Beschreibung der Software LONMPM-8DI/DO-Anwendung Version 16911210

Schnittstellenbeschreibung

1 Allgemeines

1.1 Gerätebeschreibung

Die Applikation 16911210 ist eine Software für den 3150 Prozessor des LONWORKS-Gerätes LONMPM 8DI/DO der WAREMA Renkhoff SE.
Bis zu 8 digitale Eingänge können ausgewertet und deren Zustand über das Netz gesendet werden. Das Ergebnis der Überwachung und Verknüpfung dieser Eingangsinformation oder von Netzwerkvariablen kann über weitere Netzwerkvariable gesendet werden bzw. dient der Ansteuerung von bis zu 8 digitalen Ausgänge. Ein weiterer separater Teil der Applikation kann zu verschiedenen bestimmten Zeitpunkten unterschiedliche Szenen senden.
Die Kommunikation über das LON erfolgt ausschließlich über Netzwerkvariablen bzw. LONTALK Network Management /Diagnostic Messages.
Alle Applikationsparameter können über eine Netzwerkvariable `nviConfig` eingestellt werden, alternativ über Configuration Properties.

1.2 Typografie

Standardtext in Arial Standard 10 pt
Dokumentenverweise in Arial Kursiv 10 pt
`Auszüge aus NEURON-C-Quelltexten in Courier New Standard 10 pt`
Hintergrundschattierungen in umfangreichen Aufzählungen oder Tabellen sollen deren Lesbarkeit erhöhen.

Das Layout dieses Dokuments ist an an das Layout der LONMARK Application Layer Interoperability Guidelines Version 3.0 angelehnt.
Derzeit stellen die am Markt üblichen LONWORKS-Inbetriebnahme-Tools die Daten der Netzwerkvariablen unterschiedlich dar. Zur größtmöglichen Transparenz werden diese Daten hier weitestgehend hexadezimal codiert dargestellt.

1.3 Weitere Dokumentationen

- Die *SNVT Master List and Programmers Guide May 1997* beschreibt Wertebereiche, Einheiten, und Auflösung aller Standard Netzwerkvariablen Typen (SNVT‘s)
- Die *LONMARK Application Layer Interoperability Guidelines Version 3.0* liefern Design-Richtlinien für Objekte auf LONWORKS-Geräten.
- Die *Anwendungsbeispiele der Applikation 16911210* beschreiben unterschiedliche Einsatzfälle des Gerätes LONMPM 8DI/DO mit der Appikation 16911210.

1.4 Einschränkungen

Soweit nicht anders beschrieben, sind alle hier aufgeführten Datenformate und Funktionalitäten entsprechend LONMARK-Konventionen bzw. *SNVT Master List and Programmers Guide May 1997* zu verstehen.
Grundkenntnisse der Programmiersprachen C bzw. Neuron C sind zum Verständnis einzelner Dokumentationsteile erforderlich.

1.5 Inbetriebnahmehinweis

Die korrekte Funktion der Applikation kann – z.B. nach erfolgter Parametrierung – nur erfolgen, wenn mindestens eine NV des LONMPM 8DI/DO gebunden ist.

1.6 Objekte

Ein Objekt stellt eine geschlossene abstrakte Einheit aus Verhalten (Funktionalität) und Eigenschaften (Daten) dar; es kann Schnittstellen zur Kommunikation mit der Außenwelt (z.B. andere Objekte) zur Verfügung stellen. Als Basis der Objektdefinitionen dienen die generischen Objekte, wie sie in den *LONMARK Application Layer Interoperability Guidelines Version 3.0* beschrieben sind.
In der Applikation 16911210 sind mehrere Objekte implementiert:

- das Node Objekt (#0)
- 8 Switch Objekte (#3200)
- 4 Scene Panel Objekte (#3250)
- 2 Controller Objekte (#5)
- 1 Scene Scheduler (#20003)

Alle Objekte sind gleichzeitig auf einem Gerät implementiert. Davon können einige untereinander und/oder auf Objekte entfernter Geräte gebunden werden.

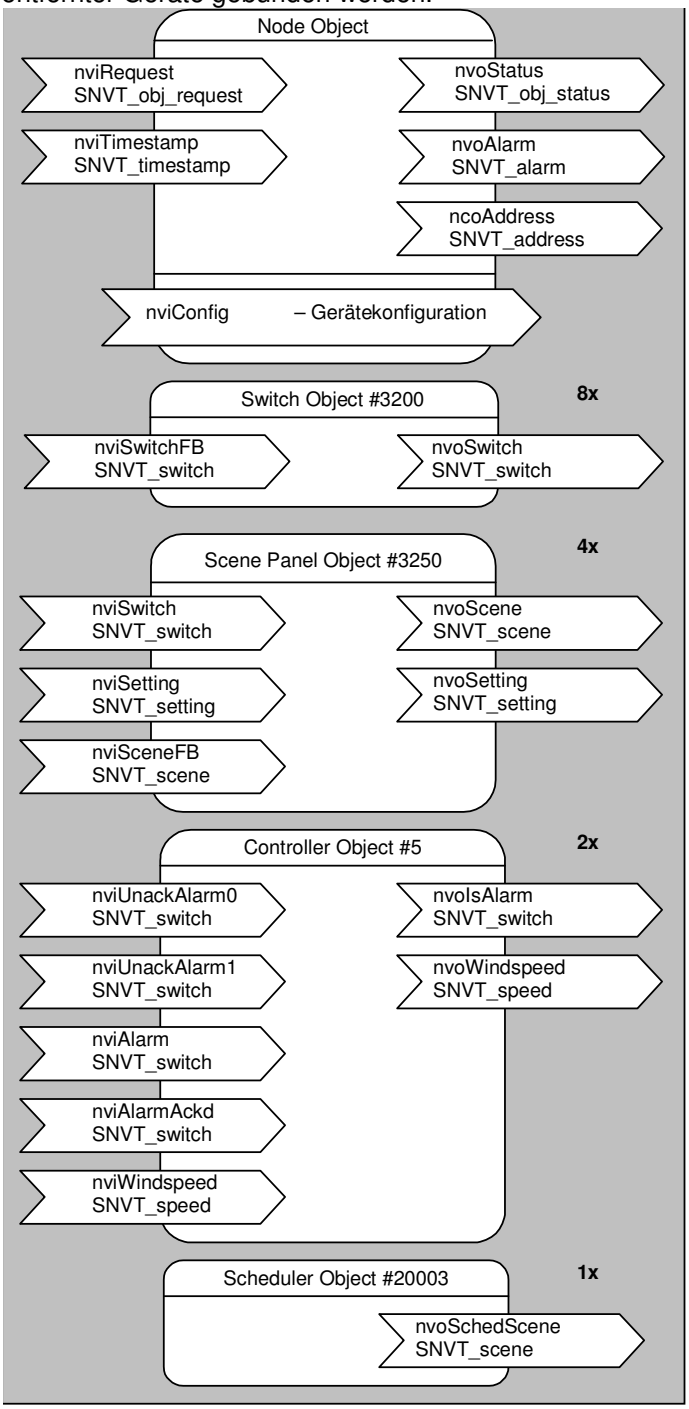

Abbildung 1 (Gesamtübersicht Interface)

1.7 Herstellerdefinierte Netzwerkvariablenformate

```
typedef struct {
        SNVT_time_sec snd;    // Send Heartbeat nvoAlarm
        SNVT_time_sec rcv;    // Receiver Heartbeat nviTimestamp
} tNodeHeartbeats;

typedef struct {
      unsigned short type;      // Verknüpfungsart feedback nv - io
      SNVT_time_sec  sndHrtbt;
} tSwitchCfg;


typedef struct {
      boolean       bLearn;     // Lernen Ein/Aus
      SNVT_time_sec sndHrtbt;   // Send Heartbeat nvoScene / nvoSetting
      unsigned int  numX;       // Szene X (Taste gedrückt)
      unsigned int  numY;       // Szene Y (Taste losgelassen)
      SNVT_setting  settingX;   // Setting X (Taste gedrückt)
      SNVT_setting  settingY;   // Setting Y (Taste losgelassen)
} tSceneCfg;


typedef struct {
      SNVT_speed    upperLimitWind;
      SNVT_speed    lowerLimitWind;
      SNVT_time_sec rcvHrtbtUnackAlarm0;
      SNVT_time_sec rcvHrtbtWind;
      SNVT_time_sec rcvHrtbtUnackAlarm1;
      SNVT_time_sec rcvHrtbtAckAlarm;
      SNVT_time_sec sndHrtbt;
      tMonitor      monitorFilter;   // was wird überwacht
      tCtrlDisplay  displayFilter;   // Zustand von nviAlarm oder
} tControllerConfig;                 // nvoIsAlarm anzeigen


typedef struct {
      unsigned int iItem;              // Index Parametersatz
      union {
          SNVT_time_stamp dayDate;
          tDayOfWeek      dayOfWeek;
          tWeekday        weekDay;
      } start;                         // Startzeit zum Senden der scene
      union {
          SNVT_time_stamp dayDate;
          tDayOfWeek      dayOfWeek;
          tWeekday        weekDay;
      } end;                           // Abbruchzeitpunkt
      SNVT_scene    scene;             // zu sendende Szene
      SNVT_time_sec sndHrtbt;          // send heartbeat
} tRtbSchedulerItem;

typedef struct {
    scene_config_t  request;
    tObjId          objectIndex;
    tConfigType     configType;
    union {
        tNodeHeartbeats     nodeHeartbeats;
        tRtbSchedulerItem   schedulerItem;
        tControllerConfig   ctrl;
        tScenePanelCfg      scpCfg;
        tIoDisplay          display;
        tSwitchConfig       switchCfg;
        unsigned short      raw[28];
    } data;
} UNVT_cfg;
```

2 Node Object

Das Node Objekt stellt Mechanismen zum Ändern und Abfragen von Zuständen aller Objekte eines Gerätes zur Verfügung. Es verfügt über Netzwerkvariablen, die nicht nur für ein einzelnes Objekt sondern das ganze Gerät gelten.

2.1 Schnittstellenbeschreibung

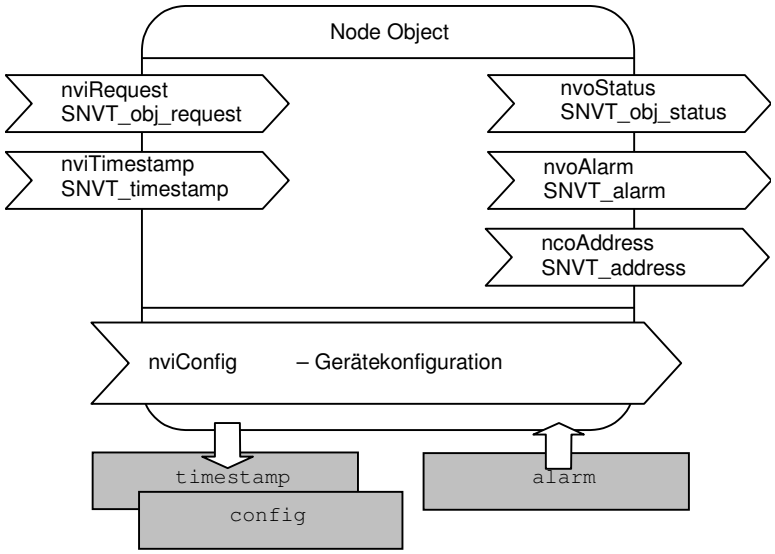

Abbildung 2 (Node Objekt)

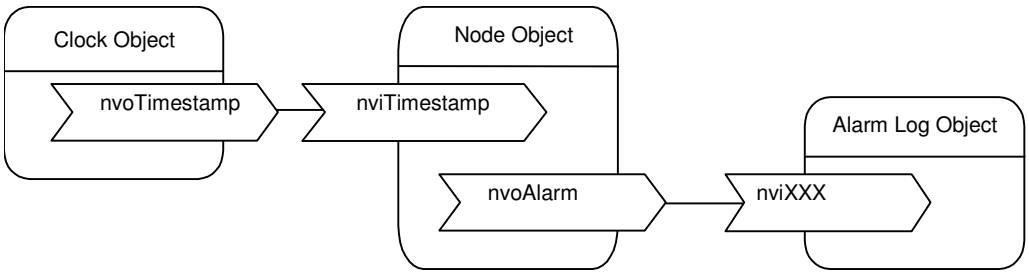

Abbildung 3 (Beispiel-Bindings mit Node Objekt)

2.1.1 Object Request

```
network input SNVT_obj_request nviRequest;
```

Diese NV unterstützt den Mechanismus zum Ändern und Abfragen von Zuständen aller Objekte des Gerätes. Unterstützte Dienste sind:

- `RQ_DISABLED`
- `RQ_NORMAL`
- `RQ_REPORT_MASK`
- `RQ_UPDATE_STATUS`
- `RQ_ALARM_NOTIFY_DISABLED`
- `RQ_ALARM_NOTIFY_ENABLED`
- `RQ_CLEAR_ALARM`
- `RQ_UPDATE_ALARM`

Wird ein gültiger Dienst auf ein existierendes Objekt ausgeführt, wird in `nvoStatus` der Status dieses Objektes zurückgemeldet bzw. `nvoAlarm` wird mit dem Alarmzustand des abgefragten Objektes aktualisiert. Bei nicht erlaubten Diensten bzw. Referenzierung nicht vorhandener Objekte liefert `nvoStatus invalid_request` bzw. `invalid_id` zurück.
Änderungen des Zustandes des Node Objektes führen zu denselben Änderungen aller anderen Objekte des Gerätes.

Valid Range

Gültige Werte sind alle Werte gemäß Tab. 1. Die Bedeutung der Felder der `SNVT_obj_request` ist erklärt in *[1]*.

Unterstützte Dienste `nviRequest.object_request`	**Wert**	**Bedeutung**
`RQ_DISABLED`	1	das Objekt wird gesperrt und kann keine Aktionen mehr ausführen
`RQ_UPDATE_STATUS`	2	das Objekt meldet seinen aktuellen Zustand
`RQ_NORMAL`	0	das Objekt wird freigegeben und kann Aktionen ausführen
`RQ_REPORT_MASK`	5	ein Objekt wird aufgefordert, seine unterstützten Status Bits zurückzuliefern
`RQ_CLEAR_STATUS`	9	löschen aller Statusbits und Löschergebnis zurückliefern
`RQ_ALARM_NOTIFY_DISABLED`	12	Alarmmeldung eines Objektes sperren
`RQ_ALARM_NOTIFY_ENABLED`	11	Alarmmeldung eines Objektes freigeben
`RQ_CLEAR_ALARM`	10	aktuellen Alarmzustand eines Objektes zurücksetzen
`RQ_UPDATE_ALARM`	4	Alarmzustand eines Objektes aktualisieren und melden

Tab. 1

Objekt	**Objektindex** `nviRequest.object_id`	**Bezeichnung gemäß** `tObjId`
Node	0	`OI_NODE`
Switch 1	1	`OI_SWITCH_0`
Switch 2	2	`OI_SWITCH_1`
Switch 3	3	`OI_SWITCH_2`
Switch 4	4	`OI_SWITCH_3`
Switch 5	5	`OI_SWITCH_4`
Switch 6	6	`OI_SWITCH_5`
Switch 7	7	`OI_SWITCH_6`
Switch 8	8	`OI_SWITCH_7`
Scene Panel 1	9	`OI_SCP_0`
Scene Panel 2	10	`OI_SCP_1`
Scene Panel 3	11	`OI_SCP_2`
Scene Panel 4	12	`OI_SCP_3`
Alarm Controller 1	13	`OI_CTRL_0`
Alarm Controller 2	14	`OI_CTRL_1`
Scheduler	15	`OI_SCHEDULER_0`

Tab. 2 (Objekt-Kennungen)

Im weiteren gilt: innerhalb einer Objektgruppe beginnt die Zählung immer mit 0.

Beispiel

Um das 4. Switch Objekt zu sperren, aktualisiere `nviRequest` mit `00 04 01`.

2.1.2 Object Status

```
network output SNVT_obj_status nvoStatus;
```

Diese NV liefert den Zustand eines Objektes des Gerätes.

Valid Range

Gültige Werte sind alle Werte in den definierten Grenzen gemäß `SNVT_obj_status`. Die Bedeutung der Felder der `SNVT_obj_status` sind erklärt in [1].

Statusbit	**Wert**	**Bedeutung**
`disabled`	0	das Objekt ist gesperrt und kann keine Aktionen mehr ausführen
	1	das Objekt ist freigegeben und kann Aktionen ausführen

invalid_id	0 1	der empfangene Objekt Code wird unterstützt ein nicht unterstützter Objekt Code wurde empfangen
invalid_request	0 1	der empfangene Dienst Code wird unterstützt ein nicht unterstützter Dienst Code wurde empfangen
report_mask	1	die vom Objekt unterstützten Statusbits werden gemeldet

Tab. 3

When Transmitted

Statusdaten können übertragen werden, wenn ein Object Request via `nviRequest` empfangen wurde. Es ist kein Object Status Max Send Timer definiert.

Update Rules

Die Anwendung 16911210 aktualisiert den Status unmittelbar nach Empfang und Ausführung eines Dienstes via `nviRequest`. Diese Aktualisierung erfolgt in derselben Critical Section wie die Abarbeitung des angeforderten Dienstes.

Default Service Type

Der Default Service Type ist acknowledged.

Beispiel

Um den aktuellen Zustand des ersten Switch Objektes abzufragen, aktualisiere `nviRequest` mit `00 01 02`. Beim folgenden Polling der NV `nvoStatus` erhälts du beispielsweise den Wert `00 01 20 00 00 00` (`disabled`).

2.1.3 Eingang Zeit und Datum

```
network input SNVT_time_stamp nviTimestamp;
```

Diese NV dient zur Synchronisierung der geräteinternen Zeit mit einer externen Systemzeit.

Valid Range

Gültige Werte sind alle Werte in den definierten Grenzen gemäß `SNVT_time_stamp` und genügen den üblichen Werten, die bei der Zeit- und Datumsdarstellung genutzt werden. Die Bedeutung der Felder der `SNVT_time_stamp` sind erklärt in [1].
Letztes gültiges Datum für diese Anwendung ist der 31.12.2099.

When Transmitted

Erst nach der ersten Datenübertragung ist die geräteinterne Zeit definiert; bis dahin können keine uhrzeit- bzw. datumsabhängigen Aktionen ausgeführt werden, wie z.B. Absenden von Szenen durch das Scheduler Objekt. Alle weiteren Übertragungen synchronisieren die geräteinterne Zeit.
Ein Receiver Heartbeat ist parametrierbar.

Default Service Type

Der Default Service Type ist acknowledged.

Hinweis

Diese NV muss mit dem Ausgang einer Uhr gebunden sein, falls das Scheduler Objekt genutzt werden soll.

Beispiel

Zur Synchronsierung auf das Datum 24.12.1998 und die Uhrzeit 14.16 Uhr und 18 Sekunden setze die NV `nviTimestamp` auf `07 CE 0C 18 0E 10 12`.

2.1.4 Alarm-Ausgang

```
network output SNVT_alarm nvoAlarm;
```

Diese NV dient der Anzeige von Alarmen, die einige der Objekte der Anwendung 16911210 auslösen können. Sollte ein Objekt einen zuvor angezeigten Alarm zurücksetzen, wird auch dies angezeigt.

Valid Range

Folgende Tabelle listet auf, welche Alarme über `nvoAlarm` gemeldet werden können. In der Spalte *Objekt* sind die möglichen Alarme pro Objekttyp genannt. In der jeweils selben Zeile einer Alarmart steht in der Spalte *kommender Alarm* bzw. *gehender Alarm* die NV, auf die sich ein Alarm bezieht und der Alarmtyp.
Objekt, Alarmart und referenzierte NV-Index können über `nvoAlarm.object_id`, `nvoAlarm.alarm_type` und `nvoAlarm.index_to_SNVT` festgestellt werden. Siehe auch [1].

Objekt (Objekt Index)	**Art des Alarms**	**Beschreibung / Wert**	
		kommender Alarm	**gehender Alarm**
Node (0)	receive timeout (s. 2.2.1)	`nviTimestamp` `AL_ALM_CONDITION`	`nviTimestamp` `AL_NO_CONDITION`
Switch 1..8 (1...8)	*diesen Objekten sind keine Alarme zugeordnet*		
Scene Panel 1..4 (9..12)	*diesen Objekten sind keine Alarme zugeordnet*		
Controller 1/2 (13/14)	Grenzwertüber-schreitung	`nviWindspeed` `AL_HIGH_LMT_ALM_1`	`nviWindspeed` `AL_HIGH_LMT_CLR_1`
	quittierbarer Alarm	`nviAlarm` `AL_HIGH_LMT_ALM_1`	`nviAlarm` `AL_HIGH_LMT_CLR_1`
	nicht quittierbarer Alarm 1	`nviUnackAlarm0` `AL_ALM_CONDITION`	`nviUnackAlarm0` `AL_HIGH_LMT_CLR_1`
	nicht quittierbarer Alarm 2	`nviUnackAlarm1` `AL_HIGH_LMT_ALM_1`	`nviUnackAlarm1` `AL_HIGH_LMT_CLR_1`
	receive timeout nviWindspeed	`nviWindspeed` `AL_ALM_CONDITION`	`nviWindspeed` `AL_NO_CONDITION`
	receive timeout nviAlarm	`nviAlarm` `AL_ALM_CONDITION`	`nviAlarm` `AL_NO_CONDITION`
	receive timeout nviUnackAlarm0	`nviUnackAlarm0` `AL_ALM_CONDITION`	`nviUnackAlarm0` `AL_NO_CONDITION`
	receive timeout nviUnackAlarm1	`nviUnackAlarm1` `AL_ALM_CONDITION`	`nviUnackAlarm1` `AL_NO_CONDITION`
Scheduler (15)	*diesem Objekt sind keine Alarme zugeordnet*		

Tabelle 4 (Alarme)

Legende:

Mögliche Werte für `nvoAlarm.alarm_type`:

`AL_NO_CONDITION` = 0
`AL_ALM_CONDITION` = 1
`AL_HIGH_LMT_ALM_1` = 11
`AL_HIGH_LMT_CLR_1` = 7

Mögliche Werte für `nvoAlarm.index_to_SNVT`:

`nviTimestamp` = 2
`nviWindspeed` = 49 (1. Controller) / 50 (2. Controller)
`nviAlarm` = 45 (1. Controller) / 46 (2. Controller)
`nviUnackAlarm0` = 41 (1. Controller) / 42 (2. Controller)
`nviUnackAlarm1` = 43 (1. Controller) / 44 (2. Controller)

Default Service Type

Der Default Service Type ist acknowledged.

2.1.5 Geräteparametrierung

```
network input UNVT_cfg nviConfig;

typedef struct {
    scene_config_t  request;
    tObjId          objectIndex;
    tConfigType     configType;
    union {
        tNodeHeartbeats     nodeHeartbeats;
        tRtbSchedulerItem   schedulerItem;
        tControllerConfig   ctrl;
        tScenePanelCfg      scpCfg;
        tIoDisplay          display;
        tSwitchConfig       switchCfg;
        unsigned short      raw[28];
    } data;
} UNVT_cfg;
```

Diese Netzwerkvariable dient zur Parametrierung der kompletten Funktionalität aller Objekte. Diese NV ist insgesamt 31 Bytes lang.

Valid Range

Element	**Wertebereich / Bedeutung**
`.request`	`SCF_SAVE` (0) Parameter überschreiben mit neuen Daten `SCF_CLEAR` (1) Bedeutung siehe folgende Abschnitte `SCF_REPORT` (2) Parameterdaten darstellen (siehe folgendes Beispiel) *weitere Dienste werden nicht unterstützt*
`.objectIndex`	Index des Objektes, auf den `.request` angewendet werden soll s. Tab. 2 (Objekt-Kennungen)
`.configType`	Auswahl Parameterart: 0 Scheduler Objekt (relevantes `union`-Element: `schedulerItem`) 1 Controller Objekt (relevantes `union`-Element: `ctrl`) 2 Scene Panel Objekt (relevantes `union`-Element: scpCfg) 3 Anzeigeart (relevantes `union`-Element: `display`) 4 Node Objekt (relevantes `union`-Element: `nodeHeartbeats`) 4 Switch Objekt (relevantes `union`-Element: `switchCfg`) 71 Abfrage Applikationsversion

Default Value

Nach Reset hat die NV `nviConfig` die Werte:

`nviConfig.request` = INVALID (FF_{hex})
`nviConfig.objectIndex` = INVALID (FF_{hex})
`nviConfig.configType` = INVALID (FF_{hex})
alle anderen Felder = 0

Default Service Type

Der Default Service Type ist acknowledged.

2.2 Parametrierung

2.2.1 Send / Receiver Heartbeats

```
typedef struct {
        SNVT_time_sec snd;    // Send Heartbeat nvoAlarm
        SNVT_time_sec rcv;    // Receiver Heartbeat nviTimestamp
} tNodeHeartbeats;
```

Der Send Heartbeat legt den maximalen zeitlichen Abstand zwischen zwei aufeinanderfolgenden zyklischen Übertragungen der `nvoAlarm` fest.

Der Receiver Heartbeat legt fest, welche Zeit zwischen zwei aufeinanderfolgenden Übertragungen der `nviTimestamp` vergehen darf. Nach Ablauf dieser Zeit wird via `nvoAlarm` der Receive Timeout Alarm ausgelöst (s. Tabelle 4).

Valid Range

Send Heartbeat:	{0, 10, 20, ..., 65530} = {0, 1, 2, 6553} Sekunden 65535 = INVALID (Sendewiederholung ausgeschaltet)
Receiver Heartbeat:	{0, 10, 20, ..., 65530} = {0, 1, 2, 6553} Sekunden 65535 = INVALID (Empfangsüberwachung ausgeschaltet)

Beispiel

Send Heartbeat:	4 Minuten (09 60_{hex})
Receiver Heartbeat:	aus (FF FF_{hex})

Setze `nviConfig` auf 00 00 04 09 60 FF FF (alle weiteren Bytes beliebig)

2.3 Version/Revision der Applikation

Folgendes muss durchgeführt werden, um die Version bzw. Revision der Applikation zu ermitteln:

1. Applikation in ein Gerät LONMPM 8DI/DO laden
2. Gerät ONLINE setzen
3. `nviConfig` setzen auf `02 00 47` (alle weiteren Bytes beliebig)
4. `nviConfig` auslesen, wobei `nviConfig` enthalten muss:
   `02 00 47 31 36 39 31 31` **`32 30 41`** (alle weiteren Bytes beliebig), wobei 32 30 41 die ASCII-Codes der Zeichen 2, 0 und A somit und stehen für die Version/Revision 2.0A; die vollständige Applikationsbezeichnung lautet 1691120A.

Alternativ kann die Applikationsversion über die Configuration Properties abgefragt werden: `SCPTdevMajVer`/`SCPTdevMinVer` = 14_{hex} / $0A_{hex}$.

3 Switch Object

In der Anwendung 16911210 sind 8 Switch Objekte implementiert. Jedes dieser Objekte repräsentiert einen digitalen Eingang und liefert deren Werte in Abhängigkeit von der logischen Verknüpfung mit einer Feedback-NV.

3.1 Schnittstellenbeschreibung

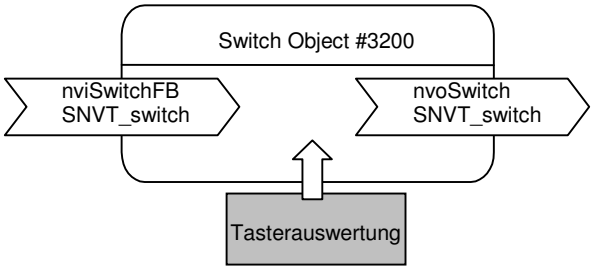

Abbildung 4 (Sensor Objekt)

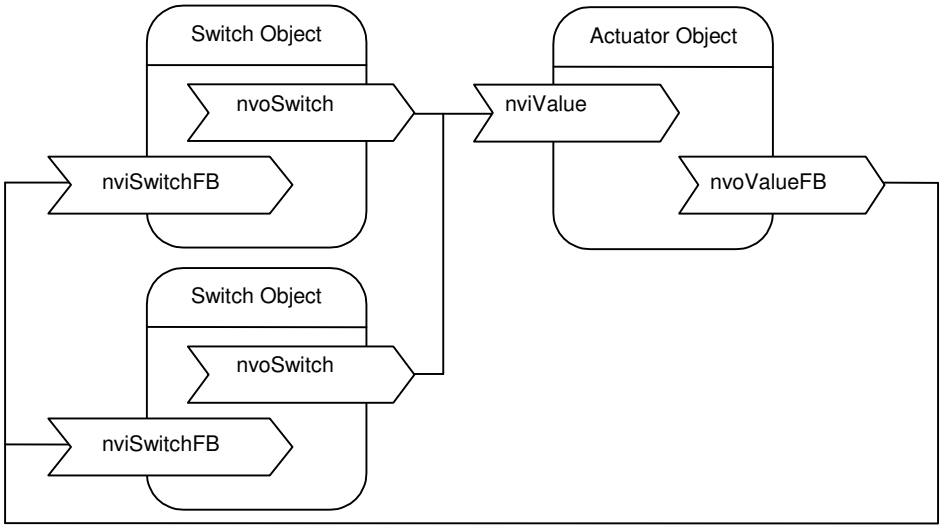

Abbildung 5 (Beispiel-Binding des Switch Objektes)

Abbildung 5 zeigt ein Binding zwischen Switch Objekten der Applikation 16911210 und einem Actuator Objekt einer anderen Anwendung. Diese Art des Bindings kann verwendet werden, um einen Lichtaktor in Wechselschaltung zu betreiben.

3.2 Digitale Eingänge

Switch Objekt	Objektindex	Klemmenpaar
1	1	digitaler Eing. 1: X9.1 – X9.2
2	2	digitaler Eing. 2: X9.3 – X9.4
3	3	digitaler Eing. 3: X9.5 – X9.6
4	4	digitaler Eing. 4: X9.7 – X9.8
5	5	digitaler Eing. 5: X9.9 – X9.10
6	6	digitaler Eing. 6: X9.11 - X9.12
7	7	digitaler Eing. 7: X9.13 - X9.14
8	8	digitaler Eing. 8: X9.15 - X9.16

Tabelle 5 (Zuordnung Switch Objekt und Anschlussklemme)

3.2.1 Value Output

```
network output SNVT_switch nvoSwitch;
```

Diese NV liefert die Daten entsprechend den Zuständen der digital Eingänge des Gerätes LONMPM 8 DI/DO und dem Feedback-Eingang `nviSwitchFB`. Die Art der logischen Verknüpfung kann parametriert werden.

Valid Range

Wertebereich und Bedeutung der `SNVT_switch` sind erklärt in *[1]*.

Default Value

Nach dem Reset oder dem Anlegen der Betriebsspannung werden alle Switch Objekte in Abhängigkeit von digitalem Eingang und des Feedback-Eingangs neu berechnet. Der ermittelte Wert wird dann über das Netzwerk gesendet.

When Transmitted

Daten können übertragen werden, wenn sich der Zustand des Tasters ändert. Änderungen können sein:

Taste gedrückt/losgelassen
Änderung des Feedback-Einganges `nviSwitchFB`

Ein Timer zur Sendewiederholung kann parametriert werden.

Update Rules

Die Anwendung 1691120a aktualisiert den Switch-Objektausgang unmittelbar nach Änderung des Verknüpfungsergebnisses aus digitalen Eingänge und Feedback-Eingang. Eine Sendewiederholrate ist parametrierbar.

Default Service Type

Der Default Service Type ist acknowledged.

3.2.2 Feedback Input

```
network input SNVT_switch nviSwitchFB;
```

Diese NV dient der Rückführung eines Aktorzustandes auf das Switch Objekt oder der Verkettung mit anderen Switch Objekten. Abhängig vom Feedback-Wert und der Art der logischen Verknüpfung mit digitalem Eingang wird der Wert der Ausgangsvariable neu berechnet.

Valid Range

Wertebereich und Bedeutung der `SNVT_switch` sind erklärt in *[1]*.

3.3 Parametrierung

```
typedef struct {
       unsigned short type;      // Verknüpfungsart feedback nv - io
       SNVT_time_sec  sndHrtbt;
} tSwitchCfg;
```

Jedem Switch Objekt ist ein gemäß `tSwitchCfg` strukturierter Parametersatz zugeordnet. Er bestimmt das grundlegende Verhalten des Objektes.
Mit `type` wird die Art der logischen Verknüpfung zwischen der Feedback-NV und dem digitalen Eingang festgelegt. Durch `sndHrtbt` wird die maximale Zeit zwischen zwei aufeinanderfolgenden Sendungen der `nvoSwitch` wird bestimmt.

3.3.1 Verknüpfungsart zwischen Feedback-NV und digitalem Eingang

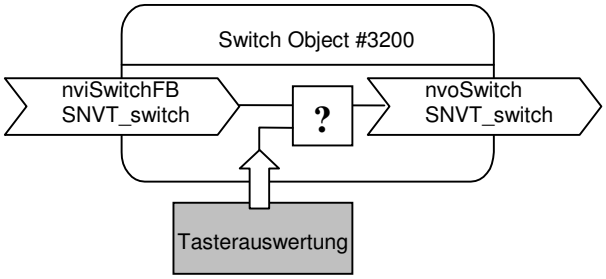

Abbildung 6

Verknüpfungsart/Funktion	Code	Bedeutung
SWITCH	0x00	zum Anschluß eines Schalters
PUSHBUTTON	0x01	zum Anschluß eines Tasters
SWITCH_DAISYCHAIN_AND	0x02	logisches UND-Glied
SWITCH_DAISYCHAIN_OR	0x03	logisches ODER-Glied
SWITCH_DAISYCHAIN_XOR	0x04	logisches EXKLUSIV-ODER-Glied
SWITCH_DAISYCHAIN_NAND	0x05	logisches negiertes UND-Glied
SWITCH_DAISYCHAIN_NOR	0x06	logisches negiertes ODER-Glied
SWITCH_DAISYCHAIN_NXOR	0x07	logisches negiertes EXKLUSIV-ODER-Glied

Tabelle 6: Verknüpfungsarten/Funktionen des Switch Objekts

SWITCH (0x00)

Dieser Modus ist vorgesehen für den Anschluß eines Schalters. Der Ausgang verhält sich wie der eines Wechselschalters. Der Wert der Ausgangsvariablen `nvoSwitch` wird bei jeder Änderung des Schaltzustandes neu überprüft und gegebenenfalls geändert. Eine Änderung am Feedback-Eingang führt nicht zu einem erneuten Versenden des Ausgangswertes. Intern wird in diesem Fall die Ausgangsvariable jedoch verändert, so daß das Polling der Ausgangsvariable bereits einen veränderten aber nicht gesendeten Wert anzeigen kann.

Ist in diesem Modus der Heartbeat aktiviert, so ist es erforderlich, daß die Feedbackvariable auf den entsprechenden Feedbackausgang des angesteuerten Aktors gebunden wird.

PUSHBUTTON (0x01)

Dieser Modus ist vorgesehen für den Anschluß eines Tasters. Der Ausgang verhält sich analog zum SWITCH (0x00). Der Wert der Ausgangsvariablen `nvoSwitch` wird jedoch bei jedem Druck des Tasters neu überprüft und gegebenenfalls geändert. Ein Loslassen des Tasters bewirkt keine Änderung.

SWITCH_DAISYCHAIN_AND (0x02)

Dieser Modus gibt den Zustand ON aus, wenn sowohl angeschlossener Taster / Schalter / Kontakt geschlossen ist, als auch der Feedbackeingang `nviSwitchFB` den Zustand ON empfangen hat. In allen anderen Fällen wird OFF ausgegeben.

Schalter- Tasterzustand	Wert von: `nviSwitchFB`	Wert von: `nvoSwitch`
Offen	OFF	OFF
Offen	ON	OFF
Geschlossen	OFF	OFF
Geschlossen	ON	ON

Tabelle 7: Wahrheitstabelle logische UND- Funktion

SWITCH_DAISYCHAIN_OR (0x03)

Dieser Modus gibt den Zustand ON aus, wenn entweder der Taster / Schalter geschlossen ist, oder/und der Feedbackeingang `nviSwitchFB` den Zustand ON empfangen hat. Ist weder der Taster / Schalter geschlossen noch die Feedbackvariable ON, wird OFF ausgegeben.

Schalter- Tasterzustand	Wert von: `nviSwitchFB`	Wert von: `nvoSwitch`
Offen	OFF	OFF
Offen	ON	ON
Geschlossen	OFF	ON
Geschlossen	ON	ON

Tabelle 8: Wahrheitstabelle logische ODER- Funktion

SWITCH_DAISYCHAIN_XOR (0x04)

Dieser Modus gibt den Zustand ON aus, wenn entweder der Taster / Schalter geschlossen ist und der Feedbackeingang `nviSwitchFB` den Zustand OFF empfangen hat, oder der Taster / Schalter offen ist und der Feedbackeingang `nviSwitchFB` den Zustand ON empfangen hat. In allen anderen Fällen wird OFF ausgegeben.

Schalter- Tasterzustand	Wert von: `nviSwitchFB`	Wert von: `nvoSwitch`
Offen	OFF	OFF
Offen	ON	ON
Geschlossen	OFF	ON
Geschlossen	ON	OFF

Tabelle 9: Wahrheitstabelle der logischen EXKLUSIV-ODER-Funktion

SWITCH_DAISYCHAIN_NAND(0x05)

Dieser Modus verhält sich ähnlich wie der Modus `SWITCH_DAISYCHAIN_AND(0x02)`. Die Ausgangsvariable nimmt jedoch den gegenteiligen Zustand an.
Der Modus gibt den Zustand OFF aus, wenn sowohl der Taster / Schalter geschlossen ist, als auch der Feedbackeingang `nviSwitchFB` den Zustand ON empfangen hat. In allen anderen Fällen wird ON ausgegeben.

Schalter- Tasterzustand	Wert von: `nviSwitchFB`	Wert von: `nvoSwitch`
Offen	OFF	ON
Offen	ON	ON
Geschlossen	OFF	ON
Geschlossen	ON	OFF

Tabelle 10: Wahrheitstabelle der logischen NEGIERT-UND-Funktion

SWITCH_DAISYCHAIN_NOR (0x06)

Dieser Modus verhält sich ähnlich wie der Modus `SWITCH_DAISYCHAIN_OR(0x03)`. Die Ausgangsvariable nimmt jedoch den gegenteiligen Zustand an.
Der Modus gibt den Zustand OFF aus, wenn entweder der Taster / Schalter geschlossen ist, oder/und der Feedbackeingang `nviSwitchFB` den Zustand ON empfangen hat. Ist weder der Taster / Schalter geschlossen, noch die Feedbackvariable ON, wird ON ausgegeben.

Schalter- Tasterzustand	Wert von: nviSwitchFB	Wert von: nvoSwitch
Offen	OFF	ON
Offen	ON	OFF
Geschlossen	OFF	OFF
Geschlossen	ON	OFF

Tabelle 11: Wahrheitstabelle der NEGIERT-ODER-Funktion

SWITCH_DAISYCHAIN_NXOR (0x07)

Dieser Modus verhält sich ähnlich wie der Modus `SWITCH_DAISYCHAIN_XOR(0x04)`. Die Ausgangsvariable nimmt jedoch den gegenteiligen Zustand an.
Der Modus gibt den Zustand OFF aus, wenn entweder der Taster / Schalter geschlossen ist und der Feedbackeingang `nviSwitchFB` den Zustand OFF empfangen hat, oder der Taster / Schalter offen ist und der Feedbackeingang `nviSwitchFB` den Zustand ON empfangen hat. In allen anderen Fällen wird ON ausgegeben.

Schalter- Tasterzustand	Wert von: nviSwitchFB	Wert von: nvoSwitch
Offen	OFF	ON
Offen	ON	OFF
Geschlossen	OFF	OFF
Geschlossen	ON	ON

Tabelle 12: Wahrheitstabelle der NEGIERT-EXKLUSIV-ODER-Funktion

Beispiel

Um im Switch 6 (06) eine Sendewiederholrate von 10 Minuten (17 70_{hex}) und die Verknüpfungsart SWITCH_DAISYCHAIN_OR (03) einzustellen, aktualisiere `nviConfig` mit 00 06 05 03 17 70 (alle weiteren Bytes beliebig).

3.3.2 Sendewiederholrate (Send Heartbeat)

Durch `sndHrtbt` wird die maximale Zeit zwischen zwei aufeinanderfolgenden Sendungen der `nvoSwitch` wird bestimmt.

Valid Range

{0, 10, 20, ..., 65530} digits entsprechen {0, 1, 2, ..., 6553} Sekunden.
65535 entspricht INVALID (Sendewiederholung Aus)

Beispiel

Um im Switch 4 (04) eine Sendewiederholrate von 10 Minuten (17 70_{hex}) und die Verknüpfungsart SWITCH_DAISYCHAIN_OR (03) einzustellen, aktualisiere `nviConfig` mit 00 04 05 03 17 70 (alle weiteren Bytes beliebig).

4 Scene Panel Objekt

In der Anwendung 16911210 sind 4 Scene Panel Objekte implementiert. Sie werden durch `SNVT_setting`- oder eine `SNVT_switch`-NV´s angesteuert und senden `SNVT_scene` oder `SNVT_setting`-NV´s aus. Szenen-Feedbacks werden über über LED´s bzw. digitale Ausgänge angezeigt.

4.1 Funktionsweise

Bestimmte Inhalte der `SNVT_switch` / `SNVT_setting`-Eingänge werden interpretiert als Zustände „Taste gedrückt“ oder „Taste losgelassen“.

Zustand	**`nviSetting.function`**	**`nviSwitch.state`**
„Taste gedrückt“	`SET_UP` / `SET_DOWN`	≠ 0
„Taste losgelassen“	`SET_OFF`	= 0

Tabelle 13

Die Elemente `nviSetting.setting`, `nviSetting.rotation` und `nviSwitch.value` können beliebige Inhalte annehmen. Sie werden nicht ausgewertet, um zu ermitteln, ob „Taste gedrückt“ oder „Taste losgelassen“.

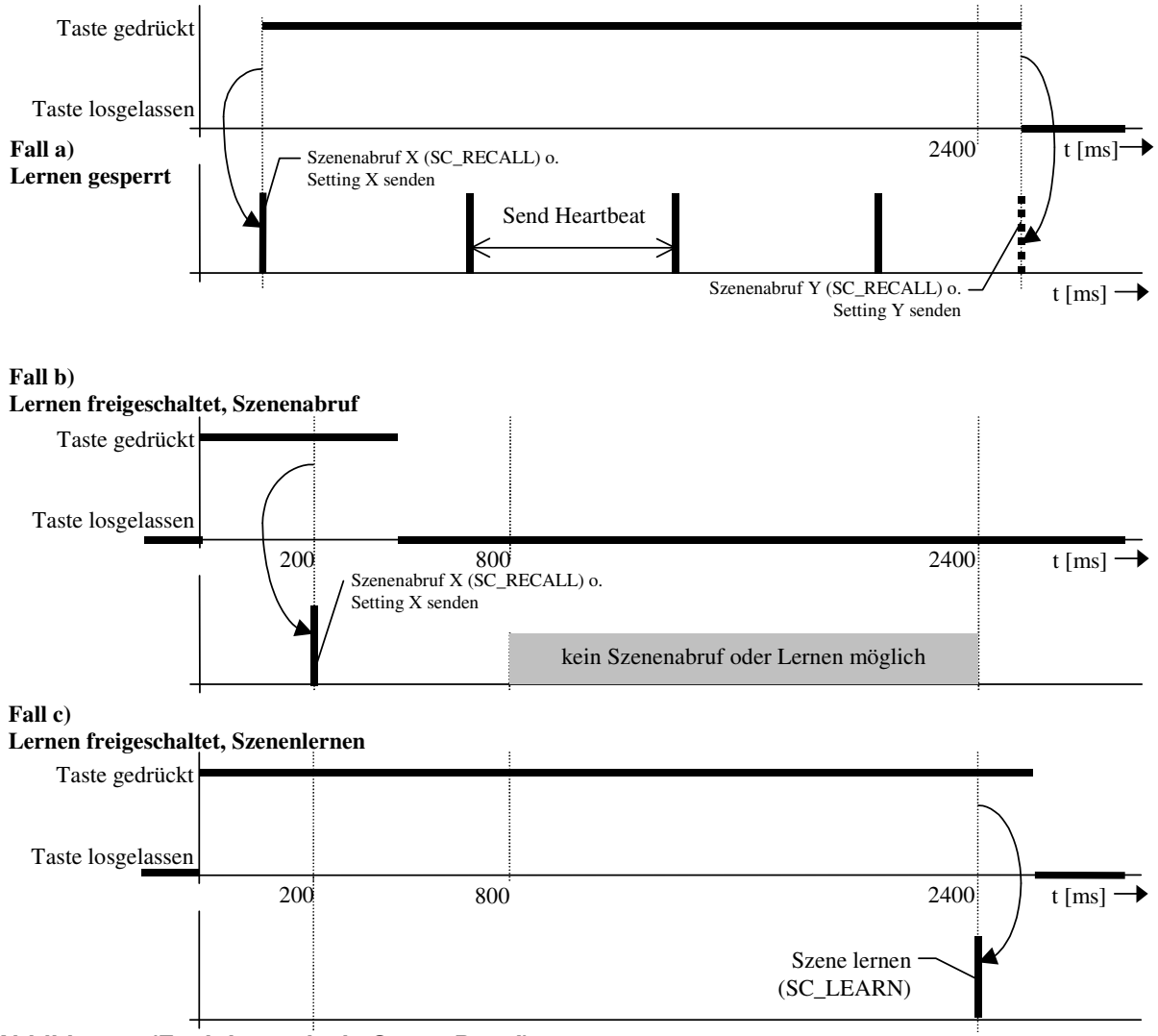

Abbildung 7 (Funktionsprinzip Scene Panel)

Jedem Scene Panel sind eine LED und ein digitaler Ausgang zugeordnet (siehe Tabelle 14). Stimmt der Szenen-Feedback mit der zu sendende Szene überein, leuchtet die LED bzw. es wird der digitale Ausgang aktiv. Wurde Szene Lernen gesendet, wird dies angezeigt durch mehrmaliges Blinken der LED bzw. mehrmaligem Umschalten des digitalen Ausgangs.

4.2 Digitale Ausgänge

Szenen-Feedbacks werden über über LED´s bzw. digitale Ausgänge angezeigt.

Scene Panel	Objektindex	LED	Klemmenpaar
1	5	K1	digitaler Ausg. 1: X1.1 - X1.2
2	6	K2	digitaler Ausg. 2: X1.3 - X1.4
3	7	K3	digitaler Ausg. 3: X4.1 - X4.2
4	8	K4	digitaler Ausg. 4: X4.3 - X4.4

Tabelle 14

4.3 Schnittstellenbeschreibung

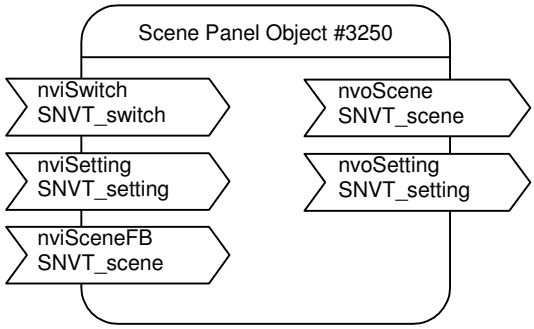

Abbildung 8 (Scene Panel Objekt)

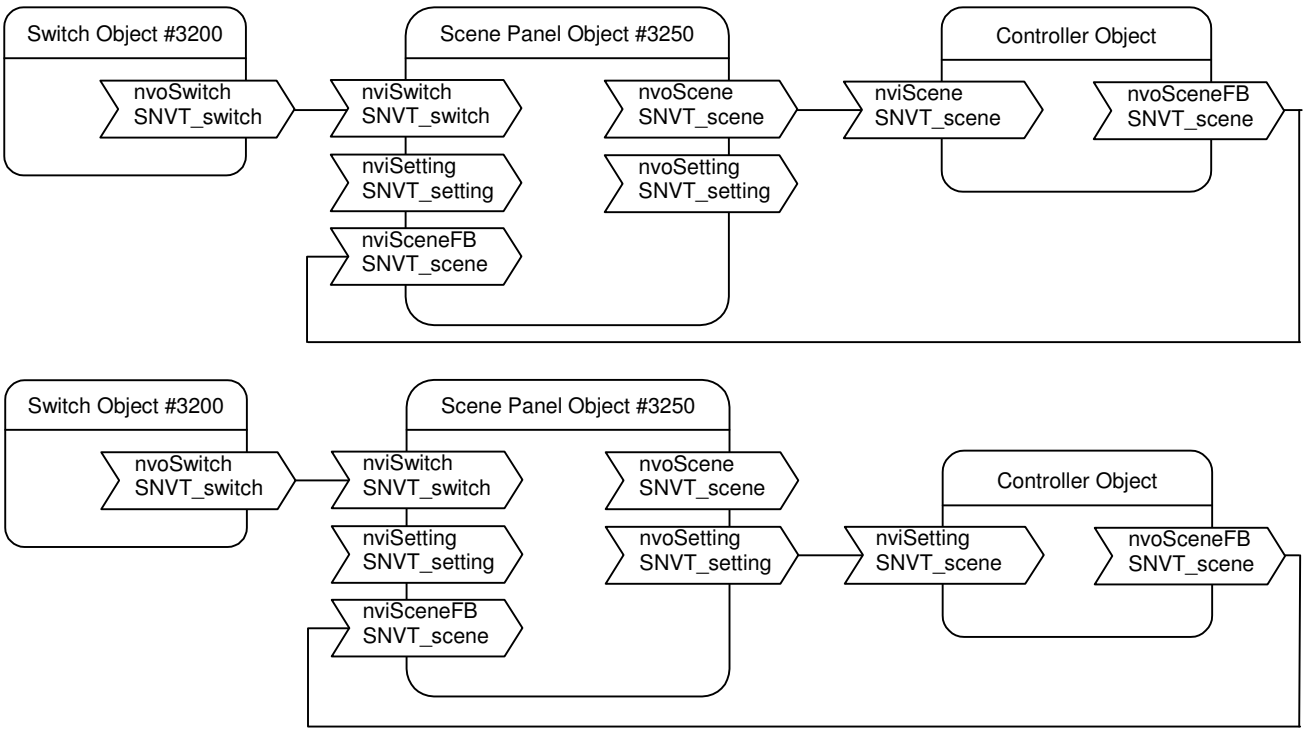

Abbildung 9 (Beispiel-Bindings des Scene Panel Objektes)

Das obere Binding der Abbildung 9 zeigt, wie mit einem Switch und einem Scene Panel Objekt der Applikation 16911210 ein Controller einer anderen Anwendung angesteuert wird, so dass z.B. mit einem Standardtaster Szenen ausgelöst werden könnten. Durch das Binding, wie es im unteren Abbildungsteil dargestellt ist, könnten z.B. mit einem Switch Objekt bzw. einem Standardtaster Settings an einen Controller gesendet werden.

4.3.1 Scene Output

```
network output SNVT_scene nvoScene;
```

Diese NV ist der Ausgang des Scene Panels, über den Szenenlernen gesendet werden kann oder Abruf einer Szene X bzw. Y.

Valid Range

Zu Bedeutung und zum Wertebereich der `SNVT_scene` siehe *[1]*. Absenden des Szenenlernens kann per Parametrierung unterdrückt werden. Für den Zustand „Taste gedrückt“ an `nviSwitch / nviSetting` kann eine parametrierbare Szene X mit Send Heartbeat gesendet werden, für den Zustand „Taste losgelassen“ einmalig eine andere parametrierbare Szene Y.

When Transmitted

Diese NV wird übertragen, unmittelbar nachdem die Applikation ihr Inhalt änderte. Zu den Änderungsbedingungen siehe Abbildung 7.

Update Rate

Eine Sendewiederholrate zum Senden der Szene X ist einstellbar.

4.3.2 Setting Output

```
network output SNVT_setting nvoSetting;
```

Diese NV ist ein optionaler Ausgang des Scene Panels, über den ein Setting X bzw. Y gesendet werden kann.

Valid Range

Zu Bedeutung und zum Wertebereich der SNVT_setting siehe *[1]*. Für den Zustand „Taste gedrückt“ an `nviSwitch / nviSetting` kann ein parametrierbarer Setting-Wert gesendet werden, für den Zustand „Taste losgelassen“ ein anderer parametrierbarer Setting-Wert.

When Transmitted

Diese NV wird übertragen, unmittelbar nachdem die Applikation ihr Inhalt änderte. Zu den Änderungsbedingungen siehe Abbildung 7.

Update Rate

Eine Sendewiederholrate zum Senden von Setting X ist einstellbar.

4.3.3 Scene Feedback Input

```
network input SNVT_scene nviSceneFB;
```

Diese NV dient der Rückführung des Wertes eines Scene Controllers o.ä. Diese NV muss dann mit dem Ausgang eines Empfängers der `nvoScene / nvoSetting` gebunden sein, wenn die Anzeige über LED´s oder digitale Ausgänge genutzt werden soll.

Valid Range

Zu Bedeutung und zum Wertebereich der `SNVT_scene` siehe *[1]*.

4.3.4 Setting Input

```
network input SNVT_setting nviSetting;
```

Diese NV dient der Anbindung eines 2-fach Jalousietasters an das Scene Panel Objekt. Somit könnten durch einen derartigen Taster nach kurzzeitigem Tastendruck Szenen ausgelöst werden. Durch langen Tastendruck kann Szenenlernen abgerufen werden.

Valid Range

`nviSetting.function`	**Interpretation**
`SET_OFF`	Taste losgelassen
`SET_DOWN`	Taste Tief gedrückt
`SET_UP`	Taste Hoch gedrückt

Weitere Werte der `nviSetting` werden nicht verarbeitet. Die beiden anderen Felder `nviSetting.setting` und `nviSetting.rotation` werden nicht verarbeitet und können beliebige Werte annehmen.

4.3.5 Switch Input

```
network input SNVT_switch nviSwitch;
```

Diese NV dient der Anbindung eines Switch Objekte an das Scene Panel Objekt. Somit könnten durch einen Taster, der durch ein Switch Objekt repräsentiert wird, nach kurzzeitigem Tastendruck Szenen ausgelöst werden. Durch langen Tastendruck kann Szenenlernen abgerufen werden.

Valid Range

nviSwitch.state	**Interpretation**
= 0	Taste losgelassen
≠ 0	Taste gedrückt

Das Feld `nviSwitch.value` wird nicht verarbeitet und kann beliebige Werte annehmen.

4.4 Parametrierung

```
typedef struct {
      boolean       bLearn;     // Lernen Ein/Aus
      SNVT_time_sec sndHrtbt;   // Send Heartbeat nvoScene / nvoSetting
      unsigned int  numX;       // Szene X (Taste gedrückt)
      unsigned int  numY;       // Szene Y (Taste losgelassen)
      SNVT_setting  settingX;   // Setting X (Taste gedrückt)
      SNVT_setting  settingY;   // Setting Y (Taste losgelassen)
} tSceneCfg;
```

Das Verhalten jedes Scene Panels wird durch eine gemäß `tSceneCfg` formatierte Struktur beschrieben.

Beispiel

Szene Panel 1 (09) soll parametriert werden wie folgt:

Lernen	Ein	01
Send Heartbeat	Aus	FF FF_{hex}
Szene X (nach Tastendruck)	4	04
Szene Y (nach Taste losgelassen)	nicht senden	00
Setting X (nach Tastendruck)	nicht senden	FF FF 7F FF_{hex}
Setting Y (nach Taste losgelassen)	100%/30°	05 C8 05 DC

Setze nviConfig auf 00 09 02 01 FF FF 04 00 FF FF 7F FF 05 C8 05 DC(alle weiteren Bytes beliebig).

4.4.1 Freigabe Szenenlernen

```
boolean bLearn;
```

Valid Range

0: Lernen gesperrt
1: Lernen freigeschaltet

4.4.2 Sendewiederholrate (Send Heartbeat)

```
SNVT_time_sec sndHrtbt;
```

Valid Range

{0, 10, 20, ..., 65530} digits entsprechen {0, 1, 2, 6553} Sekunden.
65535 entspricht INVALID (Sendewiederholung Aus).

4.4.3 Szenennummern

```
unsigned int numX;      // nach Tastendruck
unsigned int numY;      // nach Taste losgelassen
```

Nach Tastendruck kann über `nvoScene` die Szene X gesendet werden. Nachdem eine Taste losgelassen wurde kann eine Szene Y gesendet werden (siehe Abbildung 7).

Valid Range

`numX, numY` = {1,2,3,...,255}
0 entspr. INVALID (wird nicht gesendet)

4.4.4 Setting-Werte

```
SNVT_setting settingX;  // nach Tastendruck
SNVT_setting settingy;  // nach Taste losgelassen
```

Nach Tastendruck kann über `nvoSetting` der Setting-Wert X gesendet werden. Nachdem eine Taste losgelassen wurde, kann ein Setting-Wert Y gesendet werden (siehe Abb. 7).

Valid Range

Zum Wertebereich der `SNVT_setting` siehe *[1]*. Das Senden wird unterdrückt, wenn `settingX.function` bzw. `settingY.function` den Wert `SET_NUL` (FF_{hex}) haben.

5 Controller Objekt

In der Anwendung 16911210 sind 2 Controller Objekte implementiert. Sie verknüpfen und überwachen Eingangs-NV´s und erzeugen interne Alarmzustände. Diese Alarmzustände werden an LED´s und digitalen Ausgängen angezeigt und durch Ausgangs-NV´s übertragen.

5.1 Funktionsweise

Einige Applikationen der LON-MSE 2/4M230 besitzen Eingangs-NV´s für Helligkeit, Windgeschwindigkeit und -richtung; hingegen Eingänge für Regen- und Eismeldungen sindnicht vorgesehen. Um außenliegende Sonnenschutzbehänge aber bei Regen oder Vereisungsgefahr zu schützen, müssen der Steuerungen auf Regen- bzw. Eisalarme reagieren können.
Das Controller Objekt des LON-8DI/DO kann die empfangene Windgeschwindigkeit mit anderen empfangenen Alarmmeldungen bzw. -quittierungen verknüpfen. Das Verknüpfungergebnis – dargestellt als Geschwindigkeit – stellt nicht mehr die gemessene Windgeschwindigkeit dar, sondern u.U. einen höheren Wert. Dies kann dann in den Sonnenschutzsteuerungen zu einer Überschreitung der zulässigen Windgeschwindigkeit führen. Der Zustand des Windalarms tritt ein.
Voraussetzung hierfür ist, daß in der Sonnenschutzsteuerung als auch im LONMPM 8DI/DO identische Grenzwerte der Windgeschwindigkeit parametriert sind.

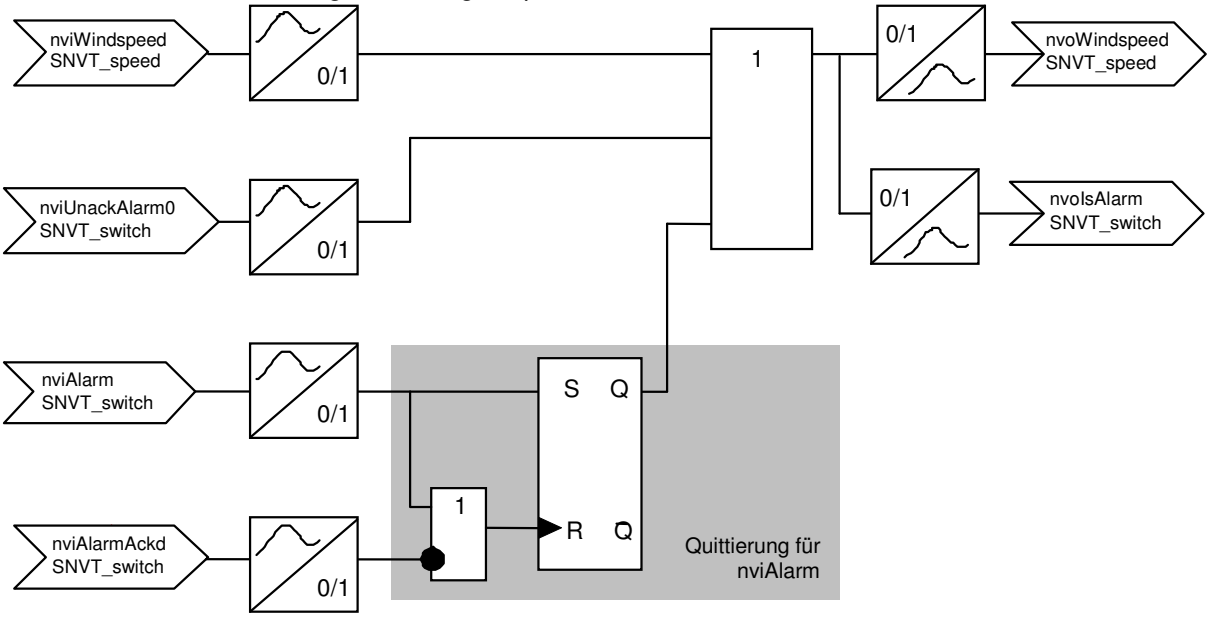

Abbildung 10 (Controller Funktionsprinzip)

Die den Eingangs-NV nachgeschalteten Module erkennen Grenzwertüberschreitung oder –unterschreitung. Hierfür können für die Windgeschwindigkeit oberer und unterer Grenzwert parametriert werden. Hingegen ist Alarm bzw. dessen Löschung/Quittierung erfüllt, wenn die Inhalte einer `SNVT_switch` ungleich 0 oder gleich 0 sind.
Das Controller Objekt generiert dann einen Alarm in Form einer überhöhten Windgeschwindigkeit bzw. eines bestimmten `SNVT_switch`-Inhaltes, wenn an einem der Eingänge ein Alarm ansteht oder wenn der Receiver Heartbeats von `nviWindspeed`, `nviAlarm`, `nviUnackAlarm0` oder `nviUnackAlarm1` abgelaufen ist.
Alarme des Controller Objektes werden auch über `nvoAlarm` des Node Objektes gemeldet (s. Tabelle 4).

5.2 Digitale Ausgänge

Jedem Controller Objekt sind je zwei LED´s bzw. digitale Ausgänge zugeordnet.

Controller	Objektindex	LED	Klemmenpaar
1	9	K5	digitaler Ausg. 5: X6.1 – X6.2
		K6	digitaler Ausg. 6: X6.3 – X6.4
2	10	K7	digitaler Ausg. 7: X7.1 – X7.2
		K8	digitaler Ausg. 8: X7.3 – X7.4

Tabelle 15

Die LED´s K5 und K7 bzw. digitalen Ausgänge 5 und 7 zeigen an,
a) ob an nviAlarm „Alarm aktiv“ gemeldet wurde oder
b) ob der interne Alarmzustand aktiv (EIN) oder inaktiv (AUS) ist.
K6 bzw. K8 zeigen, an ob ein Alarm quittiert wurde.

5.3 Schnittstellenbeschreibung

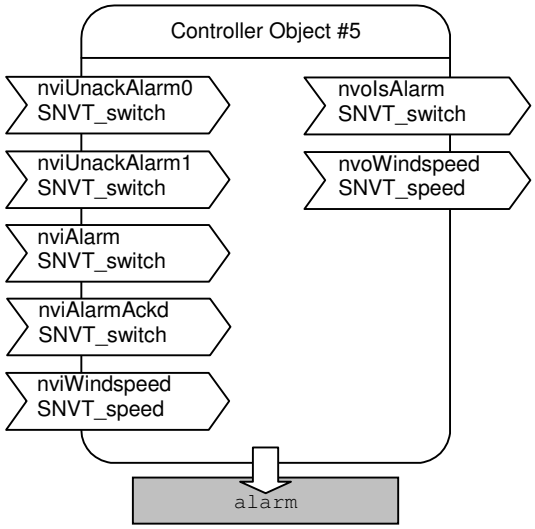

Abbildung 11 (Controller Objekt)

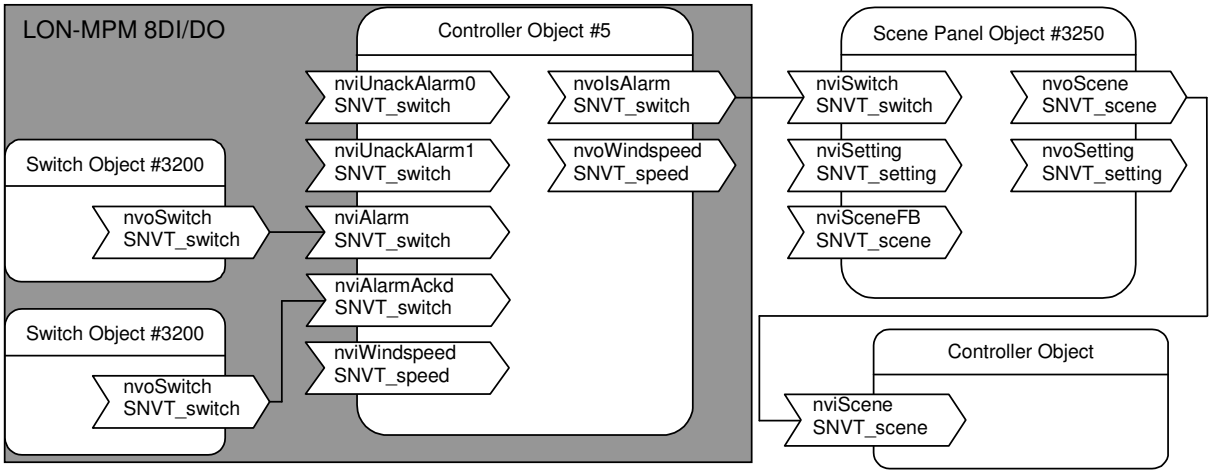

Abbildung 12 (Beispielbinding des Controllerobjektes)

5.3.1 Quittierbarer Alarm

```
network input SNVT_switch nviAlarm;
```

Diese Eingangs-NV beeinflusst den Alarmzustand des Controller Objektes. Der Alarmzustand wird angezeigt

- an LED K5 und digitalem Ausgang X6.1 – X6.2 bzw. LED K7 und X7.1 – X7.2 (s. Tabelle 15)
- über `nvoIsAlarm`
- über `nvoWindspeed`
- über `nvoAlarm` des Node Objekts (s. Tabelle 4).

Ein Alarm, der über `nviAlarm` ausgelöst wurde, kann m.H. der `nviAlarmAckd` quittiert werden.

Ein Receiver Heartbeat ist parametrierbar.

Valid Range

nviAlarm	**Zustand**
.state = 0 .value = 0	Alarm inaktiv
.state ≠ 0 .value nicht betrachtet	Alarm aktiv

Default Service Type

Der Default Service Type ist acknowledged.

5.3.2 Alarm-Quittung

```
network input SNVT_switch nviAlarmAckd;
```

Über diese Eingangs-NV kann der an nviAlarm empfangene Alarm quittiert werden. Nach erfolgter Quittierung kann erst dann erneut ein Alarm ausgelöst werden, wenn vorab an nviAlarm „Alarm inaktiv“ gemeldet wurde.

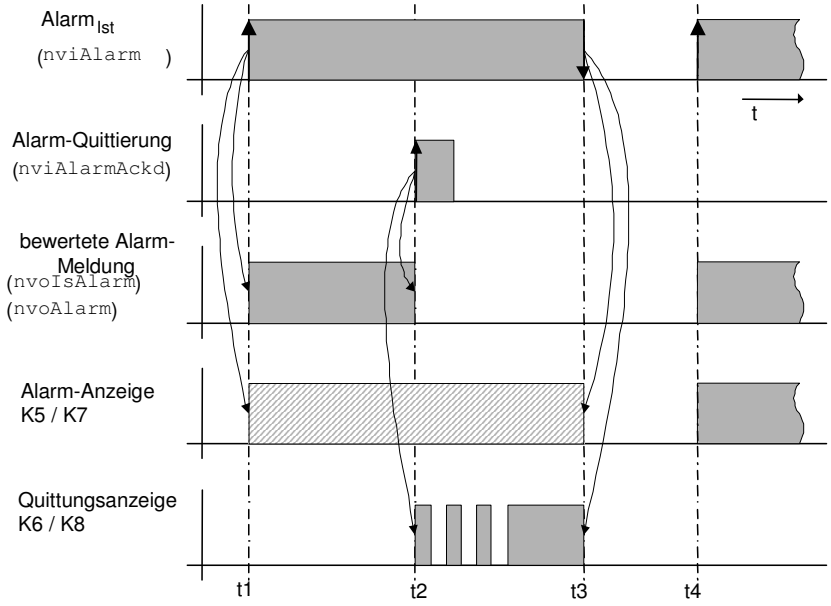

Abbildung 13

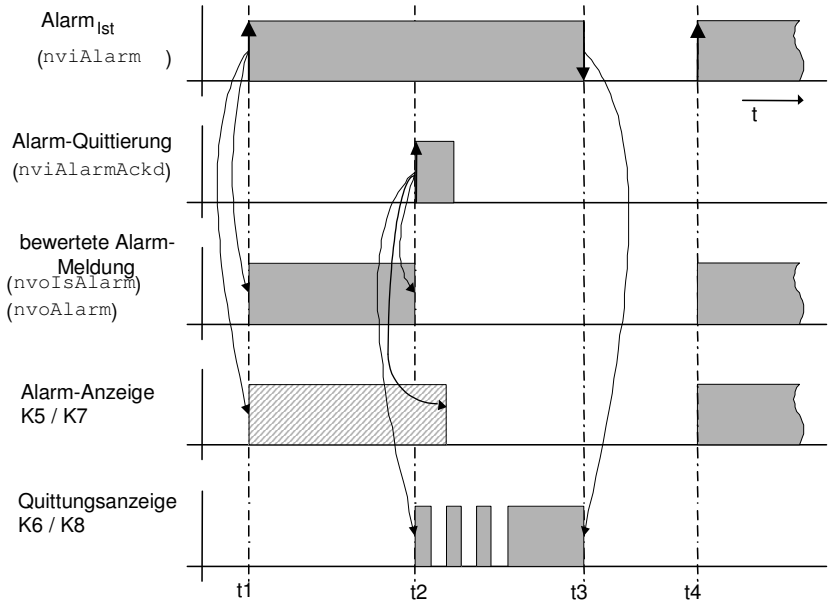

Abbildung 14

Die LED´s K5 und K7 bzw. digitalen Ausgänge 5 und 7 können anzeigen,

a) ob an `nviAlarm` „Alarm aktiv“ gemeldet wurde (s. Abbildung 13) oder
b) ob der interne Alarmzustand aktiv (EIN) oder inaktiv (AUS) ist (s. Abbildung 14).

Valid Range

nviAlarmAckd	**Bedeutung**
`.state` = 0	Quittierung nicht erfolgt
`.state` ≠ 0	Quittierung erfolgt

`nviAlarmAckd.value` wird nicht betrachtet.

Default Service Type

Der Default Service Type ist acknowledged.

5.3.3 Messwert Windgeschwindigkeit

```
network input nviWindspeed;
```

Diese Eingangs-NV beeinflusst den Alarmzustand des Controller Objektes (s. 5.1). Der Alarmzustand hängt von parametrierbaren Grenzwerten ab und wird angezeigt

- an LED K5 und digitalem Ausgang X6.1 – X6.2 bzw. LED K7 und X7.1 – X7.2 (s. Tabelle 15)
- über `nvoIsAlarm`
- über `nvoWindspeed`
- über `nvoAlarm` des Node Objekts (s. Tabelle 4).

Ein Alarm, der über `nviWindspeed` ausgelöst wurde, kann nicht quittiert werden.

Ein Receiver Heartbeat ist parametrierbar.

Valid Range

{0, 1, 2, ..., 65534} digits entsprechen {0, 0.1, 0.2, ..., 6553.4} m/s. 65535 entspricht INVALID.

Default Service Type

Der Default Service Type ist acknowledged.

5.3.4 Erster nicht quittierbarer Alarm

```
network input SNVT_switch nviUnackAlarm0;
```

Diese Eingangs-NV beeinflusst den Alarmzustand des Controller Objektes (s. 5.1). Der Alarmzustand wird angezeigt

- an LED K5 und digitalem Ausgang X6.1 – X6.2 bzw. LED K7 und X7.1 – X7.2 (s. Tabelle 15)
- über `nvoIsAlarm`
- über `nvoWindspeed`
- über `nvoAlarm` des Node Objekts (s. Tabelle 4).

Ein Alarm, der über `nviUnackAlarm0` ausgelöst wurde, kann nicht quittiert werden.

Ein Receiver Heartbeat ist parametrierbar.

Valid Range

nviUnackAlarm0	**Zustand**
`.state` = 0	Alarm inaktiv
`.state` ≠ 0	Alarm aktiv

`nviUnackAlarm0.value` wird nicht betrachtet.

Default Service Type

Der Default Service Type ist acknowledged.

5.3.5 Zweiter nicht quittierbarer Alarm

```
network input SNVT_switch nviUnackAlarm1;
```

Diese Eingangs-NV beeinflusst den Alarmzustand des Controller Objektes (s. 5.1). Der Alarmzustand wird angezeigt

- an LED K5 und digitalem Ausgang X6.1 – X6.2 bzw. LED K7 und X7.1 – X7.2 (s. Tabelle 15)
- über `nvoIsAlarm`
- über `nvoWindspeed`
- über `nvoAlarm` des Node Objekts (s. Tabelle 4).

Ein Alarm, der über `nviUnackAlarm1` ausgelöst wurde, kann nicht quittiert werden.

Ein Receiver Heartbeat ist parametrierbar.

Valid Range

nviUnackAlarm1	**Zustand**
`.state` = 0	Alarm inaktiv
`.state` ≠ 0	Alarm aktiv

`nviUnackAlarm1.value` wird nicht betrachtet.

Default Service Type

Der Default Service Type ist acknowledged.

5.3.6 Alarmanzeige

```
network output SNVT_switch nvoIsAlarm;
```

Diese Ausgangs-NV überträgt den Alarmzustand des Controller Objektes. Der Alarmzustand kann ermittelt werden, wie in 5.1 erläutert. Zusätzlich können abgelaufene Receiver Heartbeats der `nviWindspeed`, `nviAlarm`, `nviUnackAlarm0` oder `nviUnackAlarm1` zu Alarmen führen.

Valid Range

nvoIsAlarm	**Zustand**
`.state` = 0 `.value` = 0	Alarm inaktiv
`.state` = 1 `.value` = 200 (100%)	Alarm aktiv

When Transmitted

Diese NV wird übertragen, sobald deren Inhalt durch die Applikation geändert wurde, oder wenn der parametrierbare Send Heartbeat dieser NV abgelaufen ist.

Default Service Type

Der Default Service Type ist acknowledged.

5.3.7 Bewertete Windgeschwindigkeit

```
network output SNVT_speed nvoWindspeed;
```

Ist das Controller Objekt nicht im Alarmzustand, überträgt diese Ausgangs-NV die aktuelle Windgeschwindigkeit gemäß `nviWindspeed`. Sobald ein Alarmzustand eintritt, wird ein höherer Wert als die aktuelle Windgeschwindigkeit übertragen. Dieser überhöhte Wert ist größer als der parametrierbare obere Grenzwert der Windgeschwindigkeit.
Der Alarmzustand kann ermittelt werden, wie in 5.1 erläutert. Zusätzlich können abgelaufene Receiver Heartbeats der `nviWindspeed`, `nviAlarm`, `nviUnackAlarm0` oder `nviUnackAlarm1` zu Alarmen führen.

Valid Range

{0, 1, 2, ..., 65534} digits entsprechen {0, 0.1, 0.2, ..., 6553.4} m/s. 65535 entspricht INVALID.

When Transmitted

Diese NV wird übertragen, sobald deren Inhalt durch die Applikation geändert wurde, oder wenn der parametrierbare Send Heartbeat dieser NV abgelaufen ist.

Default Service Type

Der Default Service Type ist acknowledged.

5.4 Parametrierung

```
typedef struct {
      SNVT_speed    upperLimitWind;
      SNVT_speed    lowerLimitWind;
      SNVT_time_sec rcvHrtbtUnackAlarm0;
      SNVT_time_sec rcvHrtbtWind;
      SNVT_time_sec rcvHrtbtUnackAlarm1;
      SNVT_time_sec rcvHrtbtAckAlarm;
      SNVT_time_sec sndHrtbt;
      tMonitor      monitorFilter;   // was wird überwacht
      tCtrlDisplay  displayFilter;   // Zustand von nviAlarm oder
} tControllerConfig;                 // nvoIsAlarm anzeigen
```

5.4.1 Grenzwerte Windgeschwindigkeit

```
SNVT_speed upperLimitWind;
SNVT_speed lowerLimitWind;
```

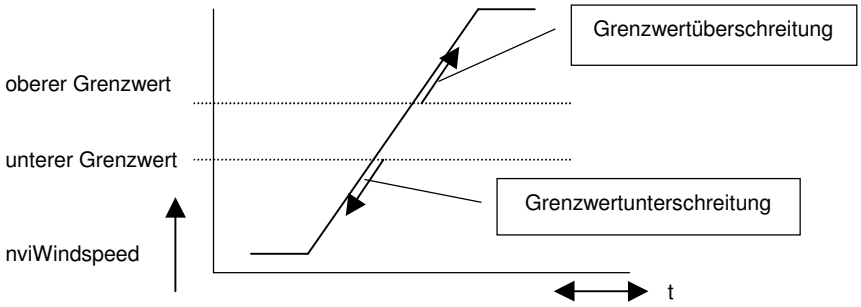

Abbildung 15

Valid Range

{0, 1, 2, ..., 65534} digits entsprechen {0, 0.1, 0.2, ..., 6553.4} m/s. 65535 entspricht INVALID.

5.4.2 Receiver Heartbeats

```
SNVT_time_sec rcvHrtbtUnackAlarm0; // Rcv. Hrtbt. der nviUnackAlarm0
SNVT_time_sec rcvHrtbtWind;        // Rcv. Hrtbt. der nviWindspeed
SNVT_time_sec rcvHrtbtUnackAlarm1; // Rcv. Hrtbt. der nviUnackAlarm1
SNVT_time_sec rcvHrtbtAckAlarm;    // Rcv. Hrtbt. der nviAlarm
```

Der Receiver Heartbeat legt fest, welche Zeit zwischen zwei aufeinanderfolgenden Übertragungen der jeweiligen NV vergehen darf. Nach Ablauf dieser Zeit wird via `nvoIsAlarm` des Controller Objekts und via `nvoAlarm` des Node Objekts der Receive Timeout Alarm ausgelöst (s. Tabelle 4).

Mit INVALID kann die Empfangsüberwachung einzelner NV´s ausgeschaltet werden.

Valid Range

{0, 10, 20, ..., 65530} digits entsprechen {0, 1, 2, ..., 6553} s. 65535 entspricht INVALID.

5.4.3 Send Heartbeat

```
SNVT_time_sec sndHrtbt;
```

Der Send Heartbeat legt den maximalen zeitlichen Abstand zwischen zwei aufeinanderfolgenden zyklischen Übertragungen der `nvoIsAlarm` fest.

Mit INVALID kann die Sendewiederholung der `nvoIsAlarm` ausgeschaltet werden.

Valid Range

{0, 10, 20, ..., 65530} digits entsprechen {0, 1, 2, ..., 6553} s. 65535 entspricht INVALID.

5.4.4 Überwachungsfilter

```
tMonitor monitorFilter;
```

Das Controller Objekt generiert Alarme durch Überwachung der Inhalte der Eingangs-NV´s und zugehöriger Receiver Heartbeats. Der Parameter `monitorFilter` bestimmt, welche Eingangs-NV überwacht werden soll.
So können z.B. die Überwachung und resultierende Alarme ausgeschaltet werden, ohne NV-Bindings ändern zu müssen.

Valid Range

0 : keine Überwachung
1 ($0000\ 0001_b$) : `nviAlarm`
2 ($0000\ 0010_b$) : `nviUnackAlarm1`
4 ($0000\ 0100_b$) : `nviWindspeed`
8 ($0000\ 1000_b$) : `nviUnackAlarm0`

Um die Überwachung mehrer NV´s einzuschalten, müssen diese Werte OR-verknüpft werden. Andere Werte als hier gelistet bzw. sich aus deren OR-Verknüpfungen ergeben, sind nicht zulässig.

Hinweis

Um die Überwachung einzelner NV´s vollständig auszuschalten, müssen sowohl das korrespondierende Bit auf 0 als auch der entsprechende Receiver Heartbeat auf INVALID gesetzt werden.

5.4.5 Anzeigefilter

```
tCtrlDisplay displayFilter;
```

Valid Range

0: Controller Objekt nutzt LED´s K5 bzw. K7 nicht
1: Anzeige des Zustands der `nviAlarm` (s. Abbildung 13)
2: Anzeige des Zustands der `nvoIsAlarm` (s. Abbildung 14)

Hinweis

Die Nutzung der LED´s K6 bzw. K8 lässt sich nicht ausschalten.

6 Scheduler Objekt

6.1 Funktionsweise

Das Scheduler Objekt kann zu bestimmten Zeitpunkten (Datum + Uhrzeit oder Wochentag + Uhrzeit) Szenen wiederholt oder einmalig senden. Für Sendewiederholungen lassen sich der zeitliche Abstand zwischen den Wiederholungen und der Endzeitpunkt der Wiederholungen parametrieren.

Es sind 50 verschiedene Szenen und deren jeweiligen Sendebedingungen parametrierbar.

Zur korrekten Funktionsweise des Scheduler Objekt muss die nviTimestamp des Node Objektes mit dem entsprechenden NV-Ausgang einer Uhr o.ä. gebunden sein.

Priorisierung der Szenen

Wird zum Startzeitpunkt einer Szene deren Sendewiederholung gestartet, werden alle bis dahin laufenden Sendewiederholungen mit niedrigerer Numerierung unterdrückt.

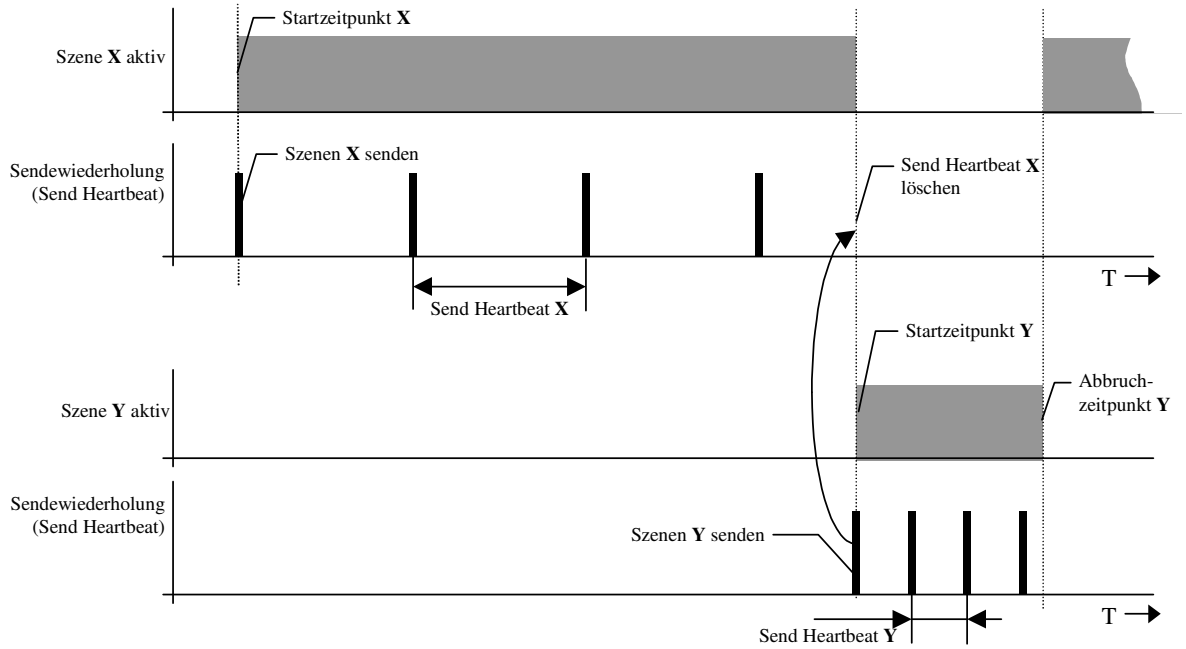

Abbildung 16

Voraussetzungen für das Verhalten entspr. Abbildung 16 sind:

- Index des Parametersatzes für Szene X ist kleiner als der Index des Parametersatzes Y
- Abbruchzeit von X später als Abbruchzeit von Y oder keine Abbruchzeit von X parametriert
- Abbruchzeiten für X und Y beide für den selben Tag wie jeweilige Startzeit

Der Index des Parametersatzes für Szene X ist kleiner als der Index des Parametersatzes Y und für X ist keine Abbruchzeit parametriert. Somit kann nach Überschreiten des Abbruchzeitpunktes für Y die Szene des Parametersatzes X weitergesendet werden.

Gleiches Verhalten wird auch dann erreicht, wenn eine Abbruchzeit für X parametriert ist und diese Abbruchzeit später als die Abbruchzeit von Y ist.

6.2 Schnittstellenbeschreibung

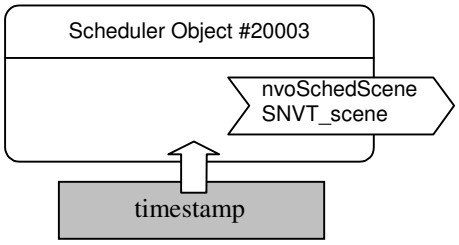

Abbildung 17 (Interface Scheduler Objekt)

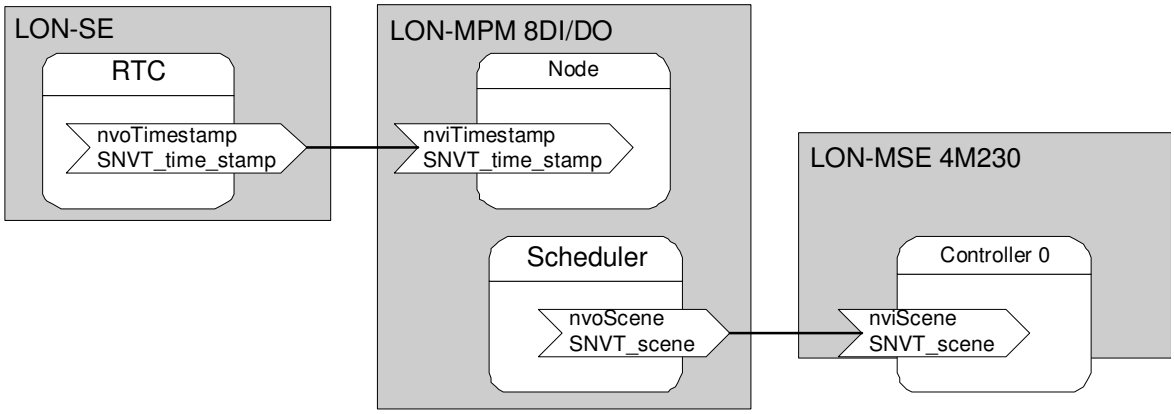

Abbildung 18 (Beispielbinding des Scheduler Objektes)

6.2.1 Szenen Ausgang

`network output SNVT_scene nvoSchedScene;`

Diese Ausgangs-NV sendet.

Valid Range

Gültige Werte sind alle Werte in den definierten Grenzen gemäß `SNVT_scene`. Die Bedeutung der Felder der `SNVT_scene` sind erklärt in ***[1]***.

When Transmitted

Diese NV wird übertragen, sobald deren Inhalt durch die Applikation geändert wurde, oder wenn der parametrierbare Send Heartbeat dieser NV abgelaufen ist.

Default Service Type

Der Default Service Type ist acknowledged.

6.3 Parametrierung

```
typedef struct {
       unsigned int iItem;              // Index Parametersatz
       union {
           SNVT_time_stamp dayDate;
           tDayOfWeek      dayOfWeek;
           tWeekday        weekDay;
       } start;                         // Startzeit zum Senden der scene
       union {
           SNVT_time_stamp dayDate;
           tDayOfWeek      dayOfWeek;
           tWeekday        weekDay;
       } end;                           // Abbruchzeitpunkt
       SNVT_scene    scene;             // zu sendende Szene
       SNVT_time_sec sndHrtbt;          // send heartbeat
} tRtbSchedulerItem;
```

Für das Scheduler Objekt ist eine Liste von Parametersätzen hinterlegt. Jeder Parametersatz ist formatiert wie `tRtbSchedulerItem` und enthält seinen Index, eine Szene und deren Sendebedingungen.

Der Parameter `iItem` dient der fortlaufenden Numerierung von Parametersätzen bei Lese- und Schreibzugriffen auf die Liste der Szenen und deren jeweiligen Sendebedingungen. Der Wertebereich von iItem ist {0, 1, ..., 49}. Je höher die Nummer eines Parametersatzes ist, um so höher ist seine Priorität (s.a. Abbildung 16).

6.3.1 Szene

```
SNVT_scene scene;
```

Dieser Parameter bezeichnet die Szene, die zwischen Startzeit und Abbruchzeit gesendet werden soll.

Valid Range

Gültige Werte sind alle Werte in den definierten Grenzen gemäß `SNVT_scene`. Die Bedeutung der Felder der `SNVT_scene` sind erklärt in ***[1]***.

6.3.2 Startzeit

```
typedef enum {
    MONTAG     = 0,
    DIENSTAG   = 1,
    MITTWOCH   = 2,
    DONNERSTAG = 3,
    FREITAG    = 4,
    SAMSTAG    = 5,
    SONNTAG    = 6,
    INVALID    = -1 (FF_hex)
} tDayOfWeek;

typedef enum {
    WOCHENTAG  = -128 (80hex)
    WOCHENENDE = -112 (90hex)
} tWeekday;

union {
    SNVT_time_stamp dayDate;   // 7 Bytes
    tDayOfWeek      dayOfWeek; // 1 Byte, am weitesten links
    tWeekday        weekDay;   // 1 Byte, am weitesten links
} start;                       // insgesamt 7 Bytes
```

Dieser Parameter bestimmt den Zeitpunkt, zu dem eine Szene gesendet werden bzw. wann das wiederholte Senden beginnen soll. Er ist nur relevant, wenn der Send Heartbeat auf einen gültige Wert parametriert ist.

Die Startzeit `start` ist als `union` deklariert ist. Somit ist die Startzeit entweder wie `dayDate` oder wie `dayOfWeek` oder wie `weekDay` formatiert. Als Startzeitpunkt lässt sich festlegen: eine Uhrzeit:

a) entweder an einem bestimmen Datum ein einem bestimmten Jahr (`dayDate`)

b) oder an einem bestimmten Datum jedes Jahr (`dayDate`)

c) oder jeden Wochentag/Wochenendtag (`dayOfWeek`)

d) oder ein bestimmter Wochentag in jeder Woche (`weekDay`)

Valid Range

`dayDate` :
(Fälle a), b)) Gültige Werte sind alle Werte in den definierten Grenzen gemäß `SNVT_time_stamp`. Die Bedeutung der Felder der `SNVT_time_stamp` sind erklärt in [1].
Wenn `dayDate.year` größer 2000 ist, gelten alle 7 Bytes von `start`. Der Parameter `start` ist dann formatiert wie `SNVT_time_stamp`.
Wenn `dayDate.year` gleich 2000 ist, wird mit Senden der Szene begonnen jedes

Jahr zum Datum gemäß `dayDate.month`, `dayDate.day`, `dayDate.hour`, `dayDate.minute`, `dayDate.second`.

Beispiel 1: 07 D2 02 14 08 09 0A bedeutet Sendebeginn am 20. Februar 2002 um 08.09:10 Uhr

Beispiel 2: 07 D0 02 14 09 0A 0B bedeutet Sendebeginn jeden 20. Februar um 09.10:11 Uhr

`dayOfWeek` : (Fall c)) Gültige Werte sind alle Werte in den definierten Grenzen gemäß `tDayOfWeek`. Wenn das erste Byte in `start` im Wertebereich gemäß `tDayOfWeek` liegt, ist dieses Byte formatiert wie `tDayOfWeek`; die nächsten 3 Bytes werden dann nicht betrachtet. Die weiteren 3 Bytes repräsentieren die Uhrzeit.

Beispiel 3: 02 xx xx xx 08 09 0A bedeutet Sendebeginn jeden Mittwoch um 08.09:10 Uhr (die mit xx besetzten Bytes werden nicht betrachtet)

`weekDay` : (Fall d)) Gültige Werte sind alle Werte in den definierten Grenzen gemäß `tWeekday`. Wenn das erste Byte von `start` im Wertebereich gemäß `tWeekday` liegt, ist dieses das Byte formatiert wie `tWeekday`; die nächsten 3 Bytes werden dann nicht betrachtet. Die weiteren 3 Bytes repräsentieren die Uhrzeit.

Beispiel 4: 80 xx xx xx 08 09 0A entspricht jedem Wochentag, 08.09:10 Uhr (die mit xx besetzten Bytes werden nicht betrachtet)

Beispiel 5: 90 xx xx xx 0A 09 08 entspricht jedem Wochenendtag, 10.09:08 Uhr (die mit xx besetzten Bytes werden nicht betrachtet)

Der komplette Parametersatz `start` lässt sich als nicht genutzt markieren, indem `dayOfWeek` auf INVALID (FF_{hex}) parametriert wird.

Hinweise

Identische Einträge in verschiedenen Parametersätzen sollten vermieden werden.

Sollten dennoch identische Einträge existieren, wird zum gegebenen Zeitpunkt derjenige mit dem höchsten Index `iItem` ausgeführt (Priorität, s.a. Abbildung 16).

Der zeitliche Abstand zwischen Startzeitpunkt und der Uhrzeit 23.59:59 sollte mindestens so groß gewählt werden, dass während dieses zeitlichen Zwischenraumes die `nviTimestamp` mindestens einmal empfangen werden kann.

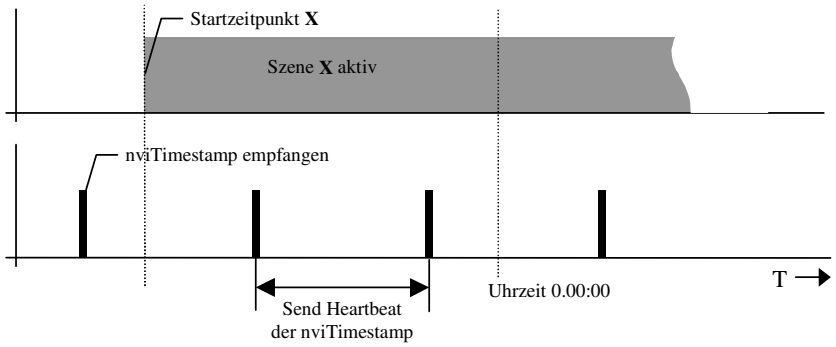

Abbildung 19

6.3.3 Abbruchzeit

```
union {
    SNVT_time_stamp dayDate;    // 7 Bytes
    tDayOfWeek      dayOfWeek; // 1 Byte, am weitesten links
    tWeekday        weekDay;   // 1 Byte, am weitesten links
} end;                         // insgesamt 7 bytes
```

Dieser Parameter bestimmt den Zeitpunkt, an dem das wiederholte Senden beendet sein soll.

Die Abbruchzeit `end` ist als `union` deklariert ist. Somit ist die Abbruchzeit entweder wie `dayDate` oder wie `dayOfWeek` oder wie `weekDay` formatiert. Als Abbruchzeitpunkt lässt sich festlegen: eine Uhrzeit:

a) entweder an einem bestimmen Datum ein einem bestimmten Jahr (dayDate)

b) oder an einem bestimmten Datum jedes Jahr (`dayDate`)

c) oder jeden Wochentag/Wochenendtag (`dayOfWeek`)

d) oder ein bestimmter Wochentag in jeder Woche (`weekDay`)

Valid Range

`dayDate` :
(Fall a), b))
Gültige Werte sind alle Werte in den definierten Grenzen gemäß `SNVT_time_stamp`. Die Bedeutung der Felder der `SNVT_time_stamp` sind erklärt in [1].
Wenn `dayDate.year` größer 2000 ist, gelten alle 7 Bytes von `start`. Der Parameter `end` ist dann formatiert wie `SNVT_time_stamp`.
Wenn `dayDate.year` gleich 2000 ist, wird das Senden abgebrochen in jedem Jahr zum Datum gemäß `dayDate.month`, `dayDate.day`, `dayDate.hour`, `dayDate.minute`, `dayDate.second`.
Beispiel 1: 07 D2 02 14 12 09 0A bedeutet Abbruch am 20. Februar 2002 um 18.09:10 Uhr
Beispiel 2: 07 D0 02 14 13 0A 0B bedeutet Abbruch jeden 20. Februar um 19.10:11 Uhr

`dayOfWeek` :
(Fall c))
Gültige Werte sind alle Werte in den definierten Grenzen gemäß `tDayOfWeek`.
Wenn das erste Byte in `end` im Wertebereich gemäß `tDayOfWeek` liegt, ist dieses Byte formatiert wie `tDayOfWeek`; die nächsten 3 Bytes werden dann nicht betrachtet. Die weiteren 3 Bytes repräsentieren die Uhrzeit.
Beispiel 3: 02 xx xx xx 12 09 0A bedeutet Sendeabbruch jeden Mittwoch um 18.09:10 Uhr (die mit xx besetzten Bytes werden nicht betrachtet)

`weekDay` :
(Fall d))
Gültige Werte sind alle Werte in den definierten Grenzen gemäß `tWeekday`.
Wenn das erste Byte von `end` im Wertebereich gemäß `tWeekday` liegt, ist dieses das Byte formatiert wie `tWeekday`; die nächsten 3 Bytes werden dann nicht betrachtet. Die weiteren 3 Bytes repräsentieren die Uhrzeit.
Beispiel 4: 80 xx xx xx 08 09 0A entspricht jedem Wochentag, 08.09:10 Uhr (die mit xx besetzten Bytes werden nicht betrachtet)
Beispiel 5: 90 xx xx xx 0A 09 08 entspricht jedem Wochenendtag, 10.09:08 Uhr (die mit xx besetzten Bytes werden nicht betrachtet)

Der komplette Parametersatz `end` lässt sich als nicht genutzt markieren, indem `dayOfWeek` auf INVALID (FF_{hex}) parametriert wird. INVALID bedeutet in diesem Fall, dass die Abbruchzeit im Unendlichen liegt.

Hinweise

Identische Einträge in verschiedenen Parametersätzen sollten vermieden werden.

Sollten dennoch identische Einträge existieren, wird zum gegebenen Zeitpunkt derjenige mit dem höchsten Index `iItem` ausgeführt.

Die Abbruchzeit wird nur berücksichtigt, wenn auch die Startzeit des selben Parametersatz auf einen gültigen Zeitpunkt parametriert ist.

Angenommen Start- und zugehörige Abbruchzeit sind nicht auf das selbe Datum oder den selben Wochentag parametriert. Starten dann Anwendung oder Scheduler Objekt an einem Tag nach der Startzeit, so löst diese keinen Sendebeginn aus. Daher sollten Start- und zugehörige Abbruchzeit immer für den selben Tag parametriert werden.

6.3.4 Send Heartbeat

```
SNVT_time_sec sndHrtbt;
```

Valid Range

{0, 10, 20, ..., 65530} digits entsprechen {0, 1, 2, 6553} Sekunden.
65535 entspricht INVALID (Sendewiederholung Aus).

7 Übersicht Configuration Properties

Folgende Tabelle listet alle Configuration Properties dieser Anwendung auf.

Name[1]	**Bedeutung**	**Referenz**[2]
SCPTdevMajVer	Version der Applikation	Node Objekt je 1x pro Gerät
SCPTdevMinVer	Revision der Applikation	
SCPTmaxSendTime	max. Zeit zwischen 2x Senden der nvoAlarm	
SCPTmaxRcvTime	max. Zeit zwischen 2x Empfang der nviTimestamp	
UCP_type_1	Liste der Parametersätze	Scheduler Objekt je 1x pro Gerät
SCPTmaxRnge	oberer Grenzwert Windgeschwindigkeit	Controller Objekt je 1x pro Controller
SCPTminRnge	unterer Grenzwert Windgeschwindigkeit	
SCPTmaxRcvTime	max. Zeit zwischen 2x Empfang der nviUnackAlarm0	
SCPTmaxRcvTime	max. Zeit zwischen 2x Empfang der nviWindspeed	
SCPTmaxRcvTime	max. Zeit zwischen 2x Empfang der nviUnackAlarm1	
SCPTmaxRcvTime	max. Zeit zwischen 2x Empfang der nviAlarmAckd	
SCPTmaxSendTime	max. Zeit zwischen 2x Senden der nvoIsAlarm und nvoWindspeed	
UCP_type_2	Überwachungsmodus	
UCP_type_3	Displaymodus	
UCP_type_4	Freischaltung Kommando „Szenenlernen“	Scene Panel Objekt je 1x pro Scene P.
SCPTmaxSendTime	max. Zeit zwischen 2x Empfang der nvoScene und nvoSetting	
SCPTsceneNmbr	zu sendende Szenennummer bei Tastendruck	
UCP_type_5	zu sendende Szenennummer nach Taste losgelassen	
UCP_type_6	zu sendende SNVT_setting-Wert bei Tastendruck	
UCP_type_7	zu sendende SNVT_setting-Wert nach Taste losgelassen	
UCP_type_8	Art der logischen Verknüpfung im Switch Objekt	Switch Objekt je 1x pro Switch
SCPTmaxSendTime	max. Zeit zwischen 2x Empfang der nvoSwitch	

Tab. 16 (Übersicht Configuration Properties)

Die Formate der Configuration Properties stimmen mit den Formaten überein, die in den Abschnitten zur Parametrierung der einzelnen Objekte beschrieben sind. Anhand dieser Abschnitte können somit auch die Configuration Properties genutzt werden, deren Name lediglich durch UCP_type_ und Nummer des User Defined Configuration Types repräsentiert wird.

Um detaillierte Formate der Configuration Properties anzeigen bzw. nutzen zu können, müssen die mit der Anwendung gelieferten Resource Files in den Device Resource File Catalog eingebunden sein. Das Einbinden kann z.B. mit dem LonMark Device Resource File Catalog Browser (ldrfcat.exe) der Echelon Corp. durchgeführt werden.

Weitere Angaben sollten nicht erforderlich sein, da zur Parametrierung eines Gerätes ein Geräte-Plug-In zur Verfügung steht bzw. einzelne Configuration Properties mit Hilfe von Kopier- und Vervielfältigungsfunktionen des Installationstools eingestellt werden können.

[1] Die Namen der einzelnen Configuration Properties entsprechen den Namen in den Resource Files.
[2] Unter Referenz ist aufgelistet, bei welchen Objekten bzw. NV´s die Config. Property wie oft vorhanden ist.

8 Ausdruck der Datei 16911210.XIF

```
File: 16911210.XIF generated by APC Revision 2.74, XIF Version 3.1
Copyright (c) 1990, 1996  by Echelon Corporation
All Rights Reserved.  Run on Mon Sep 03 09:06:34 2001

90:00:57:05:50:0A:04:01
2 15 0 21 0 3 3 0 0 5 5 11 11 9 9 0 0 16 16 1 1 128
0 5 6 13 28 920 0 15 5 3 341 4
1 7 1 0 4 4 4 15 200 0
78125 0 0 0 0 0 0 0 0 0 0 0
90 0 240 0 0 0 40 40 0 5 8 5 12 14 15
*
"&3.1@0,3200[8]Switch,3250[4]Scene Panel,5[2]Conditionl Alarm
",20003[1]Scene Scheduler;REKO LON-8DIDO 2.0 A

VAR nviRequest 0 0 0 0
0 1 63 0 0 1 0 1 0 0 0 0 0
"@0|1
92 *   2
2 0 0 0 0
1 0 0 1 0
VAR nvoStatus 1 0 0 0
0 1 63 1 0 1 0 1 0 0 0 0 0
"@0|2
93 *  26
2 0 0 0 0
3 0 1 0 0
3 1 1 0 0
3 2 1 0 0
3 3 1 0 0
3 4 1 0 0
3 5 1 0 0
3 6 1 0 0
3 7 1 0 0
3 0 1 0 0
3 1 1 0 0
3 2 1 0 0
3 3 1 0 0
3 4 1 0 0
3 5 1 0 0
3 6 1 0 0
3 7 1 0 0
3 0 1 0 0
3 1 1 0 0
3 2 1 0 0
3 3 1 0 0
3 4 1 0 0
3 5 1 0 0
3 6 1 0 0
3 7 1 0 0
3 0 8 0 0
VAR nviTimestamp 2 0 0 0
0 1 63 0 0 1 0 1 0 0 0 0 0
"@0|3;time sync input
84 *   6
2 0 0 0 0
1 0 0 0 0
1 0 0 0 0
1 0 0 0 0
1 0 0 0 0
1 0 0 0 0
VAR nvoAlarm 3 0 0 0
0 1 63 1 0 1 0 1 0 0 0 0 0
"@0|4
88 *  14
0 0 0 0  6
2 0 0 0 0
1 0 0 1 0
```

```
1 0 0 1 0
2 0 0 0 0
1 0 0 0  4
2 0 0 0 0
1 0 0 0 0
1 0 0 0 0
1 0 0 0 0
1 0 0 0 0
1 0 0 0 0
2 0 0 0 0
1 0 0 0  4
VAR nviConfig 4 0 0 0
0 1 63 0 0 1 0 1 0 0 0 0 0
"@0#100
0 *   4
1 0 0 1 0
1 0 0 1 0
1 0 0 1 0
4 0 28 0 0
VAR nvoSwitch 5 0 0 8
0 1 63 1 0 1 0 1 0 0 0 0 0
"@1-8|1
95 *   2
1 0 0 0 0
1 0 0 1 0
VAR nviSwitchFB 13 0 0 8
0 1 63 0 0 1 0 1 0 0 0 0 0
"@1-8|2
95 *   2
1 0 0 0 0
1 0 0 1 0
VAR nvoScene 21 0 0 4
0 1 63 1 0 1 0 1 0 0 0 0 0
"@9-12|1
115 *   2
1 0 0 1 0
1 0 0 0 0
VAR nviSceneFb 25 0 0 4
0 1 63 0 0 1 0 1 0 0 0 0 0
"@9-12|2
115 *   2
1 0 0 1 0
1 0 0 0 0
VAR nvoSetting 29 0 0 4
0 1 63 1 0 1 0 1 0 0 0 0 0
"@9-12|3
117 *   3
1 0 0 1 0
1 0 0 0 0
2 0 0 1 0
VAR nviSetting 33 0 0 4
0 1 63 0 0 1 0 1 0 0 0 0 0
"@9-12#100
117 *   3
1 0 0 1 0
1 0 0 0 0
2 0 0 1 0
VAR nviSwitch 37 0 0 4
0 1 63 0 0 1 0 1 0 0 0 0 0
"@9-12#101
95 *   2
1 0 0 0 0
1 0 0 1 0
VAR nviUnackAlarm0 41 0 0 2
0 1 63 0 0 1 0 1 0 0 0 0 0
"@13-14|3
95 *   2
1 0 0 0 0
1 0 0 1 0
VAR nviUnackAlarm1 43 0 0 2
```

```
0 1 63 0 0 1 0 1 0 0 0 0 0
"@13-14#102
95 *   2
1 0 0 0 0
1 0 0 1 0
VAR nviAlarm 45 0 0 2
0 1 63 0 0 1 0 1 0 0 0 0 0
"@13-14#103
95 *   2
1 0 0 0 0
1 0 0 1 0
VAR nviAlarmAckd 47 0 0 2
0 1 63 0 0 1 0 1 0 0 0 0 0
"@13-14#105
95 *   2
1 0 0 0 0
1 0 0 1 0
VAR nviWindspeed 49 0 0 2
0 1 63 0 0 1 0 1 0 0 0 0 0
"@13-14#101
34 *   1
2 0 0 0 0
VAR nvoIsAlarm 51 0 0 2
0 1 63 1 0 1 0 1 0 0 0 0 0
"@13-14|1
95 *   2
1 0 0 0 0
1 0 0 1 0
VAR nvoWindspeed 53 0 0 2
0 1 63 1 0 1 0 1 0 0 0 0 0
"@13-14#104
34 *   1
2 0 0 0 0
VAR nvoSchedScene 55 0 0 0
0 1 63 1 0 1 0 1 0 0 0 0 0
"@15|1
115 *   2
1 0 0 1 0
1 0 0 0 0
VAR ncoAddress 56 0 0 0
0 1 63 1 0 1 0 1 0 0 0 0 0
"@0|8
114 *   1
2 0 0 0 0
```

9 Stichwortverzeichnis

10 Quellenangaben

[1] SNVT Master List and Programmers Guide, Echelon Corporation, May 1997
[2] LONMARK Application Layer Interoperability Guidelines Version 3.1, LONMARK Interoperability Association, 1992-1998
[3] Anwendungsbeispiele der Applikation 16911210, WAREMA Renkhoff SE, 2008