DOKUMENTATION

LCN-Binding für openHAB

ISSENDORFF

Autor: Patrik Pastuschek
Stand: 14.04.2015

Inhaltsverzeichnis

1 Einleitung

Das LCN-Binding für openHAB wurde entwickelt um überlegene LCN Technologie mit der Vielfältigkeit von openHAB zu kombinieren. Diese Dokumentation wird den Leser bei der Installation und Einrichtung des LCN-Bindings, jedoch nicht bei der Installation von openHAB selbst, unterstützen. Für weitere Informationen zur Installation von openHAB konsultieren Sie bitte die openHAB homepage, sowie das openHAB wiki.
Sobald Sie openHAB korrekt installiert haben, kann die Installation des LCN-Binding beginnen. Bitte folgen Sie den Anweisungen genau um möglichen Fehlern vorzubeugen.
Beachten Sie, dass das LCN-Binding Java 7 (JRE 1.7) oder neuer benötigt. Ohne die korrekte Java Version, wird das LCN-Binding nicht richtig funktionieren. Außerdem wird eine Installation der neusten LCN PCHK Software (mindestens Version 2.8) vorausgesetzt.

Das LCN-openHAB-binding benötigt PCHK version 2.8 oder neuer!

Falls Sie nicht die richtigen Versionen von Java und PCHK installieren, wird das LCN-Binding nicht richtig funktionieren!

2 Konfiguration des LCN-Bindings

Die folgenden Informationen werden Ihnen helfen, das LCN-Binding richtig zu konfigurieren. Im Allgemeinen gibt es drei Definitionsebenen, welche benutzt werden können um sogenannte Items und die dazu gehörigen Bindings in openHAB zu definieren.
Die Hauptkonfigurationsdatei enthält Einstellungen, welche das LCN-Binding im Allgemeinen konfigurieren, während die Item- und Sitemap-Definitionen die einzelnen Items definieren.
Zunächst muss jedoch das LCN-Binding installiert werden, dazu reicht es aus das LCN-Binding in den 'addons' Ordner der openHAB Installtion zu kopieren. Falls trotzdem Probleme bei der Installation des LCN-Bindings auftreten, hilft ihnen der Installations Guide weiter.

2.1 Allgemeine Konfigurationen

Bevor LCN-Bindings definiert werden können ist es notwendig die 'openhab.cfg' anzupassen, welche sich im 'configurations' Ordner der openHAB Installation befindet.
In der 'openhab.cfg' sollten Sie einen eigenen Bereich nur für das LCN-Binding finden. Falls ein solcher Bereich nicht existiert können Sie die folgenden Befehle auch manuell einfügen. Die einzigen Einstellungen, welche immer benötigt werden um das LCN-Binding zu benutzen sind:

```
lcn:id1=local
lcn:address1=127.0.0.1:4114
lcn:username1=max
lcn:password1=maximum
lcn:mode1=default
```

Eine beliebige Anzahl an id-address-username-password Kombinationen ist möglich, achten Sie darauf die Bezeichner zu inkrementieren (i.e. id1, id2, … address1, address2, …). Überspringen Sie keine Zahl beim inkrementieren und beginnen Sie stets mit 1, andernfalls können die Informationen nicht richtig eingelesen werden. Außerdem müssen sämtliche IDs einzigartig sein. In dem Beispiel würden die Adresse '127.0.0.1:4114', der Benutzername 'max' und das Password 'maximum' an die ID 'local' gebunden. Falls Sie den Port der Adresse nicht definieren (der durch den Doppelpunkt getrennt Teil), so wird der Standard Port 4114 eingesetzt.
Mit 'mode' wird der Modus des LCN-Busses eingestellt, dabei kann zwischen 'default' und 'advanced' gewählt werden. Der 'advanced' Modus ist allerdings nur für Systeme verfügbar, welche **ausschließlich** Module mit einer Firmware von 160B13 (Dez. 2012) aufwärts, beinhalten.
Der Unterschied zwischen den beiden Modi besteht darin, dass bei 'advanced' die Dimmwerte eine vierfach höhere Auflösung (0-200 statt 0-50) haben, als bei 'default'. Für die weitere Nutzung des Bindings spielt dies aber keine Rolle.
Falls Sie sich nicht sicher sind in welchem Modus der Bus derzeit läuft, können Sie auch den Wert 'unknown' setzen, dann wird der derzeitige Modus beibehalten. Alternativ können Sie auch die gesamte Zeile auslassen und den Wert überhaupt nicht setzen.

Darüber hinaus gibt es noch eine optionale Konfiguration, welche das Ping Verhalten reguliert. Wenn PCK innerhalb eines bestimmten Zeitraums (definierbar direkt in der PCK Software) keinen Befehl erhält, wird die Verbindung geschlossen. Das LCN-Binding stellt

die Verbindung automatisch wieder her, allerdings kommt es dabei zu einer kurzen Verzögerung wenn zu diesem Zeitpunkt Befehle gesendet werden müssen, da sich das LCN-Binding erst wieder bei der PCK anmelden muss. Um dies zu verhindern sendet das LCN-Binding in regelmäßigen Abständen einen Ping an die PCK. Dieser Abstand (angegeben in Sekunden) wird definiert durch die folgende Konfiguration:

```
lcn:ping=600
```

Dieses Beispiel setzt den Abstand zwischen zwei Pings auf 600 (10 Minuten). Wird dieser Wert nicht gesetzt, wird der Standardwert 600 übernommen. Wenn überhaupt keine Pings gesendet werden sollen, setzen Sie den Wert bitte auf 0.

2.2 Item Definitionen

Items werden in speziellen '*.Items' Dateien definiert, welche sich im Ordner '…/configurations/Items/' befinden. Ein normales Item-binding in openHAB mit dem LCN-Binding, sieht etwa so aus:

```
Switch Light_Test "Light" {lcn="[ON:local:ON.0.23.2], [OFF:local:'MAP(test.map)']"}
```

Ein Teil der Syntax sollte vom allgemeinen openHAB-Gebrauch bekannt sein, trotzdem werden alle Teile hier noch einmal kurz erklärt.

```
Switch Light_Test "Light" {lcn="[ON:local:ON.0.23.2], [OFF:local:'MAP(test.map)']"}
```

Definiert das Item für openHAB. 'Switch' (Schalter) ist der Typ des Items, während 'Light_Test' der Bezeichner ist.

```
Switch Light_Test "Light" {lcn="[ON:local:ON.0.23.2], [OFF:local:'MAP(test.map)']"}
```

Definiert den Namen des Items, welcher später in der grafischen Oberfläche sichtbar ist.

```
Switch Light_Test "Light" {lcn="[ON:local:ON.0.23.2], [OFF:local:'MAP(test.map)']"}
```

Dies ist die tatsächliche LCN-Binding-Definition. Jede LCN-Binding-Definition beginnt immer mit einem 'lcn=' und wird gefolgt von einer, oder mehreren LCN-Binding-Definitionen. Ein Binding muss der folgenden Syntax folgen, damit das Programm das Binding lesen kann:

Syntax: [<openHAB_cmd>:]<id>:<lcn_cmd>

Ein LCN-Binding kann ein openHAB Kommando beinhalten, welcher dazu dienen kann, das LCN-Kommando an eine bestimme Aktion in openHAB zu binden. Im obigen Beispiel wurde der eine Befehl an das openHAB Kommando 'ON' gebunden, während der andere an 'OFF' gebunden wurde. Da es sich bei dem Item um einen Schalter (Switch) handelt, wird der 'ON' Befehl ausgeführt wenn der Schalter eingeschaltet, und der 'OFF' Befehl wenn der Schalter ausgeschaltet wird.
Trotzdem bleibt das openHAB Kommando optional und wird nicht benötigt um ein LCN-Binding zu definieren. Dies gilt insbesondere dann, wenn das LCN-Binding Daten empfangen soll (z.B. von einem Sensor).

Die ID wird benötigt um die LCN-Kommandos an das richtige System zu schicken. IDs werden in der allgemeinen Konfiguration definiert, wobei IP, Port, Benutzername und Password an eine ID gebunden werden.

Der LCN Befehl kann ein eigentlicher LCN Befehl sein oder ein sogenanntes Mapping über eine Map. In unserem Beispiel verwenden wir die Map 'test.map', welche im Ordner '...\configurations\transform' vorhanden sein muss. Um ein Mapping benutzen zu können wird jedoch stets ein openHAB-Kommando benötigt, da dieses als Schlüssel innerhalb der Map benutzt wird. Die Map könnte zum Beispiel so aussehen:

```
undefined=unknown
ON=ON.0.23.2
OFF=OFF.0.23.2
INCREASE=ADD.0.23.2.4
DECREASE=SUB.0.23.2.4
```

Wird also das zweite LCN-Binding gerufen, wird tatsächlich der Befehl 'OFF.0.23.2' benutzt. Darüber hinaus können die LCN Befehle noch flexibler gestaltet werden, indem der Platzhalter '%i' benutzt wird.
Mit diesem Platzhalter können einzelne Parameter des LCN Befehls erst zur Laufzeit bestimmt werden ohne dass mehrere LCN-Bindings oder sogar Items nötig sind.

```
String Flicker_Test "Flicker" {lcn="[local:FLICKER.0.23.2.%i.%i.5]"}
```

Wie genau die Variablen definiert werden, welche den Platzhalter ersetzen können, ist Teil der Sitemap Definition im folgenden Kapitel. Beachten Sie bitte, dass Items, welche den Platzhalter '%i' benutzen sollen, im allgemeinen als 'String' Item definiert werden sollten. Grund dafür ist, dass der Platzhalter durch ein openHAB Kommando (oder einen Teil davon) ersetzt wird. Die meisten Items können nur einige wenige openHAB Kommandos erhalten, während String Items beinahe jeden Befehl empfangen können.
Die Art und Weise wie ein Item in der grafischen Oberfläche tatsächlich dargestellt wird, wird durch Sitemap Definitionen entschieden.

2.3 Sitemap Definitionen

Im allgemeinen sind die Sitemap Definitionen relativ simpel und identisch mit den gewöhnlichen openHAB Sitemap Definitionen. Falls Sie noch nicht mit openHAB Sitemaps gearbeitet haben, schauen Sie sich bitte die entsprechende Wiki Seite des openHAB Wikis an. Die zuvor genannten Platzhalter Variablen können durch Mapping Definitionen bestimmt werden. Eine solche Definition könnte etwa so aussehen:

```
Switch item=Flicker_Test mappings=[LOW_SLOW="LOW and SLOW", MEDIUM_MEDIUM="MEDIUM and MEDIUM",
HIGH_FAST="HIGH and FAST"]
```

Die Sitemap Definition umhüllt unser String Item mit einem Switch Item, dadurch kombinieren wir das String Item, welches jeden beliebigen Befehl empfangen darf, mit der Darstellung eines Schalters (welcher unter normalen Umständen lediglich 'ON' und 'OFF' empfangen könnte).
Jedes Mapping besteht aus zwei Elementen. Ein openHAB Kommando und ein

Bezeichner. Der Bezeichner bestimmt, wie der entsprechende Knopf später in der grafischen Oberfläche heißt, während das openHAB Kommando später an unser Binding gesendet wird (und falls vorhanden, einen oder mehrere Platzhalter erstetzt).
Um eine noch höhere Flexibilität zu erreichen, kann ein openHAB Kommando einen oder mehrere Unterstriche '_' enthalten. An dieser Stelle wird das Kommando später aufgetrennt und an die entsprechenden Platzhalter übergeben.

Drücken des Knopfes "LOW and SLOW" würde also zu folgendem LCN-Kommando führen:
FLICKER.0.23.2.LOW.SLOW.5
Ein Druck auf den Knopf "HIGH and FAST" hingegen, würde zu folgendem führen:
FLICKER.0.23.2.HIGH.FAST.5
Beachten Sie, dass diese Technik Einfluss auf die automatischen Item Updates haben kann, da das Binding (z.B. im Falle eines variablen Segments) das Update nicht sicher zuordnen kann.

3 Schnellreferenz

Dieser Abschnitt dient zum Nachschlagen der Syntax eines bestimmten LCN-Kommandos. Während die grundsätzliche Syntax zwar vorhanden ist, wird die genaue Bedeutung einzelner Teile des Kommandos nicht behandelt. Falls Sie genauere Erläuterungen zu einem Kommando benötigen, konsultieren Sie bitte das Kapitel "4 LCN Kommandos", wo jedes einzelne Kommando noch einmal spezifisch behandelt wird.

3.1 LCN-Kommandos

Die folgende Tabelle enthält alle unterstützten LCN-Kommandos. In der linken Spalte ("Name") befindet sich der Name des jeweiligen Kommandos, während in der rechten Spalte ("Syntax") die Syntax für das Kommando zu finden ist. Dabei besteht die Syntax aus verschiedenen Bezeichnern (wie z.B. "segment"). Um ein Kommando zu konstruieren, müssen sämtliche Bezeichner durch entsprechende Werte und Terme ersetzt werden, welche in der Tabelle "3.2 LCN-Bezeichner" näher erläutert werden.

Bitte beachten Sie, dass sie nach jedem Kommandonamen eine Zielspezifikation einfügen können. Diese Spezifikation kann entweder "GROUP" oder "MODULE" sein und ist optional. Wenn Sie keine Spezifikation angeben, nimmt das Programm an, dass ein "MODULE" gemeint war.
Mit "GROUP" können Sie zuvor definierte Gruppen, anstatt nur eines einzelnen Moduls, ansprechen. Diese Gruppen können mit dem Befehl "GROUPS" definiert werden.
Beispiel anhand des "ON" Kommandos:

```
ON.15.35.2
```

Würde Ausgang 2, des Moduls 35, in Segment 15, einschalten.
Es wäre ebenfalls möglich diesen Befehl als "ON.MODULE.15.35.2" zu schreiben.

```
ON.GROUP.15.35.2
```

Würde den Ausgang 2, aller Module in Gruppe 35, in Segment 15, einschalten.

Available Commands:	
Name:	**Syntax:**
ON	ON.segment.module.output[.ramp]
OFF	OFF.segment.module.output[.ramp]
DIM	DIM.segment.module.output.percent.ramp
TOGGLE	TOGGLE.segment.module.output[.ramp]
ADD	ADD.segment.module.output.percent
SUB	SUB.segment.module.output.percent
FLICKER	FLICKER.segment.module.output.pitch.speed.amount(1...15)
TIMED	TIMED.segment.module.output.timeAmount(6...240).timeunit.speed
PTIMED	TIMED.segment.module.output.timeAmount(6...240).timeunit.speed

BINARY_STATE	BINARY_STATE.segment.module.binarySensor(1...8)
RELAY	RELAY.segment.module.binary
RELAY_STATE	RELAY_STATE.segment.module.relay(1...8)
VAR_VALUE	VAR_VALUE.segment.module.datatype[.dataID(1...12)][.modifier]
SETPOINT_VALUE	SETPOINT_VALUE.segment.module.regulator.setpointAction[.operator] [.setpointValue(0...2999)][.modifier]
DIM_VALUE	DIM_percent.segment.module.output
LIMIT	LIMIT.segment.module.output.percent.value.timeunit
MEMORY	MEMORY.segment.module.output.ramp
RAMP_STOP	RAMP_STOP.segment.module.output
DIM_ALL	DIM_ALL.segment.module.percent.percent[.percent.percent.ramp]
ALL_BRIGHTNESS	ALL_BRIGHTNESS.segment.module.percent
ALL_ON	ALL_ON.segment.module.ramp
ALL_OFF	ALL_OFF.segment.module.ramp
ALL_TOGGLE	ALL_TOGGLE.segment.module.ramp
QUICKTIMER	QUICKTIMER.segment.module.output.percent[.ramp]
SHUTTER	SHUTTER.segment.module
DALI	DALI.segment.module.daliTarget[.percent].daliCommand[.percent]
DALI_RAW	DALI_RAW.segment.module.byteValue(0...255).byteValue(0...255)
LIGHT_SCENE	LIGHT_SCENE.segment.module.sceneAction.channel(0...7) .(scene[.ramp] | binaryCall)
CHOOSE_REGISTER	CHOOSE_REGISTER.segment.module.register
WRITE_SCENE	WRITE_SCENE.segment.module.scene.register.percent1.ramp1 .percent2.ramp2[.percent3.ramp3[.percent4.ramp4]]
READ_SCENE	READ_SCENE.segment.module.scene.register[.output[.type]]
RELAY_TIMER	RELAY_TIMER.segment.module.byteValue(0...255).binary
MOTOR	MOTOR.segment.module.motorNumber(1...7).motorAction [.value | .reportType]
SEND_KEYS	SEND_KEYS.segment.module.buttonAction.buttonAction.buttonAction [.buttonAction].binary
DELAY_KEYS	DELAY_KEYS.segment.module.table.value.advTimeUnit.binary
LOCK_KEYS	LOCK_KEYS.segment.module.table.advBinary
TIMELOCK_KEYS	TIMELOCK_KEYS.segment.module.value.advTimeUnit.simpleBinary
LED	LED.segment.module.ledNumber(1...12).ledAction
LED_STATE	LED_STATE.segment.module[.ledNumber(1...12)]
VAR_ADD	VAR_ADD.segment.module.dataID(1...12).bigValue[.modifier]
VAR_SUB	VAR_SUB.segment.module.dataID(1...12).bigValue[.modifier]
THRESHOLD_VALUE	THRESHOLD_VALUE.segment.module[.thresholdRegister] [.thresholdNumber][.modifier]
MOVE_THRESHOLD	MOVE_THRESHOLD.segment.module.origin.thresholdValue.operator .thresholdRegister.thresholdNumber[.modifier]
MOVE_THRESHOLD_OLD	MOVE_THRESHOLD_OLD.segment.module.origin.thresholdValue.operator .binary[.modifier]
GET_INFO	GET_INFO.segment.module.infoID

GET_OEM	GET_OEM.segment.module.oemID(1...4)
GROUPS	GROUPS.segment.module.operator.group
BEEP	BEEP.segment.module.beepType.amount(1...15)
TEXT	TEXT.segment.module.line(1..4).block(1...5).message
TIMED_TEXT	TIMED_TEXT.segment.module.line(1...4).ramp
MRS	MRS.segment.module.mrsCommand.<mrsCommand>Action
LANGUAGE	LANGUAGE.segmant.module.language
GET_COUPLER	GET_COUPLER
STATUS	STATUS.segment.module.statusCommand
SN	SN.segment.module
FW	FW.segment.module

3.2 LCN-Bezeichner

Die folgende Tabelle definiert alle Bezeichner, welche von den verschiedenen LCN-Kommandos benutzt werden, dabei kann ein Bezeichner wiederum aus weiteren Bezeichnern bestehen, welche ebenfalls aufgelöst werden müssen.

Der Name definiert den Namen, welcher auch in der "Syntax" Spalte der Tabelle "LCN-Kommandos" benutzt wird. In der Spalte "Range:" werden die möglichen Wertebereiche angegeben. Dies können entweder numerische Werte, Schlüsselwörter (welche in Großbuchstaben geschrieben sind), oder Bezeichner sein (erkennbar an den spitzen Klammern "<" und ">").

Werte und Bezeichner, welche durch ein "|" getrennt werden, zeigen Optionen auf. Nur einer der Werte kann benutzt werden. Der "pitch" Bezeichner kann nur durch LOW, MEDIUM, HIGH oder OFF ersetzt werden. Blauer Text in Klammern sind kurze Kommentare um einzelne Werte zu erklären.

Available Identifiers:	
Name:	**Range:**
segment	0, 3(=all), 5...127
module	5...254
output	1...4
percent	0...100
ramp	0...250
pitch	LOW | MEDIUM | HIGH | OFF
speed	SLOW | MEDIUM | FAST | QUICK
amount	1...15
timeAmount	6...240
timeunit	S (Sekunden) | M (Minuten)
binarySensor	1...8
relay	1...8
binary	0 (aus) | - (keine Änderung) | 1 (ein)
binaryCall	0 (nicht rufen) | 1 (rufen)

datatype	VAR | SETPOINT | COUNTER
dataID	1...12
modifier	CELSIUS | LUX | LUX_T | VOLT | AMP | WIND | MOISTURE | CO2 | NONE
regulator	REGULATOR1 | REGULATOR2
setpointAction	SET | PUSH_CURRENT | PUSH_PROG | ACTIVATE | DEACTIVATE
setpointValue	0...2999
daliTarget	SINGLE | GROUP | BROADCAST
daliCommand	LIGHT_SCENE | SET | OFF | UP | DOWN | STEP_UP | STEP_DOWN | MAX | MIN | DOWN_OFF | ON_UP
byteValue	0...255
value	0...999
sceneAction	LOAD | SAVE
scene	0...9
register	0...9
channel	0...7
motorNumber	1...7
motorAction	CLOSE | OPEN | FORCE_OPEN | STOP | LIMIT | GOTO | GOTO_HIGH | ADD | SUB | LEARN | REPORT1 | REPORT2
buttonAction	SHORT | LONG | RELEASE | -
advTimeUnit	S | M | H | D
table	A | B | C | D
advBinary	0 (aus) | 1 (ein) | - (keine Änderung) | U (umschalten)
simpleBinary	0 (aus) | 1 (ein)
ledNumber	1...12
ledAction	OFF | ON | BLINK | FLICKER
bigValue	0...30000
infoID	NAME1 | NAME2 | COMMENT1 | COMMENT2 | COMMENT3
oemID	1...4
message	*Bis zu 12 Zeichen*
operator	+ | -
beepType	NORMAL | SPECIAL
statusCommand	OUTPUTS | PPORT | RELAYS | BINARY | ALL
line	1...4
block	1...5
mrsCommand	CALL | VOLUME | SOURCE | RADIO | EQUALIZER | IMPRESSION | STANDBY
callAction	(START.<band>) | END
volumeAction	<mrsOutput>.(((UP | DOWN | ONGOING_UP | ONGOING_DOWN).<volume>) | RAMP_STOP | (SET.<percent>) | (SET_GROUP.<percent>) | MUTE)
sourceAction	<mrsOutput>.((SET.<source>) | PREVIOUS | NEXT)
radioAction	PLAY | STOP | PREVIOUS | NEXT | (SET.<byteValue>)
equalizerAction	<mrsOutput>.<band>.<db>
impressionAction	<mrsOutput>.(LOAD | SAVE).<impression>
standbyAction	(empty)

mrsOutput	1...12 | OPTICAL
impression	1...16
band	1...6
db	-15...15
source	1...6 | OPTICAL | WEBRADIO
volume	0...7
language	DE | EN | ES | FR | RU | AR | PL | TR
group	5...254
origin	PROG | CURRENT
thresholdNumber	0...4
thresholdRegister	0...4
thresholdValue	0...1000
type	RAMP | VALUE
reportType	POSITION | LIMIT | STEP_OUT | STEP_IN

4 LCN Kommandos

In diesem Kapitel werden sämtliche Kommandos noch einmal näher erläutert. Dabei wird nicht nur auf den Nutzen des Kommandos, sondern auch auf die genaue Definition des Kommandos eingegangen. Abgerundet wird die Erläuterung durch ein Beispiel Kommando, sowie gegebenenfalls Tipps zur Nutzung in openHAB.

ON	Das 'ON' Kommando wird benutzt um einen einzelnen Ausgang eines LCN Moduls einzuschalten. Es wird also meist benutzt um Lampen oder ähnliche Geräte zu bedienen.
Items	Switch, Dimmer
Syntax	**ON.segment.module.output[.ramp]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4. **ramp:** (optional) ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**ON.0.23.2**
	`Switch On_Test "Switch" {lcn="[ON:local:ON.0.23.2]"}`
	Dieses Kommando schaltet Ausgang 2, von Module 23, in Segment 0, ein.
Hinweise	

OFF	Das 'OFF' Kommando wird benutzt um einen einzelnen Ausgang eines LCN Moduls auszuschalten. Es wird also meist benutzt um Lampen oder ähnliche Geräte zu bedienen.
Items	Switch, Dimmer
Syntax	**OFF.segment.module.output[.ramp]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4.

	ramp: (optional) ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**OFF.15.23.2**
	`Switch Off_Test "Switch" {lcn="[OFF:local:OFF.15.23.2]"}`
	Dieses Kommando schaltet Ausgang 2, von Module 23, in Segment 15, aus.
Hinweise	

DIM	Das 'DIM' Kommando wird benutzt um einen Dimm-Wert eines einzelnen Ausgang eines LCN Moduls einzustellen.
Items	Switch
Syntax	**DIM.segment.module.output.percent[.ramp]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4. **percent:** ist der Dimm-Wert in Prozent, Maximum ist 100. **ramp:** (optional) ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**DIM.12.23.2.75.0**
	`Switch Dim_Test "Dimmer" {lcn="[ON:local:DIM.0.23.2.75.0]”}`
	Dieses Kommando dimmt den Ausgang 2, von Modul 23, in Segment 12, auf 75%.
Hinweise	

TOGGLE	Das 'TOGGLE' Kommando wird benutzt um einen einzelnen Ausgang eines LCN Moduls umzuschalten.
Items	Switch
Syntax	**TOGGLE.segment.module.output[.ramp]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0.

	module: ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4. **ramp:** (optional) ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**TOGGLE.17.89.1**
	`Switch Toggle_Test "Toggle" {lcn="[ON:local:TOGGLE.17.89.1]"}`
	Dieses Kommando schaltet den Ausgang 1, von Modul 89, in Segment 17, um.
Hinweise	

ADD	Das 'ADD' Kommando wird benutzt um den Dimm-Wert eines einzelnen Ausgangs, eines LCN Moduls zu erhöhen. Der Dimm-Wert kann nur erhöht werden, um den Wert zu senken benutzen Sie bitte "SUB".
Items	Switch, Dimmer
Syntax	**ADD.segment.module.output.percent**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4. **percent:** ist der Dimm-Wert in Prozent, Maximum ist 100.
Beispiel	**ADD.12.35.2.20**
	`Dimmer Add_Test "Dimmer" {lcn="[INCREASE:local:ADD.12.35.2.20]"}`
	Dieses Kommando erhöht den Dimm-Wert von Ausgang 2, von Modul 35, in Segment 12, um 20%.
Hinweise	

SUB	Das 'SUB' Kommando wird benutzt um den Dimm-Wert eines einzelnen Ausgangs, eines LCN Moduls zu verringern. Der Dimm-Wert kann nur verringert werden, um den Wert zu erhöhen benutzen Sie bitte "ADD".
Items	Switch, Dimmer
Syntax	**ADD.segment.module.output.percent**

	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4. **percent:** ist der Dimm-Wert in Prozent, Maximum ist 100.
Beispiel	**SUB.12.35.2.15**
	`Dimmer Sub_Test "Dimmer" {lcn="[DECREASE:local:SUB.12.35.2.15]"}`
	Dieser Befehl verringert den Dimm-Wert von Ausgang 2, von Modul 35, in Segment 12, um 15%.
Hinweise	

FLICKER	Das 'FLICKER' Kommando wird benutzt um einen einzelnen Ausgang eines LCN Moduls wiederholt schnell umzuschalten, um einen Flackereffekt hervorzurufen.
Items	Switch
Syntax	**FLICKER.segment.module.output.pitch.speed.amount**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4. **pitch:** ist der maximale Dimm-Wert welcher beim Flackern benutzt werden soll. Mögliche Werte sind: LOW MEDIUM HIGH und OFF. **speed:** ist die Geschwindigkeit des Flackerns. Mögliche Werte sind: SLOW MEDIUM und FAST. **amount:** gibt an wie oft geflackert werden soll, Maximum ist 15.
Beispiel	**FLICKER.45.13.3.HIGH.FAST.7**
	`Switch Flicker_Test "Flicker"` `{lcn="[ON:local:FLICKER.45.13.3.HIGH.FAST.7]"}`
	Dieser Befehl lässt den Ausgang 3, von Modul 13, in Segment 45, mit hoher Geschwindigkeit und hell, 7 mal flackern.
Hinweise	Dieser Befehl wird hauptsächlich für Lampen verwendet.

TIMED	Der 'TIMED' Befehl schaltet einen Ausgang eines LCN Moduls für eine festgelegte Zeit ein und danach wieder aus.
Items	Switch
Syntax	**TIMED.segment.module.output.timeAmount.timeunit.speed**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4. **timeAmount:** ist die Zeitspanne, nach der der Ausgang wieder ausgeschaltet werden soll. Werte liegen zwischen 6 und 240. **timeunit:** ist die Zeiteinheit, welche im Zusammenhang mit timeAmount verwendet wird. Ist entweder S (für Sekunden) oder M (für Minuten). **speed:** ist die Geschwindigkeit mit der die gesetzte Helligkeit beim Einschalten erreicht werden soll. Mögliche Werte sind: SLOW, MEDIUM und QUICK.
Beispiel	**TIMED.3.17.2.10.S.FAST**
	`Switch Timed_Test "Timed" {lcn="[ON:local:TIMED.3.17.2.10.S.FAST]"}`
	Dieser Befehl schaltet Ausgang 2, von Modul 17, in Segment 3, schnell auf ein und nach 10 Sekunden wieder aus.
Hinweise	

PTIMED	Der 'PTIMED' Befehl schaltet einen Ausgang eines LCN Moduls für eine festgelegte Zeit ein und danach wieder aus. Im Gegensatz zum 'TIMED' Befehlt, verlängert dieser Befehl noch laufende andere 'PTIMED' und 'TIMED' Befehle.
Items	Switch
Syntax	**PTIMED.segment.module.output.timeAmount.timeunit.speed**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4. **percent:** ist der Dimm-Wert in Prozent, Maximum ist 100. **timeAmount:** ist die Zeitspanne, nach der der Ausgang wieder ausgeschaltet werden soll. Werte liegen zwischen 6 und 240. **timeunit:** ist die Zeiteinheit, welche im Zusammenhang mit timeAmount

	verwendet wird. Ist entweder S (für Sekunden) oder M (für Minuten). **speed:** ist die Geschwindigkeit mit der die gesetzte Helligkeit beim Einschalten erreicht werden soll. Mögliche Werte sind: SLOW, MEDIUM und QUICK.
Beispiel	**PTIMED.3.17.2.10.S.FAST**
	`Switch PTimed_Test "PTimed" {lcn="[ON:local:PTIMED.3.17.2.10.S.FAST]"}`
	Dieser Befehl schaltet Ausgang 2, von Modul 17, in Segment 3, schnell auf ein und nach 10 Sekunden wieder aus. Falls der Ausgang bereits durch einen Zeit-Befehl eingeschaltet ist, wird die Dauer um 10 Sekunden verlängert.
Hinweise	

BINARY_STATE	Der 'BINARY_STATE' Befehl liefert den derzeitigen Status eines Binärsensors.
Items	String, Contact
Syntax	**BINARY_STATE.segment.module.binarySensor**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **binarySensor:** ist die ID des Binärsensors. Werte sind zwischen 1 und 8.
Beispiel	**BINARY_STATE.40.29.4**
	`String Binary_Test "Binary [%s] {lcn="[local:BINARY_STATE.40.29.4]"}`
	Dieser Befehl erhält den Status von Binärsensor 4, von Modul 29 , in Segment 40 und schreibt diesen als 'true' oder 'false' and die position von '%s'.
Hinweise	Aus offensichtlichen Gründen, kann ein Binärsensor nur ausgelesen, aber nicht geschrieben werden.

RELAY	Der 'RELAY' Befehl schaltet ein oder mehrere Relais eines LCN Moduls.
Items	Switch
Syntax	**RELAY.segment.module.binary**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0.

	module: ist die ID des Moduls selbst, Werte zwischen 5 und 127. **binary:** sind die binären Zustände der Relais. Dieser Parameter besteht aus 8 Zeichen, welche eine beliebige Kombination aus folgenden Zeichen sein kann: 1 (einschalten), 0 (ausschalten), - (nichts tun) oder U (umschalten).
Beispiel	**RELAY.0.20.---1----**
	`Switch Relay_Test "Relay" {lcn="[ON:local:RELAY.0.20.---1----]`
	Dieser Befehl schaltet das 4te Relais von Modul 20, in Segment 0, ein, während alle anderen Relais ihren derzeitigen Stand beibehalten.
Hinweise	

RELAY_ STATE	Der 'RELAY_STATE' Befehl liefert den Zustand eines einzelnen Relais.
Items	String
Syntax	**RELAY_STATE.segment.module.relay**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **relay:** ist die ID des Relais, zwischen 1 und 8.
Beispiel	**RELAY_STATE.0.20.4**
	`String Relay_Sensor "Relay Sensor [%s]"` `{lcn="[ON:local:RELAY_STATE.0.20.4]`
	Dieser Befehl liefert den Zustand des 4ten Relais, von Modul 20, in Segment 0, und schreibt ihn ('true' oder 'false') an die Position von '%s'.
Hinweise	

VAR_ VALUE	Der 'VAR_VALUE' Befehl liefert Daten von nicht binären Sensoren.
Items	String, Number
Syntax	**VAR_VALUE.segment.module.datatype[.dataID][.modifier]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127.

	datatype: ist der erwartete Datentyp. Mögliche Werte sind: *VAR, SETPOINT und COUNTER*. VAR: Zähl- / Rechen-Variable (dataID: 1 - 12) SETPOINT: Sollwert (dataID: 1 - 2) COUNTER: Zähl-Variable (dataID: 1 - 4) **[dataID]:** (optional) abhängig vom Datentyp. Beachten Sie, das ältere Module deutlich weniger Variablen verfügbar haben. Nur 1 für VAR, kein COUNTER und 2 für alle anderen. **[modifier]:** (optional) rechnet den erhaltenen Wert automatisch in die gewünschte Form um. Mögliche Werte sind: CELSIUS, LUX, LUX_T, VOLT, AMP, WIND, MOISTURE, CO2 und NONE. (Achtung: für LUX und LUX_T ist es nicht möglich Werte zu konvertieren, welche an den Bus geschickt werden sollen!)
Beispiel	**VAR_VALUE.0.8.VAR.2.CELSIUS**
	`String Temp_Sensor "Temperature Sensor [%s °C]" {lcn="[local:VAR_VALUE.0.8.VAR.2.CELSIUS]"}`
	Dieser Befehl liefert die Temperatur (in °C) von Sensor 2, von Modul 8, in Segment 0, und schreibt sie nach '%s'.
Hinweise	

SETPOINT_VALUE	Der 'SETPOINT_VALUE' Befehl ändert den Sollwert eines LCN Moduls.
Items	Switch, Dimmer
Syntax	**SETPOINT_VALUE.segment.module.regulator.setpointAction[.operator][.setpointValue][.modifier]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **regulator:** ist die ID des Reglers, entweder REGULATOR1 oder REGULATOR2 **setpointAction:** kann eine der folgenden Aktionen sein: *SET, PUSH_CURRENT, PUSH_PROG, ACTIVATE, DEACTIVATE*. SET: Setzt den Controller auf einen bestimmten Wert. PUSH_CURRENT: verschiebt den derzeitigen Wert des Reglers (benötigt einen operator und eine setpointValue). PUSH_PROG: verschiebt den programmierten Wert des Reglers (benötigt einen operator und eine setpointValue).

	ACTIVATE: aktiviert den Regler für weitere Befehle. DEACTIVATE: deaktiviert den Regler für weitere Befehle. **operator:** ist entweder + oder - **setpointValue:** ist der Wert für SET und PUSH Aktionen, Maximum ist 2000 für PUSH und 2999 für SET. **[modifier]:** (optional) rechnet den erhaltenen Wert automatisch in die gewünschte Form um. Mögliche Werte sind: CELSIUS, LUX, LUX_T, VOLT, AMP, WIND, MOISTURE, CO2 und NONE. (Achtung: für LUX und LUX_T ist es nicht möglich Werte zu konvertieren, welche an den Bus geschickt werden sollen!)
Beispiel	**SETPOINT_VALUE.0.8.REGULATOR1.PUSH_CURRENT.+.1**
	`Dimmer Setpoint_Mover {lcn="[INCREASE:local:SETPOINT_VALUE.0.8.REGULATOR1.PUSH_CURRENT.+.1], [DECREASE:local:SETPOINT_VALUE.0.8.REGULATOR1.PUSH_CURRENT.-.1]"}`
	Dieser Befehl erhöht den Sollwert um 1, während der 'Hoch' Knopf, und verringert den Sollwert um 1, während der 'Runter' Knopf gedrückt wird.
Hinweise	

DIM_VALUE	Der 'DIM_VALUE' Befehl liefert den genauen Dimm-Wert eines Ausgangs eines LCN Moduls.
Items	Number
Syntax	**DIM_VALUE.segment.module.output**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4.
Beispiel	**DIM_VALUE.12.23.2**
	`Number Dim_Value "Dim value [%d]" {lcn="[local:DIM_VALUE.12.23.2]"}`
	Dieser Befehl liefert den derzeitigen Wert von Ausgang 2, von Modul 23, in Segment 12, und schreib ihn nach '%d'.
Hinweise	

LIMIT	Der 'LIMIT' Befehl beschränkt einen Ausgang eines LCN Moduls, indem der

	Maximalwert des Ausgangs (temporär) reduziert wird.
Items	Switch
Syntax	**LIMIT.segment.module.output.percent.value.timeunit**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4. **percent:** das Limit zwischen 0 und 100 in Schritten von 4. Ungültige Werte werden automatisch angepasst. **value:** die Zeitdauer des Limits. Maximum ist abhängig von timeunit. **timeunit:** die Zeiteinheit, entweder S (1 - 60 Sekunden), M (1 – 90 Minuten), H (1 – 50 Stunden) oder D (1 – 45 Tage).
Beispiel	**LIMIT.0.23.2.25.10.S**
	`Switch Limit "Light Limit" {lcn="[ON:local:LIMIT.0.23.2.25.10.S]"}`
	Dieser Befehl limitiert Ausgang 2, von Modul 23, in Segment 0, auf ein Maximum von 24% für 10 Sekunden. Beachten Sie, wie der Wert automatisch angepasst wird!
Hinweise	

MEMORY	Der 'MEMORY' Befehl liest entweder einen Dimmwert ein oder gibt einen Dimmwert wieder aus. Dabei wird ein Wert eingelesen (und der Ausgang gleichzeitig auf 0 gestellt), wenn der Ausgang derzeit einen Dimmwert hat (ungleich 0). Hat der Ausgang den Wert 0, so wird der zuletzt gespeicherte Dimmwert angenommen.
Items	Switch
Syntax	**MEMORY.segment.module.output.ramp**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4. **ramp:** ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**MEMORY.0.23.2.0**

	`Switch Memory_Test "Memory" {lcn="[ON:local:MEMORY.0.23.2.0]"}`
	Dieser Befehl würde entweder den derzeitigen Dimmwert von Ausgang 2, Modul 23, in Segment 0, einlesen, oder ihn wieder ausgeben. Abhängig vom derzeitigen Status des Ausgangs.
Hinweise	

RAMP_STOP	Beendet die Rampe des Ausgang eines LCN Moduls frühzeitig.
Items	Switch
Syntax	**RAMP_STOP.segment.module.output**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4.
Beispiel	**RAMP_STOP.0.23.2**
	`Switch Stop_Test "Ramp Stopper" {lcn="[ON:local:RAMP_STOP.0.23.2]"}`
	Dieser Befehlt beendet die derzeitige Rampe des Ausgangs 2, von Modul 23, in Segment 0.
Hinweise	

DIM_ALL	Dimmt alle Ausgänge eines einzelnen LCN Moduls. Die Ausgänge 3 und 4 können erst ab einer Firmware von 160901 benutzt werden. Module ab Firmware 10061A bis Firmware 160901 haben bereits 4 Ausgänge, unterstützen aber noch nicht den 'DIM_ALL' Befehl mit 4 Ausgängen. Für diesen Fall benutzen Sie bitte den Befehl 'ALL_BRIGHTNESS'.
Items	Switch, Dimmer
Syntax	**DIM_ALL.segment.module.percent.percent2[.percent3.percent4.ramp]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **percent:** ist der Dimm-Wert für Ausgang 1, Maximum ist 100. **percent2:** ist der Dimm-Wert für Ausgang 2, Maximum ist 100.

	percent3: (optional) ist der Dimm-Wert für Ausgang 3, Maximum ist 100. **percent4:** (optional) ist der Dimm-Wert für Ausgang 4, Maximum ist 100. **ramp:** (optional) ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**DIM_ALL.0.23.100.0.100.0.0**
	`Switch Dim_Test "Dim All" {lcn="[ON:local:DIM_ALL.0.23.100.25.100.25.0]"}`
	Dieser Befehl spricht Modul 23, in Segment 0 an und schaltet sofort die Ausgänge 1 und 3 auf 100%, während die Ausgänge 2 und 4 auf 25% gedimmt werden.
Hinweise	Wenn einer der optionalen Parameter benutzt werden soll, so müssen sämtliche optionale Parameter benutzt werden.

ALL_BRIGHTNESS	Der 'ALL_BRIGHTNESS' Befehl dimmt alle Ausgänge eines LCN Moduls gleichzeitig.
Items	Switch, Dimmer
Syntax	**ALL_BRIGHTNESS.segment.module.percent**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **percent:** ist der Dimm-Wert in Prozent, Maximum ist 100.
Beispiel	**ALL_BRIGHTNESS.0.23.75**
	`Switch On_Test "Switch" {lcn="[ON:local:ALL_BRIGHTNESS.0.23.75]"}`
	Dieser Befehl dimmt alle Ausgänge von Modul 23, in Segment 0, auf 75%.
Hinweise	Der Unterschied zu 'DIM_ALL' besteht darin, dass 'ALL_BRIGHTNESS' nur einen Dimm-Wert für alle Ausgänge festlegen kann, während 'DIM_ALL' jedem Ausgang eigene Werte zuweisen kann.

ALL_ON	Der 'ALL_ON' Befehl schaltet alle Ausgänge eines LCN Moduls ein.
Items	Switch, Dimmer

Syntax	**ALL_ON.segment.module.ramp**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **ramp:** ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**ALL_ON.0.23.0**
	`Switch On_Test "Switch" {lcn="[ON:local:ALL_ON.0.23.0]"}`
	Dieser Befehl schaltet sofort alle Ausgänge des Moduls 23, in Segment 0, auf ein.
Hinweise	

ALL_OFF	Der 'ALL_OFF' Befehl schaltet alle Ausgänge eines LCN Moduls aus.
Items	Switch, Dimmer
Syntax	**ALL_OFF.segment.module.ramp**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **ramp:** ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**ALL_OFF.0.23.0**
	`Switch Off_Test "Switch" {lcn="[ON:local:ALL_OFF.0.23.0]"}`
	Dieser Befehl schaltet sofort alle Ausgänge des Moduls 23, in Segment 0, auf aus.
Hinweise	

ALL_TOGGLE	Der 'ALL_TOGGLE' Befehl schaltet alle Ausgänge eines LCN Moduls um.
Items	Switch, Dimmer
Syntax	**ALL_TOGGLE.segment.module.ramp**

	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **ramp:** ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**ALL_TOGGLE.0.23.0**
	`Switch On_Test "Switch" {lcn="[ON:local:ALL_TOGGLE.0.23.0]"}`
	Dieser Befehl schaltet sofort alle Ausgänge des Moduls 23, in Segment 0, um.
Hinweise	

QUICKTIMER	Der 'QUICKTIMER' Befehl stellt den Ausgang sofort auf den angegebenen Wert und schaltet ihn dann mit der Rampe auf 0.
Items	Switch
Syntax	**QUICKTIMER.segment.module.output.percent[.ramp]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **output:** ist die ID des Ausgangs, welcher angesprochen werden soll, der maximale Wert ist 4. Ältere Module können nur 2 Ausgänge ansprechen. **percent:** ist der Dimm-Wert in Prozent, Maximum ist 100. **ramp:** (optional) ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**QUICKTIMER.120.77.2.50.2**
	`Switch Quick "Quicktimer" {lcn="[local:QUICKTIMER.120.77.2.50.2]`
	Dieser Befehl stellt den Ausgang 2, von Modul 77, in Segment 120, sofort auf 50% und regelt ihn dann mit Rampe 2 (siehe Anhang) auf 0%.
Hinweise	

SHUTTER	Der 'SHUTTER' Befehl kontrolliert Rolladen.

Items	Switch
Syntax	**SHUTTER.segment.module**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127.
Beispiel	**SHUTTER.120.77**
	`String Shutter "Shutter [%s]" {lcn="[local:SHUTTER.120.77]`
	Dieser Befehl steuert das LCN Modul 77, in Segment 120, als Rolladen.
Hinweise	

DALI	Der 'DALI' Befehl ermöglicht die Nutzung von DALI-EVGs.
Items	Switch, Dimmer
Syntax	**DALI.segment.module.daliTarget[.percent].daliCommand[.percent]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **daliTarget:** das Ziel des Befehls, mögliche Werte sind: *SINGLE, GROUP, BROADCAST*. SINGLE: nur ein einzelnes Modul. GROUP: eine DALI Gruppe. BROADCAST: alle Module. **percent:** (optional) bei SINGLE oder GROUP muss das Ziel mit einer Adresse spezifiziert werden. Für SINGLE liegt diese zwischen 0 und 63 und für GROUP zwischen 0 und 15. **daliCommand:** das eigentliche Kommando, mögliche Werte sind: *LIGHT_SCENE, SET, OFF, UP, DOWN, STEP_UP, STEP_DOWN, MAX, MIN, DOWN_OFF, ON_UP*. LIGHT_SCENE: ruft eine bestimme Lichtszene. SET: setzt die Helligkeit. OFF: schaltet aus. UP: erhöht die Helligkeit. DOWN: verringert die Helligkeit. STEP_UP: erhöht die Helligkeit um einen Schritt. STEP_DOWN: verringert die Helligkeit um einen Schritt.

	MAX: ruft den maximal Wert. MIN: ruft den minimal Wert. DOWN_OFF: verringert die Helligkeit um einen Schritt und schaltet dann aus. ON_UP: Schaltet ein und erhöht die Helligkeit um einen Schritt. **percent:** (optional) bei LIGHT_SCENE oder SET, muss das Ziel (bzw. der Wert) spezifiziert werden. Für LIGHT_SCENE kann der Wert zwischen 0 und 15 und für SET zwischen 0 und 254 (wobei 254, 100% entspricht).
Beispiel 1	**DALI.0.23.BROADCAST.OFF**
	`Switch Dali "Dali" {lcn="[ON:local:DALI.0.23.BROADCAST.OFF]"}`
	Dieser Befehl schaltet alle Module an Modul 23, in Segment 0 aus.
Beispiel 2	DALI.0.23.SINGLE.45.LIGHT_SCENE.12
	`Switch Dali "Dali" {lcn="[ON:local:DALI.0.23.SINGLE.45.LIGHT_SCENE.12]"}`
	Dieser Befehl ruft die Lichtszene 12 für DALI-EVG 45, an LCN Modul 23, in Segment 0.
Hinweise	

DALI_RAW	Der 'DALI_RAW' Befehl ermöglicht direkte Befehle (in Byte Form) an DALI-EVGs.
Items	Switch, Dimmer
Syntax	**DALI_RAW.segment.module.byteValue.byteValue**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **byteValue:** beliebiger Byte-Wert (0 – 255).
Beispiel	**DALI_RAW.0.23.10.20**
	`Switch Dali_Raw "Dali Raw" {lcn="[ON:local:DALI_RAW.0.23.10.20]"}`
	Dieser Befehl sendet die zwei Byte-Werte 10 und 20 an Modul 23 in Segment 0.
Hinweise	Dieser Befehl sollte nur in Zusammenhang mit ausreichenden Kenntnissen über DALI Befehle verwendet werden.

LIGHT_SCENE	Der 'LIGHT_SCENE' Befehl liest (bzw. schreibt) eine Lichtszene.

	Entweder über Kanäle oder Relais.
Items	Switch
Syntax	**LIGHT_SCENE.segment.module.sceneAction.channel** **.(scene[.ramp] | binaryCall)**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **sceneAction:** ist entweder *LOAD* oder *SAVE*. LOAD: lädt eine Lichtszene. SAVE: speichert den derzeitigen Wert in eine Lichtszene. **channel:** der Ausgang für den Kanal. Werte können zwischen 0 und 7 sein, und haben folgende Bedeutung: 0 = benutzt Relais Informationen. (benötigt binaryCall Parameter). 1 = Ausgang 1 2 = Ausgang 2 3 = Ausgang 1 und 2 4 = Ausgang 3 5 = Ausgang 1 und 3 6 = Ausgang 2 und 3 7 = alle Ausgänge **scene:** die ID der Szene, Werte zwischen 0 und 9 **ramp:** (optional) (nicht kompatibel mit Kanal 0) ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255. **binaryCall:** definiert welche Relais benutzt werden sollen. Dieser Parameter besteht aus 8 Zeichen, welche aus folgenden Zeichen bestehen: 1 (rufen), 0 (nicht rufen).
Beispiel 1	**LIGHT_SCENE.0.23.LOAD.2.4.0**
	`Switch Load_Scene "Load Light Scene: " {lcn="[ON:local:LIGHT_SCENE.0.23.LOAD.2.4.0]"}`
	Dieser Befehl lädt sofort Lichtszene Nummer 4, von Kanal 2, an Modul 23, in Segment 0.
Beispiel 2	LIGHT_SCENE.0.23.LOAD.0.2.11110000
	`Switch Load_Scene "Load Light Scene: " {lcn="[ON:local:LIGHT_SCENE.0.23.LOAD.0.2.11110000]"}`
	Dieser Befehl lädt Lichtszene Nummer 2, mit den Relais 1 bis 4, an

	Modul 23, in Segment 0.
Hinweise	

CHOOSE_ REGISTER	Der 'CHOOSE_REGISTER' Befehl wechselt das Register, mit welchem Lichtszenen geladen und gespeichert werden.
Items	Switch
Syntax	**CHOOSE_REGISTER.segment.module.register**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **register:** ist die ID des Registers, Maximalwert ist 9.
Beispiel	**CHOOSE_REGISTER.0.23.0**
	`Switch Register "Choose Register" {lcn="[ON:local:CHOOSE_REGISTER.0.23.0]"}`
	Dieser Befehl wählt Register 0, für Modul 23, in Segment 0.
Hinweise	Benutzen Sie diesen Befehl in Kombination mit 'LIGHT_SCENE' um die Kapazität an Speicherplätzen zu erhöhen.

WRITE_ SCENE	Der 'WRITE_SCENE' Befehl schreibt eine Lichtszene direkt. Er ist also unabhängig vom derzeitigen Wert.
Items	Switch
Syntax	**WRITE_SCENE.segment.module.scene.register.percent1.ramp1.perse nt2.ramp2[.percent3.ramp3[.percent4.ramp4]]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **scene:** ist die ID der Szene, Werte zwischen 0 und 9 **register:** ist die ID des Registers, Maximalwert ist 9. **percent(i):** ist der Dimm-Wert in halben Prozent, Maximum ist 200. **ramp(i):** ist die Zeit welche das Modul brauchen soll um zu dem Dimm-Wert zu gelangen. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**WRITE_SCENE.0.23.2.1.100.0.50.0**

	`Switch Write_Scene "Direct Write Scene" {lcn="[ON:local:WRITE_SCENE.0.23.2.1.100.0.50.0]"}`
	Dieser Befehl schreibt die Lichtszene in Register 1, Szene 2, von Modul 23, in Segment 0. Die Lichtszene wird sofort den Ausgang 1 auf 100% und den Ausgang 2 auf 50% schalten.
Hinweise	

READ_SCENE	Der 'READ_SCENE' Befehl liefert eine bestimmte Lichtszene.
Items	String
Syntax	**READ_SCENE.segment.module.scene.register[.output[.type]]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **scene:** ist die ID der Szene, Werte zwischen 0 und 9 **register:** ist die ID des Registers, Maximalwert ist 9. **output:** (optional) ist die ID des Ausgangs, Maximum ist 4. **type:** (optional) entweder RAMP oder VALUE.
Beispiel	**READ_SCENE.0.23.2.1**
	`String Read_Scene "Direct Read Scene: [%s]" {lcn="[local:READ_SCENE.0.23.2.1]"}`
	Dieser Befehl liest die Lichtszene in Register 1, Szene 2, von Modul 23, in Segment 0, und schreibt den Inhalt an nach '%s'.
Hinweise	

RELAY_TIMER	Der 'RELAY_TIMER' Befehl schaltet ein Relais sofort auf ein und nach einer festgelegten Zeit wieder aus.
Items	Switch
Syntax	**RELAY_TIMER.segment.module.byteValue.binary**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0.

	module: ist die ID des Moduls selbst, Werte zwischen 5 und 127. **byteValue:** beliebiger Byte-Wert (1 – 255). 1 = 0.03s, 255 = 240.96s. Eine genaue Auflisten befindet sich im Anhang. **binary:** sind die binären Zustände der Relais. Dieser Parameter besteht aus 8 Zeichen, welche eine beliebige Kombination aus folgenden Zeichen sein kann: 1 (einschalten), 0 (ausschalten) oder - (nichts tun).
Beispiel	**RELAY_TIMER.0.23.1.----1---**
	`Switch Relay_Timer "Timed Relay" {lcn="[ON:local:RELAY_TIMER.0.23.1.----1---]"}`
	Dieser Befehl schaltet Relais 5, von Modul 23, in Segment 0 ein und nach 0,03 Sekunden wieder aus.
Hinweise	

MOTOR	Der 'MOTOR' Befehl steuert einen Motor an einem LCN Modul.
Items	Switch, Dimmer
Syntax	**MOTOR.segment.module.motorNumber.motorAction[.value | .reportType]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **motorNumber:** die ID des Motors, zwischen 1 und 7. Nicht benötigt für die motorActions REPORT1 und REPORT2. **motorAction:** mögliche Werte sind: *CLOSE, OPEN, FORCE_OPEN, STOP, LIMIT, GOTO, GOTO_HIGH, ADD, SUB, LEARN, REPORT, REPORT_LONG1* und *REPORT_LONG2.* CLOSE: schließt (z.B. ein Fenster). OPEN: öffnet (z.B. ein Fenster). FORCE_OPEN: erzwingt das Öffnen (z.B. von einem Fenster). STOP: stoppt den Motor. LIMIT: setzt einen Maximalwert. GOTO: fährt zu einer bestimmten Position. GOTO_HIGH: fährt zu einer bestimmten Position, mit erhöhter Genauigkeit. ADD: erhöht die derzeitigen Position. SUB: verringert die derzeitige Position. LEARN: macht eine Lehrfahrt. REPORT1: erweiterter Bericht von Motor 1 und 2. (benötigt keine

	motorNumber). REPORT2: erweiterter Bericht von Motor 3 und 4. (benötigt keine motorNumber). **value:** (optional) LIMIT, und GOTO benötigen einen Wert zwischen 0 und 100 (in %), während GOTO_HIGH, ADD und SUB einen Wert zwischen 0 und 200 (in 0.5%) benötigen. **reportType:** (optional) für REPORT1 und REPORT2 können folgende Typen abgefragt werden. Falls kein Typ angegeben wird, so werden alle Informationen in einer einzelnen Nachricht zurückgegeben. POSITION, LIMIT, STEP_IN und STEP_OUT. POSITION: die derzeitige Position des Motors. LIMIT: das Limit des Motors. STEP_IN: der step IN des Motors. STEP_OUT: der step OUT des Motors.
Beispiel	**MOTOR.0.23.1.STOP**
	`Switch Motor "Motor Stop" {lcn="[ON:local:MOTOR.0.23.1.STOP]"}`
	Dieser Befehlt stoppt den ersten Motor, von Modul 23, in Segment 0.
Hinweise	

SEND_ BUTTON	Der 'SEND_KEYS' Befehl sendet ein Kommando an einen oder mehrere Knöpfe von einer oder mehreren Tabellen.
Items	Switch
Syntax	**SEND_KEYS.segment.module.buttonAction.buttonAction .buttonAction[.buttonAction].binary**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **buttonAction:** Die 3 (bzw. 4) buttonActions repräsentieren die verschiedenen Tabellen A bis C (bzw. D). Mögliche Werte sind: *SHORT, LONG, RELEASE und -*. SHORT: Sendet einen 'kurz' Befehl an die Tabelle. LONG: Sendet einen 'lang' Befehl an die Tabelle. RELEASE: Sendet einen 'loslassen' Befehl an die Tabelle. - : Sendet keinen Befehl an die Tabelle. **binary:** sind die binären Zustände. Dieser Parameter besteht aus 8 Zeichen, welche eine beliebige Kombination der folgenden Zeichen sein

	kann: 1 (Knopf drücken), 0 (nichts tun).
Beispiel	**SEND_KEYS.0.23.SHORT.-.-.-.1000000**
	`Switch SendButton "Send Button" {lcn="[ON:local:SEND_KEYS.0.23.SHORT.-.-.-.10000000]"}`
	Dieser Befehl sendet einen 'kurz' Befehl an den ersten Knopf der Tabelle A.
Hinweise	

DELAY_KEYS	Der 'DELAY_KEYS' Befehl sendet einen verzögerten 'kurz' Befehl an einen oder mehrere Knöpfe einer Tabelle. Sendet immer einen 'kurz' Befehl.
Items	Switch
Syntax	**DELAY_KEYS.segment.module.table.value.advTimeUnit.binary**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **table:** die Tabelle, mögliche Werte sind: A, B, C und D. **value:** die Zeitdauer des Limits. Maximum ist abhängig von timeunit. **advTimeUnit:** die Zeiteinheit, entweder S (1 - 60 Sekunden), M (1 – 90 Minuten), H (1 – 50 Stunden) oder D (1 – 45 Tage). **binary:** sind die binären Zustände. Dieser Parameter besteht aus 8 Zeichen, welche eine beliebige Kombination der folgenden Zeichen sein kann: 1 (Knopf drücken), 0 (nichts tun).
Beispiel	**DELAY_KEYS.0.23.B.10.S.00100000**
	`Switch DelayButton "Delay Button" {lcn="[ON:local:DELAY_KEYS.0.23.B.10.S.00100000]"}`
	Nach 10 Sekunden sendet dieser Befehl einen 'kurz' Befehl an den 3ten Knopf der Tabelle B.
Hinweise	

LOCK_KEYS	Der 'LOCK_KEYS' Befehl (ent-)sperrt einen oder mehrere Knöpfe einer Tabelle.
Items	Switch
Syntax	**LOCK_KEYS.segment.module.table.advBinary**

	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **table:** die Tabelle, mögliche Werte sind: A, B, C und D. **advBinary:** sind die binären Zustände. Dieser Parameter besteht aus 8 Zeichen, welche eine beliebige Kombination der folgenden Zeichen sein kann: 1 (sperren), 0 (entsperren) oder U (umschalten).
Beispiel	**LOCK_KEYS.0.23.C.11111111**
	`Switch LockButton "Lock Button" {lcn="[ON:local:LOCK_KEYS.0.23.C.11111111]"}`
	Dieser Befehl sperrt alle Knöpfe von Tabelle C.
Hinweise	

TIMELOCK_KEYS	Der 'TIMELOCK_KEYS' Befehl (ent-)sperrt einen oder mehrere Knöpfe von Tabelle A. Dieser Befehl benutzt immer Tabelle A!
Items	Switch
Syntax	**TIMELOCK_KEYS.segment.module.value.advTimeUnit .simpleBinary**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **value:** die Zeitdauer des Limits. Maximum ist abhängig von timeunit. **advTimeUnit:** die Zeiteinheit, entweder S (1 - 60 Sekunden), M (1 – 90 Minuten), H (1 – 50 Stunden) oder D (1 – 45 Tage). **binary:** sind die binären Zustände. Dieser Parameter besteht aus 8 Zeichen, welche eine beliebige Kombination der folgenden Zeichen sein kann: 1 (sperren), 0 (nichts tun).
Beispiel	**TIMELOCK_KEYS.0.23.10.M.10000000**
	`Switch TimelockButton "Timelock Button" {lcn="[ON:local:TIMELOCK_KEYS.0.23.10.M.10000000]"}`
	Dieser Befehl sperrt den ersten Knopf von Tabelle A für 10 Minuten.
Hinweise	

LED	Der 'LED' Befehl kontrolliert LEDs, schaltet sie ein und aus, etc.
Items	Switch
Syntax	**LED.segment.module.ledNumber.ledAction**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **ledNumber:** die ID der LED, Werte zwischen 1 und 12. **ledAction:** mögliche Werte sind: *OFF, ON, BLINK u*nd *FLICKER.* OFF: schaltet die LED aus. ON: schaltet die LED ein. BLINK: lässt die LED blinken. FLICKER: lässt die LED flackern.
Beispiel	**LED.0.23.2.ON**
	`Switch Led "LED" {lcn="[local:LED.0.23.2.ON]"}`
	Schaltet die 2te LED von Module 23, in Segment 0, ein.
Hinweise	

LED_ STATE	Der 'LED_STATE' Befehl liefert den derzeitigen Status der LEDs eines LCN Moduls.
Items	String
Syntax	**LED_STATE.segment.module[.ledNumber]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **ledNumber:** (optional) die ID der LED, Werte zwischen 1 und 12.
Beispiel	**LED_STATE.0.23**
	`String LED_STATE "LED: [%s]" {lcn="[local:LED_STATE.0.23]"}`
	Liest den Status der LEDs von Module 23, in Segment 0, und schreibt ihn nach '%s'.
Hinweise	

VAR_ADD	Der 'VAR_ADD' Befehl erhöht eine Variable eines LCN Moduls.
Items	Switch, Dimmer
Syntax	**VAR_ADD.segment.module.dataID.bigValue[.modifier]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **dataID:** ist die ID der Variable, maximal 12. Beachten Sie, dass ältere Module deutlich weniger Variablen haben können. **bigValue:** Wert zwischen 0 und 30000 **[modifier]:** (optional) rechnet den erhaltenen Wert automatisch in die gewünschte Form um. Mögliche Werte sind: CELSIUS, LUX, LUX_T, VOLT, AMP, WIND, MOISTURE, CO2 und NONE. (Achtung: für LUX und LUX_T ist es nicht möglich Werte zu konvertieren, welche an den Bus geschickt werden sollen!)
Beispiel	**VAR_ADD.0.23.1.2500**
	`Switch VAR_ADD "Var Add" {lcn="[ON:local:VAR_ADD.0.23.1.2500]"}`
	Dieser Befehl erhöht die erste Variable des Moduls 23, in Segment 0, um 2500.
Hinweise	

VAR_SUB	Der 'VAR_SUB' Befehl verringert eine Variable eines LCN Moduls.
Items	Switch, Dimmer
Syntax	**VAR_SUB.segment.module.dataID.bigValue[.modifier]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **dataID:** ist die ID der Variable, maximal 12. Beachten Sie, dass ältere Module deutlich weniger Variablen haben können. **bigValue:** Wert zwischen 0 und 30000 **[modifier]:** (optional) rechnet den erhaltenen Wert automatisch in die gewünschte Form um. Mögliche Werte sind: CELSIUS, LUX, LUX_T, VOLT, AMP, WIND, MOISTURE, CO2 und NONE. (Achtung: für LUX und LUX_T ist es nicht möglich Werte zu konvertieren, welche an den Bus geschickt werden sollen!)
Beispiel	**VAR_SUB.0.23.1.720**

	`Switch VAR_SUB "Var Sub" {lcn="[ON:local:VAR_SUB.0.23.1.720]"}`
	Dieser Befehl verringert die erste Variable des Moduls 23, in Segment 0, um 720.
Hinweise	

THRESHOLD_VALUE	Der 'THRESHOLD_VALUE' Befehl liefert eine Liste aller Schwellwerte eine LCN Moduls.
Items	String
Syntax	**THRESHOLD_VALUE.segment.module.register.thresholdNumber [.modifier]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **register:** nur für Module von 2012 oder später. Werte zwischen 1 und 4. **thresholdNumber:** Werte zwischen 1 und 4 (5 für Module vor 2012). **[modifier]:** (optional) rechnet den erhaltenen Wert automatisch in die gewünschte Form um. Mögliche Werte sind: CELSIUS, LUX, LUX_T, VOLT, AMP, WIND, MOISTURE, CO2 und NONE. (Achtung: für LUX und LUX_T ist es nicht möglich Werte zu konvertieren, welche an den Bus geschickt werden sollen!)
Beispiel	**THRESHOLD_VALUE.0.23.1.1**
	`String THRESHOLD_VALUE "Threshold Value: [%s]" {lcn="[local:THRESHOLD_VALUE.0.23.1.1]"}`
	Dieser Befehl liefert den Schwellwerte Nummer 1 aus Register 1 von Modul 23, in Segment 0, und schreibt ihn nach '%s'.
Hinweise	

MOVE_THRESHOLD	Der 'MOVE_THRESHOLD' Befehl verschiebt einen einzelnen Schwellwert eines LCN Moduls. Dieser Befehl ist nur für Module mit Firmware 160B13 oder neuer geeignet.
items	String
syntax	**MOVE_THRESHOLD.segment.module.origin.thresholdValue .operator.thresholdRegister.thresholdNumber[.modifier]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet,

	zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **origin:** ist entweder PROG oder CURRENT. PROG: verschiebt den Schwellwert relativ zum programmierten Wert. CURRENT: verschiebt den Schwellwert relativ zum aktuellen Wert. **thresholdValue:** ist der Wert um den der Schwellwert verschoben werden soll. Werte zwischen 1 und 1000 sind zulässig. **operator:** ist entweder + oder - **thresholdRegister:** das Register des Schwellwerts, maximal 4. **thresholdNumber:** die ID des Schwellwerts, maximal 4. **[modifier]:** (optional) rechnet den erhaltenen Wert automatisch in die gewünschte Form um. Mögliche Werte sind: CELSIUS, LUX, LUX_T, VOLT, AMP, WIND, MOISTURE, CO2 und NONE. (Achtung: für LUX und LUX_T ist es nicht möglich Werte zu konvertieren, welche an den Bus geschickt werden sollen!)
example	**MOVE_THRESHOLD.0.23.CURRENT.100.-.2.2**
	`Switch Move_Threshold "Move Threshold" {lcn="[local:MOVE_THRESHOLD.0.23.CURRENT.100.-.2.2]"}`
	Dieser Befehl verringert den Schwellwert 2, in Register 2, von Modul 23, in Segment 0, in Relation zum derzeitigen Schwellwert, um 100.
notes	

MOVE_THRESHOLD_OLD	Der 'MOVE_THRESHOLD_OLD' Befehl verschiebt weinen oder mehrere Schwellwerte eines LCN Moduls. Dieser Befehl ist nur für Module mit Firmware 140C0D oder älter geeignet.
items	String
syntax	**MOVE_THRESHOLD_OLD.segment.module.origin .thresholdValue.operator.binary[.modifier]**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **origin:** ist entweder PROG oder CURRENT. PROG: verschiebt den Schwellwert relativ zum programmierten Wert. CURRENT: verschiebt den Schwellwert relativ zum aktuellen Wert. **thresholdValue:** ist der Wert um den der Schwellwert verschoben werden soll. Werte zwischen 1 und 1000 sind zulässig.

	operator: ist entweder + oder - **binary:** markiert welche Schwellwerte verschoben werden sollen. Dieser Parameter besteht aus 5 Zeichen und kann sich aus folgenden zusammensetzten: 1 (verschieben), - (nicht verschieben), 0 (nicht verschieben). **[modifier]:** (optional) rechnet den erhaltenen Wert automatisch in die gewünschte Form um. Mögliche Werte sind: CELSIUS, LUX, LUX_T, VOLT, AMP, WIND, MOISTURE, CO2 und NONE. (Achtung: für LUX und LUX_T ist es nicht möglich Werte zu konvertieren, welche an den Bus geschickt werden sollen!)
example	**MOVE_THRESHOLD_OLD.0.23.PROG.100.+.01-10**
	`Switch Move_Threshold_Old "Move Threshold (old)" {lcn="[local:MOVE_THRESHOLD_OLD.0.23.PROG.100.+.01-10]"}`
	Dieser Befehl erhöht die Schwellwerte 2 und 4, von Modul 23, in Segment 0, in Relation zum programmierten Schwellwert, um 100.
notes	

GET_INFO	Der 'GET_INFO' Befehl liefert den Namen und/oder die Kommentare eines LCN Moduls.
Items	String
Syntax	**GET_INFO.segment.module.infoID**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **infoID:** mögliche Werte sind: *NAME1, NAME2, COMMENT1, COMMENT2 u*nd *COMMENT3.* NAME(i): liest Teil (i) des Modulnamens. COMMENT(i): liest Teil (i) der Kommentare des Moduls.
Beispiel	**GET_INFO.0.23.NAME1**
	`String Get_Info "Get Info [%s]" {lcn="[local:GET_INFO.0.23.NAME1]"}`
	Dieser Befehl liefert den ersten Part des Modulnamens von Modul 23, in Segment 0, und schreibt ihn nach '%s'.
Hinweise	

GET_OEM	Der 'GET_OEM' Befehl liefert die OEM Informationen eines LCN Moduls.
Items	String
Syntax	**GET_OEM.segment.module.oemID**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **oemID:** die ID des OEM Textes, Werte zwischen 1 und 4.
Beispiel	**GET_OEM.0.23.1**
	`String Get_OEM "Get OEM: [%s]" {lcn="[local:GET_OEM.0.23.1]"}`
	Dieser Befehl liest den ersten Teil der OEM Informationen von Module 23, in Segment 0, und schreibt ihn nach '%s'.
Hinweise	

GROUPS	Der 'GROUPS' Befehl fügt LCN Module einer dynamischen Gruppe hinzu oder entfernt sie wieder.
Items	Switch
Syntax	**GROUPS.segment.module.operator.group**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **operator:** ist entweder + oder - **group:** die ID der Gruppe. Zwischen 5 und 254.
Beispiel 1	**GROUPS.0.23.+.10**
	`Switch Groups "Add To Group" {lcn="[ON:local:GROUPS.0.23.+.10]"}`
	Dieser Befehl fügt Modul 23, aus Segment 0, der Gruppe 10 hinzu.
Beispiel 2	GROUPS.0.23.-.0
	`Switch Del_Groups "Delete Groups" {lcn="[ON:local:GROUPS.0.23.-.0]"}`
	Dies ist ein Sonderkommando und löscht alle Gruppen!
Hinweise	

BEEP	Der 'BEEP' Befehl sendet ein Piep-Signal an ein LCN Modul.
Items	Switch
Syntax	**BEEP.segment.module.beepType.amount**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **beepType:** mögliche Werte sind SPECIAL und NORMAL. **amount:** die Anzahl an Pieps, Maximum ist 15.
Beispiel	**BEEP.0.23.NORMAL.5**
	`Switch Beeper "Beep" {lcn="[ON:local:BEEP.0.23.NORMAL.5]"}`
	Dieser Befehl sendet 5 normale Pieps an Modul 23, in Segment 0.
Hinweise	

TEXT	Der 'TEXT' Befehl setzt den Text eines Displaymoduls.
Items	Switch
Syntax	**TEXT.segment.module.line.block.message**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **line:** ist die ID der Textzeile, zwischen 1 und 4. **block:** ist die ID des Textblocks, zwischen 1 und 5. **message:** bis zu 12 beliebige Zeichen.
Beispiel	**TEXT.0.23.2.4.welcome**
	`Switch Txt "Text" {lcn="[ON:local:TEXT.0.23.2.4.welcome]"}`
	Dieser Befehl setzt den Text der Textzeile 2, von Textblock 4, von Modul 23, in Segment 0, auf "welcome".
Hinweise	Manche Zeichen, wie z.B. die Umlaute, benötigen zwei oder sogar mehr Zeichenplätze. Überschüssige Zeichen werden automatische entfernt.

TIMED_TEXT	Der 'TIMED_TEXT' Befehl stellt die Dauer einer Textzeile eines Displaymoduls ein.

Items	Switch
Syntax	**TEXT_TIMED.segment.module.line.ramp**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **line:** ist die ID der Textzeile, zwischen 1 und 4. **ramp:** ist die Anzeigedauer der Textzeile. Eine Liste aus möglichen Werten und deren Bedeutung befindet sich im Anhang. Maximum ist 255.
Beispiel	**TIMED_TEXT.0.23.2.9**
	`Switch Timed_Txt "Timed Text" {lcn="[ON:local:TIMED_TEXT.0.23.2.9]"}`
	Dieser Befehl setzt die Anzeigedauer der Textzeile 2, von Modul 23, in Segment 0, auf 5 Sekunden.
Hinweise	

MRS	Der 'MRS' Befehl
Items	Switch
Syntax	**MRS.segment.module.mrsCommand.<mrsCommand>Action**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **mrsCommand:** mögliche Werte sind: *CALL, VOLUME, SOURCE, RADIO, EQUALIZER, IMPRESSION* und *STANDBY.* CALL: Pflichtruf Aktionen START.<band>: startet ein Band. Werte für <band> sind 1 – 16. END: beendet alle. VOLUME.<mrsOutput>: Lautstärke Aktionen. Werte für <mrsOutput> sind 1 – 12 oder OPTICAL. UP.<volume>: erhöht die Lautstärke um einen Schritt. Werte für <volume> sind 0 – 7. DOWN.<volume>: verringert die Lautstärke um einen Schritt. Werte für <value> sind 0 – 7. ONGOING_UP.<volume>: erhöht fortlaufend die Lautstärke. Werte für <volume> sind 0 – 7. ONGOING_DOWN.<volume>: verringert fortlaufend die Lautstärke. Werte für <volume> sind 0 – 7.

	RAMP_STOP: stoppt die Rampe für ONGOING_UP oder ONGOING_DOWN. SET.<percent>: setzt die Lautstärke auf <percent>. Werte für <percent> sind 0 – 100. SET_GROUP.<percent>: Setzt die Lautstärke für die gesamte Syncgruppe auf <percent>. Values for <percent> are 0 – 100. MUTE: schaltet den lautlos Modus um. SOURCE.<mrsOutput>: Quellen Aktionen. Werte für <mrsOutput> sind 1 – 12 oder OPTICAL. SET.<source>: setzt die Quelle. Werte für <source> sind: 1 – 6, OPTICAL oder WEBRADIO. PREVIOUS: wählt die vorherige Quelle. NEXT: wählt die nächste Quelle. RADIO: Webradio Aktionen. PLAY: spielt das Webradio. STOP: stoppt das Webradio. PREVIOUS: wählt der vorherigen Radiokanal. NEXT: wählt den nächsten Radiokanal. SET.<byteValue>: wählt einen Kanal. Werte für <byteValue> sind 0 – 255. EQUALIZER.<mrsOutput>.<band>.<db>: Equalizer Aktionen. <mrsOutput>: Setzt den Ausgang, mögliche Werte sind 1 – 12 oder OPTICAL. <band>: setzt das Band, mögliche Werte sind 1 – 6. <db>: setzt die Lautstärke, mögliche Werte sind -15 bis 15. IMPRESSION.<mrsOutput>: Klangbild Aktionen. Werte für <mrsOutput> sind 1 – 12 oder OPTICAL. LOAD.<impression>: lädt ein Klangbild. Werte für <impression> sind 1 – 16. SAVE.<impression>: speichert ein Klangbild. Werte für <impression> sind 1 – 16. STANDBY: Aktiviert den Standby Modus. <mrsCommand>Action: abhängig vom <mrsCommand> muss eine geeignete Aktion ausgewählt werden. Verfügbare Aktionen stehen im jeweiligen Eintrag.
Beispiel	**MRS.0.23.VOLUME.7.SET.70**
	`Switch Mrs "MRS" {lcn="[ON:local:MRS.0.23.VOLUME.7.SET.70]"}`
	Dieser Befehl setzt die Lautstärke des MRS Ausgangs 7, an Modul 23, in Segment 0, auf 70%.

Hinweise	

LANGUAGE	Der 'LANGUAGE' Befehl stellt die Sprache für den I-Port (z.B. MRS, GT4D, GT10D) ein.
Items	Switch
Syntax	**LANGUAGE.segment.module.language**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **language:** ist die Sprache. Mögliche Werte sind: *DE, EN, ES, FR, RU, AR, PL,* und *TR*. DE: Deutsch EN: Englisch ES: Spanisch FR: Französisch RU: Russisch AR: Arabisch PL: Polnisch TR: Türkisch
Beispiel	**LANGUAGE.0.23.EN**
	`Switch Lang "Language" {lcn="[ON:local:LANGUAGE.0.23.EN]"}`
	Dieser Befehl setzt die Sprache am I-Port von Modul 23, in Segment 0, auf Englisch.
Hinweise	

Die folgenden Befehle werden hauptsächlich intern benutzt und haben wenig Bedeutung für den Benutzer. Der Vollständigkeit halber, werden diese Befehle hier trotzdem aufgelistet:

GET_ COUPLER	Der 'GET_COUPLER' Befehl fordert Informationen über Segmentkoppler an.
Items	Switch
Syntax	**GET_COUPLER**
	keine Parameter benötigt
Beispiel	**GET_COUPLER**
	`Switch Get_Coupler "Get Coupler" {lcn="[ON:local:GET_COUPLER]"}`
	Dieser Befehl fordert Informationen über die Segmentkoppler an.
Hinweise	

STATUS	Der 'STATUS' Befehl fordert Statusberichte eines LCN Moduls an.
Items	Switch
Syntax	**STATUS.segment.module.statusCommand**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127. **statusCommand:** mögliche Werte sind: OUTPUT*S, PPORT, RELAYS, BINARY* und *ALL.* OUTPUTS: fordert Berichte der Ausgänge an. PPORT: fordert Berichte der P-Ports an. RELAYS: fordert Berichte der Relais. BINARY: fordert Berichte der Binärsensoren. ALL: fordert Berichte aller Bauteile.
Beispiel	**STATUS.0.23.ALL**
	`Switch Status "Status" {lcn="[ON:local:STATUS.0.23.ALL]"}`
	Dieser Befehl fordert Berichte aller Bauteile von Modul 23, in Segment 0 an.
Hinweise	

SN	Der 'SN' Befehl liefert die Seriennummer des LCN Moduls.
Items	String
Syntax	**SN.segment.module**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127.
Beispiel	**SN.140.77**
	`String Serial_Test "Serial [%s]" {lcn="[local:SN.140.77]`
	Dieser Befehl liefert die Seriennummer von Modul 77, in Segment 140, und schreibt sie nach '%s'.
Hinweise	

FW	Der 'FW' Befehl liefert die Firmware Version des LCN Moduls.
Items	String
Syntax	**FW.segment.module**
	segment: ist die ID des Segments, wo sich das LCN Modul befindet, zulässige Werte sind zwischen 5 und 127, sowie 0. **module:** ist die ID des Moduls selbst, Werte zwischen 5 und 127.
Beispiel	**FW.140.77**
	`String FW_Test "Firmware [%s]" {lcn="[local:FW.140.77]`
	Dieser Befehl liefert die Firmware Version von Modul 77, in Segment 140, und schreibt sie nach '%s'.
Hinweise	

5 Anhang

Im Anhang finden Sie einige nützliche Materialien und Informationen, welche Ihnen mit bestimmten Aspekten des LCN-Bindings helfen werden. Dazu gehören je eine Tabelle für Rampen und Relais, welche die Bedeutung der verschiedenen möglichen Werte aufzeigen.

5.1 Zeiten für Rampen

Diese Tabelle zeigt die verschiedenen möglichen Werte und deren Bedeutung für Rampen. Bitte beachten Sie, dass die Zeit Werte nicht linear verlaufen.

Ramp	Seconds		Ramp	Seconds
0	0.00		6	2.00
1	0.25		7	3.00
2	0.50		8	4.00
3	0.66		9	5.00
4	1.00		10...250	2 * (ramp - 10) + 6
5	1.40			

5.2 Zeiten für Relais

Diese Tabelle zeigt die verschiedenen möglichen Werte und deren Bedeutung für das RELAY_TIMER Kommando. Genauso wie schon in der Tabelle für Rampen, sind die Werte weder linear noch proportional angeordnet.

	0,03	32	0,96 s	64	2,88 s	96	6,72 s	128	14,4 s	160	29,76 s	192	60,48 s	224	121,92 s
1	0,03 s	33	1,02 s	65	3 s	97	6,96 s	129	14,88 s	161	30,72 s	193	62,4 s	225	125,76 s
2	0,06 s	34	1,08 s	66	3,12 s	98	7,2 s	130	15,36 s	162	31,68 s	194	64,32 s	226	129,6 s
3	0,09 s	35	1,14 s	67	3,24 s	99	7,44 s	131	15,84 s	163	32,64 s	195	66,24 s	227	133,44 s
4	0,12 s	36	1,2 s	68	3,36 s	100	7,68 s	132	16,32 s	164	33,6 s	196	68,16 s	228	137,28 s
5	0,15 s	37	1,26 s	69	3,48 s	101	7,92 s	133	16,8 s	165	34,56 s	197	70,08 s	229	141,12 s
6	0,18 s	38	1,32 s	70	3,6 s	102	8,16 s	134	17,28 s	166	35,52 s	198	72 s	230	144,96 s
7	0,21 s	39	1,38 s	71	3,72 s	103	8,4 s	135	17,76 s	167	36,48 s	199	73,92 s	231	148,8 s
8	0,24 s	40	1,44 s	72	3,84 s	104	8,64 s	136	18,24 s	168	37,44 s	200	75,84 s	232	152,64 s
9	0,27 s	41	1,5 s	73	3,96 s	105	8,88 s	137	18,72 s	169	38,4 s	201	77,76 s	233	156,48 s
10	0,3 s	42	1,56 s	74	4,08 s	106	9,12 s	138	19,2 s	170	39,36 s	202	79,68 s	234	160,32 s
11	0,33 s	43	1,62 s	75	4,2 s	107	9,36 s	139	19,68 s	171	40,32 s	203	81,6 s	235	164,16 s
12	0,36 s	44	1,68 s	76	4,32 s	108	9,6 s	140	20,16 s	172	41,28 s	204	83,52 s	236	168 s
13	0,39 s	45	1,74 s	77	4,44 s	109	9,84 s	141	20,64 s	173	42,24 s	205	85,44 s	237	171,84 s
14	0,42 s	46	1,8 s	78	4,56 s	110	10,08 s	142	21,12 s	174	43,2 s	206	87,36 s	238	175,68 s
15	0,45 s	47	1,86 s	79	4,68 s	111	10,32 s	143	21,6 s	175	44,16 s	207	89,28 s	239	179,52 s
16	0,48 s	48	1,92 s	80	4,8 s	112	10,56 s	144	22,08 s	176	45,12 s	208	91,2 s	240	183,36 s
17	0,51 s	49	1,98 s	81	4,92 s	113	10,8 s	145	22,56 s	177	46,08 s	209	93,12 s	241	187,2 s
18	0,54 s	50	2,04 s	82	5,04 s	114	11,04 s	146	23,04 s	178	47,04 s	210	95,04 s	242	191,04 s
19	0,57 s	51	2,1 s	83	5,16 s	115	11,28 s	147	23,52 s	179	48 s	211	96,96 s	243	194,88 s
20	0,6 s	52	2,16 s	84	5,28 s	116	11,52 s	148	24 s	180	48,96 s	212	98,88 s	244	198,72 s
21	0,63 s	53	2,22 s	85	5,4 s	117	11,76 s	149	24,48 s	181	49,92 s	213	100,8 s	245	202,56 s
22	0,66 s	54	2,28 s	86	5,52 s	118	12 s	150	24,96 s	182	50,88 s	214	102,72 s	246	206,4 s
23	0,69 s	55	2,34 s	87	5,64 s	119	12,24 s	151	25,44 s	183	51,84 s	215	104,64 s	247	210,24 s
24	0,72 s	56	2,4 s	88	5,76 s	120	12,48 s	152	25,92 s	184	52,8 s	216	106,56 s	248	214,08 s
25	0,75 s	57	2,46 s	89	5,88 s	121	12,72 s	153	26,4 s	185	53,76 s	217	108,48 s	249	217,92 s
26	0,78 s	58	2,52 s	90	6 s	122	12,96 s	154	26,88 s	186	54,72 s	218	110,4 s	250	221,76 s
27	0,81 s	59	2,58 s	91	6,12 s	123	13,2 s	155	27,36 s	187	55,68 s	219	112,32 s	251	225,6 s
28	0,84 s	60	2,64 s	92	6,24 s	124	13,44 s	156	27,84 s	188	56,64 s	220	114,24 s	252	229,44 s
29	0,87 s	61	2,7 s	93	6,36 s	125	13,68 s	157	28,32 s	189	57,6 s	221	116,16 s	253	233,28 s
30	0,9 s	62	2,76 s	94	6,48 s	126	13,92 s	158	28,8 s	190	58,56 s	222	118,08 s	254	237,12 s
31	0,93 s	63	2,82 s	95	6,6 s	127	14,16 s	159	29,28 s	191	59,52 s	223	120 s	255	240,96 s