMY = 13,  
    ...  
    /* -1 */ HVAC_NUL = -1  
} hvac_t;
```

HVAC_AUTO: El bucle de control cambia automáticamente entre los modos de funcionamiento (p.e FRÍO y CALOR) para conseguir el objetivo fijado por la consigna fijada por el usuario o, en su defecto, la establecida en `nciSetpnts.occupied_xxx`. El equipo **no** se encenderá si el estado anterior era HVAC_OFF.

HVAC_HEAT: El equipo funciona en modo CALOR. La consigna adoptada es la fijada por el usuario o, en su defecto, la establecida en `nciSetpnts.occupied_heat`. El equipo se encenderá si el estado anterior era HVAC_OFF.

HVAC_COOL: El equipo funciona en modo FRÍO. La consigna adoptada es la fijada por el usuario o, en su defecto, la establecida en `nciSetpnts.occupied_cool`.

El equipo se encenderá si el estado anterior era HVAC_OFF.

HVAC_ECONOMY: El equipo entra en modo de bajo consumo, adoptando una de las consignas establecidas en `nciSetpnts.unoccupied_xxx` o `nciSetpnts.standby_xxx`, según el modo de ocupación presente en el equipo. El equipo se encenderá si el estado anterior era HVAC_OFF.

HVAC_OFF: El equipo deja de regular la temperatura y desconecta todas las salidas de válvulas y ventiladores.

HVAC_NUL: El equipo no realizará ninguna acción y seguirá actuando igual que lo hacía antes del cambio.

Según el modo de climatización, se le permiten o se le restringen operaciones al usuario:

HVAC_AUTO	<u>Permite</u> <ul style="list-style-type: none"> • Cambio de modo Frío/Calor • Cambio de consigna • Cambio de velocidad fan-coil • Apagar la climatización
HVAC_COOL HVAC_HEAT	<u>Permite</u> <ul style="list-style-type: none"> • Cambio de consigna • Cambio de velocidad fan-coil • Apagar la climatización <u>No permite</u> <ul style="list-style-type: none"> • Cambio de modo Frío/Calor
HVAC_ECONOMY	<u>Permite</u> <ul style="list-style-type: none"> • Apagar la climatización <u>No permite</u> <ul style="list-style-type: none"> • Cambio de modo Frío/Calor • Cambio de consigna • Cambio de velocidad fan-coil
HVAC_OFF	<u>Permite</u> <ul style="list-style-type: none"> • Encender la climatización <u>No permite</u> <ul style="list-style-type: none"> • Cambio de modo Frío/Calor • Cambio de consigna • Cambio de velocidad fan-coil

Valor por defecto

{ 0 } HVAC_AUTO, el equipo establecerá el modo de funcionamiento necesario para seguir la consigna.

3.1.4. nviEnergyHoldOff

```
network input SNVT_switch nviEnergyHoldOff;
```

Esta variable de entrada permite compartir entre varios equipos la información proporcionada por un sensor de contacto instalado en una ventana. Al activarse la variable, el equipo detiene la climatización (paro del ventilador y apertura de la válvula) hasta que se desactiva la entrada.

Si su valor no es inválido, su información tiene preferencia sobre la del sensor local del equipo.

Tipo

```
SNVT_switch  
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0 .. 200 (0% .. 100%, con resolución 0,5%)

state

0	Apagado
1	Encendido (Nota: Si value = 0, se interpreta como Apagado)
-1	Inválido: Lectura proporcionada por el sensor local.

Valor por defecto

{ 0, -1 } Lectura proporcionada por el sensor local.

3.1.5. nviFanSpeedCmd

```
network input SNVT_switch nviFanSpeedCmd;
```

Esta variable de entrada permite conectar un equipo externo para controlar la velocidad del ventilador o permitir que un dispositivo supervisor pueda sobrescribir la velocidad del ventilador establecida por el algoritmo de climatización del equipo.

Tipo

```
SNVT_switch
typedef struct
{
    unsigned value;
    signed state;
} SNVT_switch;
```

Margen de valores

state	value	Velocidad del Ventilador
0 (OFF)	no aplic.	Parado
1 (ON)	0% ó >100%	Parado
1 (ON)	0,5% - 33%	Velocidad LOW
1 (ON)	33,5% - 66%	Velocidad MIDDLE
1 (ON)	66,5% - 100%	Velocidad HIGH
-1	no aplic.	Velocidad AUTO, velocidad controlada por el algoritmo de climatización.

Valor por defecto

{ 0, -1 } Velocidad AUTO, el equipo determina la velocidad del ventilador en función de las necesidades de climatización.

3.1.6. nviOccCmd

```
network input SNVT_occupancy nviOccCmd;
```

Esta variable permite cambiar el comportamiento del equipo en función del estado de ocupación.

Tipo

SNVT_occupancy

```
typedef occup_t SNVT_occupancy;
```

Margen de valores

```
typedef enum occup_t
{
    /* 0 */ OC_OCCUPIED,
    /* 1 */ OC_UNOCCUPIED,
    /* 2 */ OC_BYPASS,
    /* 3 */ OC_STANDBY,
    /* -1 */ OC_NUL = -1
} occup_t;
```

OC_OCCUPIED: El equipo adopta uno de los siguientes modos de funcionamiento: HVAC_AUTO, HVAC_COOL o HVAC_HEAT.

OC_UNOCCUPIED: El equipo adopta el modo de funcionamiento HVAC_ECONOMY si el bit 2 de la variable nciConfig está activado, HVAC_OFF en caso contrario. Si el modo previo al cambio de estado fuera HVAC_OFF, el equipo continuaría en este modo en cualquier caso.

OC_BYPASS: El equipo pasa temporalmente a OC_OCCUPIED y se inicia el temporizador de *bypass*. Al expirar, el equipo vuelve al estado OC_UNOCCUPIED.

OC_STANDBY: El equipo adopta el modo de funcionamiento HVAC_ECONOMY si el modo actual es distinto de HVAC_OFF.

OC_NUL: El equipo continúa en el modo de funcionamiento que tenía previamente.

Valor por defecto

{ -1 }

OC_NUL

3.1.7. nviWaterTemp

```
network input SNVT_temp_p nviWaterTemp;
```

Esta variable permite el cambio de modo automático entre FRÍO/CALOR en función de la temperatura del agua que proporcione el sistema de climatización. Si el valor de esta variable no es inválido, su información tiene prioridad sobre la proporcionada por el sensor local del equipo.

Tipo

SNVT_temp_p

```
typedef signed long SNVT_temp_p;
```

Margen de valores

Si `nciConfig.bit0` es 0 (temperatura en grados Celsius)
500...4500 (5,00 °C...45,00 °C, con resolución 0,5°)

Si `nciConfig.bit0` es 1 (temperatura en grados Fahrenheit)
4100...11300 (41,00 °F...113,00 °F, con resolución 1°)

32767 (+327,67° valor inválido) lectura del sensor local.

Valor por defecto

{ 32767 } +327,67°, lectura del sensor local

3.1.8. nviUserLockout

```
network input SNVT_switch nviUserLockout;
```

Esta variable permite bloquear el teclado y el receptor IR del equipo para evitar que los usuarios manipulen el equipo. Cuando está activada, el usuario no puede cambiar la temperatura de consigna, el modo de funcionamiento (FRÍO/CALOR, ON/OFF) ni la velocidad de los ventiladores.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Bloqueo Desactivado

1 Bloqueo Activado (**Nota:** Si value = 0, se interpreta como Desactivado)

-1 Bloqueo Desactivado

Valor por defecto

{ 0, -1 } Bloqueo Desactivado

3.2. Variables de salida

3.2.1. nvoHeatOutput

```
network output SNVT_lev_percent nvoHeatOutput;
```

Esta variable indica la posición de la válvula de calor y puede utilizarse para controlar un equipo actuador o, en caso de control local, para monitorización. Cuando el objeto Fan Coil Unit esté configurado para una instalación a 2 tubos (una sola válvula), esta variable sirve tanto para controlar la válvula en modo CALOR como en modo FRÍO.

Tipo

SNVT_lev_percent

```
typedef signed long SNVT_lev_percent;
```

Margen de valores

0...20000 (0%...100%, con resolución 0,005%)

Valor por defecto

{ 0 } 0%, válvula desactivada.

3.2.1. nvoCoolOutput

```
network output SNVT_lev_percent nvoCoolOutput;
```

Esta variable indica la posición de la válvula de frío y puede utilizarse para controlar un equipo actuador o, en caso de control local, para monitorización.

Tipo

SNVT_lev_percent

```
typedef signed long SNVT_lev_percent;
```

Margen de valores

0...20000 (0%...100%, con resolución 0,005%)

Valor por defecto

{ 0 } 0%, válvula desactivada.

3.2.2. nvoFanSpeed

```
network output SNVT_switch nvoFanSpeed;
```

Esta variable refleja la velocidad actual proporcionada por el equipo. Puede ser usada para establecer un bucle de control o para monitorización.

Tipo

SNVT_switch

```
typedef struct
{
    unsigned value;
    signed state;
} SNVT_switch;
```

Margen de valores

state	value	Velocidad del Ventilador
0 (OFF)	no aplic.	Parado
1 (ON)	0%	Parado
1 (ON)	33%	Velocidad LOW
1 (ON)	66%	Velocidad MIDDLE
1 (ON)	100%	Velocidad HIGH
0xFF	no aplic.	Velocidad AUTO, velocidad controlada por el algoritmo de climatización.

Valor por defecto

{ 0, 0 } Ventiladores parados.

3.2.3. nvoEnergyHoldOff

```
network output SNVT_switch nvoEnergyHoldOff;
```

Esta variable se usa para comunicar a otros dispositivos el estado de un sensor de contacto local instalado en una ventana, o retransmitir el valor de nviEnergyHoldOff.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value

0...200 (0%...100%, con resolución 0,5%)

state

0 Apagado

1 Encendido (**Nota:** Si value = 0, se interpreta como Apagado)

-1 Inválido: El valor leído no es válido o no es fiable. No realizar ninguna acción.

Valor por defecto

{ 0, 0 } Energy Hold Off desactivado.

3.2.4. nvoEffectSetPt

```
network output SNVT_temp_p nvoEffectSetPt;
```

Esta variable permite monitorizar la consigna de temperatura que está aplicando el equipo, que puede depender del usuario o del modo de climatización activo.

Tipo

SNVT_temp_p

```
typedef signed long SNVT_temp_p;
```

Margen de valores

Si `nciConfig.bit0` es 0 (temperatura en grados Celsius)
500...4500 (5,00 °C...45,00 °C, con resolución 0,5°)

Si `nciConfig.bit0` es 1 (temperatura en grados Fahrenheit)
4100...11300 (41,00 °F...113,00 °F, con resolución 1°)

32767 (+327,67° valor inválido)

Valor por defecto

{ 32767 } +327,67°, valor inválido

3.2.5. nvoOccCmd

```
network output SNVT_occupancy nvoOccCmd;
```

Esta variable permite monitorizar el estado de ocupación bajo el que trabaja el equipo.

Tipo

SNVT_occupancy

```
typedef occup_t SNVT_occupancy;
```

Margen de valores

```
typedef enum occup_t  
{  
    /* 0 */ OC_OCCUPIED,  
    /* 1 */ OC_UNOCCUPIED,  
    /* 2 */ OC_BYPASS,  
    /* 3 */ OC_STANDBY,  
    /* -1 */ OC_NUL  
} occup_t;
```

Valor por defecto

```
{-1}          OC_NUL
```

3.2.6. nvoSpaceTemp

```
network output SNVT_temp_p nvoSpaceTemp;
```

Esta variable permite monitorizar la temperatura proporcionada por el sensor local del equipo o la temperatura recibida a través de nviSpaceTemp.

Tipo

SNVT_temp_p

```
typedef signed long SNVT_temp_p;
```

Margen de valores

Si nciConfig.bit0 es 0 (temperatura en grados Celsius)
500...4500 (5,00 °C...45,00 °C, con resolución 0,5°)

Si nciConfig.bit0 es 1 (temperatura en grados Fahrenheit)
4100...11300 (41,00 °F...113,00 °F, con resolución 1°)

32767 (+327,67°, valor inválido)

Valor por defecto

{ 32767 } +327,67°, valor inválido

3.2.7. nvoApplicMode

```
network output SNVT_hvac_mode nvoApplicMode;
```

Esta variable permite coordinar el objeto Fan Coil Unit con otro equipo que proporcione la energía, por ejemplo el agua caliente o fría. Refleja el modo de climatización en el que opera el equipo.

Tipo

SNVT_hvac_mode

```
typedef hvac_t SNVT_hvac_mode;
```

Margen de valores

```
typedef enum hvac_t  
{  
    /* 1 */ HVAC_HEAT = 1,  
    /* 3 */ HVAC_COOL = 3,  
    /* 6 */ HVAC_OFF = 6,  
} hvac_t;
```

Valor por defecto

```
{ 6 } HVAC_OFF.
```

3.3. Variables de configuración

3.3.1. nciSndHrtBt

```
network input SCPTmaxSendTime cp nciSndHrtBt;
```

Esta propiedad de configuración establece el máximo periodo de tiempo que puede pasar antes de que el perfil funcional propague automáticamente los valores de las variables de salida enlazadas.

Tipo

SNVT_time_sec

```
typedef unsigned long SNVT_time_sec;
```

Margen de valores

0...6553 (0...6553 segundos, con resolución 1 segundo)

Valor por defecto

{ 0 } no se propagan automáticamente las variables de salida.

3.3.2. nciSetPnts

```
network input SCPTsetPnts cp nciSetPnts;
```

Esta propiedad de configuración define las consignas a aplicar para los modos frío/calor.

Tipo

SCPTsetPnts, derivado de SNVT_time_setpt

```
typedef struct
{
    signed long occupied_cool;
    signed long standby_cool;
    signed long unoccupied_cool;
    signed long occupied_heat;
    signed long standby_heat;
    signed long unoccupied_heat;
} SNVT_temp_setpt;
```

Margen de valores

Si `nciConfig.bit0` es 0 (temperatura en grados Celsius)
500...4500 (5,00 °C...45,00 °C, con resolución 0,5°)

Si `nciConfig.bit0` es 1 (temperatura en grados Fahrenheit)
4100...11300 (41,00 °F...113,00 °F, con resolución 1°)

Valor por defecto

<code>occupied_cool</code>	{ 2300 }	(23 °C)
<code>standby_cool</code>	{ 2500 }	(25 °C)
<code>unoccupied_cool</code>	{ 2800 }	(28 °C)
<code>occupied_heat</code>	{ 2100 }	(21 °C)
<code>standby_heat</code>	{ 1900 }	(19 °C)
<code>unoccupied_heat</code>	{ 1600 }	(16 °C)

Notas

- Los valores `occupied_cool/heat` son los que se utilizan por defecto al iniciar el equipo y/o el modo se ocupación sea `OC_OCCUPIED` o `OC_BYPASS`.

- Los valores `standby_cool/heat` se utilizan cuando el equipo entra en el modo de bajo consumo (`HVAC_ECONOMY`) y el modo de ocupación sea `OC_STANDBY`.
- Los valores `unoccupied_cool/heat` se utilizan cuando el equipo entra en el modo de bajo consumo (`HVAC_ECONOMY`) y el modo de ocupación sea distinto de `OC_STANDBY`.

3.3.3. nciConfig

```
network input SCPTdayNightCntrl cp nciConfig;
```

Esta propiedad de configuración establece las diferentes opciones de funcionamiento del equipo.

Tipo

SCPTdayNightCntrl, derivado de SNVT_state

```
typedef struct
{
    unsigned bit0 : 1;
    unsigned bit1 : 1;
    unsigned bit2 : 1;
    unsigned bit3 : 1;
    unsigned bit4 : 1;
    unsigned bit5 : 1;
    unsigned bit6 : 1;
    unsigned bit7 : 1;
    unsigned bit8 : 1;
    unsigned bit9 : 1;
    unsigned bit10 : 1;
    unsigned bit11 : 1;
    unsigned bit12 : 1;
    unsigned bit13 : 1;
    unsigned bit14 : 1;
    unsigned bit15 : 1;
} SNVT_state;
```

Margen de valores

Bit #	Descripción	"0"	"1"
0	Unidades de Temperatura	[° Celsius]	° Fahrenheit
1	Núm. Velocidades Ventilador	[3 Velocidades]	1 Velocidad
2	Modo climatización en estado Desocupado (p.e. sin tarjeta)	[Paro]	Bajo Consumo
3	Comportamiento Entrada Analógica 1	[Entrada Auxiliar 2]	Cambio Automático de modo por Temperatura del Agua
4	Cambio Automático de modo por Consigna	[Desactivado]	Activado
5	Ventilador en marcha sin Demanda Frío	Desactivado	[Activado]
6	Ventilador en marcha sin Demanda Calor	Desactivado	[Activado]
7	Valor a mostrar en Display	Temperatura	[Consigna]
8 .. 15	Reservado	-	-

Valor por defecto

{ 0, 0, 0, 0, 0, 1, 1, 1 ... }

Los marcados entre corchetes en el apartado anterior.

3.3.4. nciDeadBand

```
network input SCPTtemperatureHysteresis cp nciDeadBand;
```

Esta propiedad de configuración establece la diferencia de grados que debe existir entre la temperatura ambiente y la consigna para que el equipo cambie automáticamente a modo frío/calor, si está configurado para ello.

Tipo

SCPTtemperatureHysteresis, derivado de SNVT_temp_p

```
typedef signed long SNVT_temp_p;
```

Margen de valores

Si `nciConfig.bit0` es 0 (temperatura en grados Celsius)
0...1000 (0,00 °C...10,00 °C)

Si `nciConfig.bit0` es 1 (temperatura en grados Fahrenheit)
0...5000 (0,00 °F...50,00 °F)

Valor por defecto

{ 150 } 1,50 °C

3.3.5. nciTmpOffset

```
network input SCPTtempOffset cp nciTmpOffset;
```

Esta propiedad de configuración se usa para ajustar el offset de la temperatura ambiente medida por el equipo.

Tipo

SCPTtempOffset, derivado de SNVT_temp_p

```
typedef signed long SNVT_temp_p;
```

Margen de valores

Si `nciConfig.bit0` es 0 (temperatura en grados Celsius)
-1000...1000 (-10,00 °C...10,00 °C)

Si `nciConfig.bit0` es 1 (temperatura en grados Fahrenheit)
-5000...5000 (-50,00 °F...50,00 °F)

Valor por defecto

{ 0 } 0,00 °C

3.3.6. nciValveDelay

```
network input SCPTdelayTime cp nciValveDelay;
```

Esta propiedad de configuración establece el tiempo de cierre de las electroválvulas. Al esperar a que se cierre la electroválvula antes de que se abra la otra, se evita el paso de fluidos entre los dos circuitos y se prolonga la vida útil de las mismas al evitar que un usuario pueda cambiar constantemente de modo.

Tipo

SCPTdelayTime, derivado de SNVT_time_sec

```
typedef unsigned long SNVT_time_sec;
```

Margen de valores

10...65530 (1 segundo...6553 segundos)

Valor por defecto

{ 300 } 30 segundos

Notas

Esta variable no permite valores inferiores a 1 segundo. Si se intentara establecer un valor inferior a 1 segundo, el equipo automáticamente establecerá el valor equivalente a 1 segundo.

3.3.7. nciMaxSetPoint

```
network input SCPTmaxRnge cp nciMaxSetpoint;
```

Esta propiedad de configuración establece la consigna máxima que se puede establecer en el equipo.

Tipo

SCPTmaxRnge, derivado de SNVT_temp_p

```
typedef signed long SNVT_temp_p;
```

Margen de valores

Si `nciConfig.bit0` es 0 (temperatura en grados Celsius)
500...4500 (5,00 °C...45,00 °C, con resolución 0,5°)

Si `nciConfig.bit0` es 1 (temperatura en grados Fahrenheit)
4100...11300 (41,00 °F...113,00 °F, con resolución 1°)

Valor por defecto

{ 3200 } 32,00 °C

3.3.8. nciMinSetPoint

```
network input SCPTminRnge cp nciMinSetPoint;
```

Esta propiedad de configuración establece la consigna máxima que se puede establecer en el equipo.

Tipo

SCPTminRnge, derivado de SNVT_temp_p

```
typedef signed long SNVT_temp_p;
```

Margen de valores

Si `nciConfig.bit0` es 0 (temperatura en grados Celsius)
500...4500 (5,00 °C .. 45,00 °C, con resolución 0,5°)

Si `nciConfig.bit0` es 1 (temperatura en grados Fahrenheit)
4100...11300 (41,00 °F .. 113,00 °F, con resolución 1°)

Valor por defecto

{ 1500 } 15,00 °C

Perfil Funcional:

Room Medic Alarm

08504 v1.0

Contenido

1. Descripción	3
1.1. Funcionamiento	3
2. Interfaz de red	4
3. Variables de red	5
3.1. Variables de salida	5
3.1.1. nvoMedicAlarm	5
3.1.2. nvoMedicCall	6

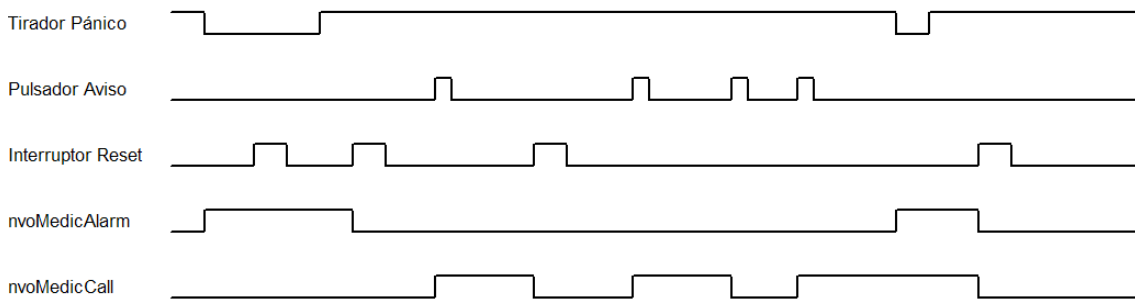
1. Descripción

El objeto *Room Medic Alarm* es un objeto sensor que se usa para indicar necesidades y emergencias médicas a un centro de control.

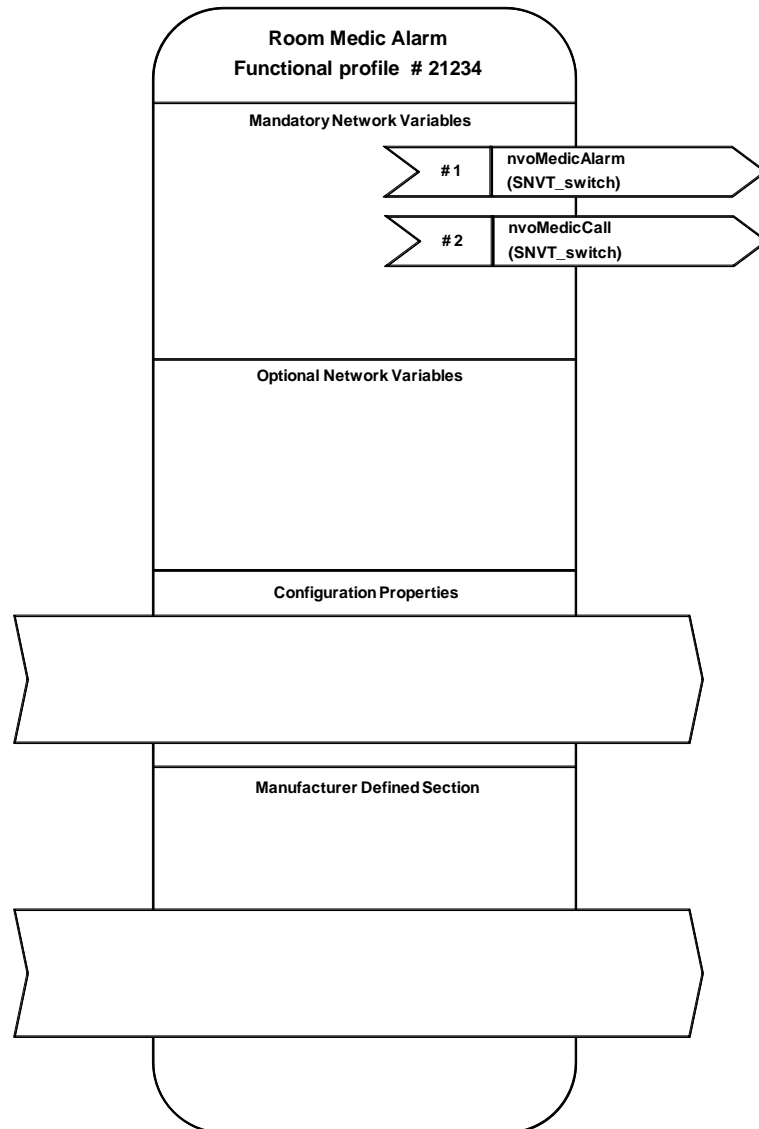
1.1. Funcionamiento

La salida `nvoMedicAlarm` se activa al tirar de un interruptor/tirador instalado en la estancia, y se resetea mediante otro interruptor que normalmente sólo puede ser manipulado por personal autorizado. La variable no se resetea si la entrada aún presenta condición de alarma.

La salida `nvoMedicCall` es un aviso menos prioritario, pudiéndose activar y desactivar mediante el mismo pulsador. También se puede resetear mediante la entrada de reset mencionada anteriormente.



2. Interfaz de red



3. Variables de red

3.1. Variables de salida

3.1.1. nvoMedicAlarm

```
network output SNVT_switch nvoMedicAlarm;
```

Esta variable indica si se ha producido una alarma médica en la estancia. Además de ser monitorizada, suele enlazarse a un objeto actuador que produzca una señal luminosa o acústica de la condición de alarma.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value	state	Descripción
0	0	Sin alarma.
200	1	Condición de alarma.

Valor por defecto

{ 0, 0 } Sin alarma

3.1.2. nvoMedicCall

```
network output SNVT_switch nvoMedicCall;
```

Esta variable indica la necesidad de atención médica por parte del personal sanitario.

Tipo

SNVT_switch

```
typedef struct  
{  
    unsigned value;  
    signed state;  
} SNVT_switch;
```

Margen de valores

value	state	Descripción
0	0	No se requiere asistencia.
200	1	Se requiere asistencia.

Valor por defecto

{ 0, 0 } No se requiere asistencia