

CUx-Daemon

CUx-Daemon VELUX KLF200 Dokumentation

Version 2.11



Inhaltsverzeichnis

1	Velux/Somfy io-homecontrol Anbindung über KLF200.....	3
1.1	(CUX1900xxx) 1. Szenentaster.....	5
1.2	(CUX1901xxx) 2. Rolladen.....	8
1.3	(CUX1902xxx) 3. Fensteröffner.....	13
1.4	(CUX1903xxx) 4. Schalter.....	17
1.5	(CUX1904xxx) 5. Dimmer.....	18
2	KLF200 Befehlsfolgen.....	19
3	KLF200 Verbindungsprobleme und Restart.....	21

1 Velux/Somfy io-homecontrol Anbindung über KLF200

Der CUxD ist eine universelle Schnittstelle zwischen der CCU-Logikschicht (ReGa HSS) und externen Geräten. Als Ergänzung zur ausführlichen CUxD-Dokumentation geht es im folgenden nur um die Anbindung eines [KLF200 Gateways](#) mit Firmware 2.0.0.71 per LAN-Schnittstelle und TLS verschlüsselter TCP/IP Netzwerk-Verbindung über die von Velux offiziell veröffentlichte API an den CUxD.

An das KLF200 Gateway können bis zu **200** Geräte/Antriebe ohne weitere Hardware über das verschlüsselte **io-homecontrol** Funkprotokoll direkt und ohne Cloud Zwang registriert/gekoppelt werden. Grundsätzlich ist neben Velux und Somfy Antrieben mit **io-homecontrol** Unterstützung auch die Kopplung aller **io-homecontrol** kompatiblen Geräte (Rolläden, Markisen, Fenster- und Toröffner, Vorhänge usw.) möglich.

Das Koppeln kann entweder mittels einer Zweiwege-Bedienung (z.B. Velux KLR200) oder mittels Einweg-Bedienung (z.B. Velux KLI31x) erfolgen. Die genaue Vorgehensweise ist in der [KLF200-Dokumentation](#) beschrieben. Über die per WLAN Access Point erreichbare WebUI des KLF200 Gateways können alle gekoppelten Geräte mit eindeutigen Namen versehen, sowie zusätzliche Szenen (Programme) konfiguriert werden. Eine Aktivierung der konfigurierten Szenen und die direkte Steuerung aller zuvor gekoppelten Geräte mit bidirektionaler Rückmeldung des Gerätezustandes sind danach mittels CUxD möglich.

Die Nutzungslizenz besteht aus einem **KEY** der für die Freischaltung dieser Funktionalität pro CCU separat [erworben](#) werden muss und dann im CUxD-Setup als Parameter hinzugefügt wird.

CUxD-Einstellungen:
zum Aktivieren speichern (siehe Status-Menü)

```

DEVLOGMOVE-HR=
DEVLOGEXPORT=
DEVTIMEFORMAT=%Y-%m-%dT%X
DEVDATAFORMAT=
DEVLOGDIALIAS=0
SUBSCRIBE-RF=1
SUBSCRIBE-WR=1
AUTOSAVE=1
MOUNTCMD=
UMOUNTCMD=
BACKUPCMD=
SYSLOGMOVE=
SYSLOGMOVEDAILY=
TH-DIR=
TH-DIR-FILTER=
ADDRESS-BUFFER=120

;Velux KLF200
STARTUPCMD=extra/pty2tcp start ttySSL0 -ssl klf200:51200 verify=0
TTYADD=ttySSL0
TTYASSIGN=ttySSL0:KLF200:XXXXXXXXXXXX
KEY=KL:D_XBfx43R2nU+?Y?
  
```

Speichern Parameterabgleich

Firmware-Update:
Gerät durch Drücken des Tasters bzw. Terminalbefehl
in Update-Modus versetzen (siehe Dokumentation)

Gerät suchen

Kein DFU Gerät im Updatemodus gefunden!

KEY=KL:

Zu ergänzende Zeilen Im CUxD-Setup:

```

STARTUPCMD=extra/pty2tcp start ttySSL0 -ssl <klf200_IP>:51200 verify=0
TTYADD=ttySSL0
TTYASSIGN=ttySSL0:KLF200:<klf200-wlan-password>
KEY=KL: . . .
  
```

Per Default und nach einem Factory-Reset handelt es sich laut [Dokumentation](#) bei dem `<klf200-wlan-passwort>` zur Authentifikation um das auf der Rückseite der Box gedruckte *Wi-Fi Passwort*. In Einzelfällen kann es aber auch das *Passwort der Web-Anmeldung* sein. Also ggf. ist beides zu testen.

TTY-Status nach Verbindung zum virtuellen **ttySSL0**, noch **keine Verbindung** zum KLF200 Gateway:

```
TTY - {KLF200} (0000) [COMM] - /dev/ttySSL0 [R] {:5s} - connected - ...
```

nach erfolgreichem **Verbindungsaufbau** und **fehlerhafter Authentifikation** am KLF200 Gateway:

```
TTY - {KLF200} (0000) [COMM] - /dev/ttySSL0 [R] {:54s} - authentication failed - ...
```

nach erfolgreichem **Verbindungsaufbau** und erfolgreicher **Authentifikation** am KLF200 Gateway:

```
TTY - {KLF200} (0000) [COMM] - /dev/ttySSL0 [R] {:13s} - FW(2.0.0.71), HW(6) ...
```

beim Betrieb mit aktiviertem **Home Status Monitor HSM** (Default) und **gültigem LizenzKEY**:

```
TTY - {KLF200} (0000) [COMM] - /dev/ttySSL0 [R] {:11s} - FW(2.0.0.71), HW(6), HSM - KEY - CONF -
```

Sind io-homecontrol Geräte an das KLF200 Gateway angelernet, dann werden beim Klick auf **CONF** auf einer Info-Seite alle vom KLF200 ausgelesenen Nodes, Szenen und Gruppen mit weiteren Informationen (ID, Type+Subtype [Seite 26 u. 104 API], Seriennummer, Name, Szenenkonfiguration) angezeigt. Diese Daten werden auf der CCU zum Download zusätzlich auch in der Datei `/var/cache/cuxd_klf200_ttySSL0.txt` abgelegt.

Aktuelle Status Information

SERVICE
ADDR
Open

TTY: ttySSL0, updated: 19.06.2023 13:58:15

nodeID	atst	atype	stype	pgrp	ptype	nvar	serial	name
1	0080	2	0	1	5	0	53xxxxxxxxxxxxxx	Dach Rolladen1
7	0080	2	0	1	5	0	56xxxxxxxxxxxxxx	Dach Rolladen2
3	0080	2	0	1	5	0	53xxxxxxxxxxxxxx	Dach Rolladen3
6	0080	2	0	1	5	0	53xxxxxxxxxxxxxx	Dach Rolladen4
5	0101	4	1	3	7	0	53xxxxxxxxxxxxxx	Kind Fenster
4	0080	2	0	1	5	0	56xxxxxxxxxxxxxx	Kind Rolladen
0	0101	4	1	3	1	0	56xxxxxxxxxxxxxx	Flur Fenster
2	0080	2	0	1	1	0	53xxxxxxxxxxxxxx	Flur Rolladen

sceneID	name
3	Dachrollos auf
node[3].MP = 0000	
node[1].MP = 0000	
4	Dachrollos zu
node[1].MP = C800	
node[3].MP = C800	

groupID	type	name	>> nodeIDs
50	0:USER	Group 0	>> 1 3 4
51	0:USER	Group 1	>> 0

Bei jeder Initialisierung der Verbindung zum konfigurierten KLF200 Interface, werden diese Daten im CUxD aktualisiert. Sie bilden die Grundlage für die Auswahllisten bei der Gerätekonfiguration.

Das KLF200 Gateway muss im **Schnittstellenmodus** konfiguriert und das LAN-Interface in den Power-Optionen auf „**always on**“ gesetzt sein. Ist das KLF200 Gateway als **Repeater** konfiguriert, dann funktioniert diese Schnittstelle nicht!

Es sind alle mit dem KLF200 gekoppelten io-homecontrol Geräte über dieses Interface steuerbar. Das schießt auch Geräte ein, für die in diesem AddOn noch keine Steuerelemente implementiert wurden. Dafür haben alle KLF200 CUxD-Geräte spezielle Experten-Datenpunkte um mit entsprechendem Wissen aus der veröffentlichten API Dokumentation [technical specification for klf 200 api-ver3-18.pdf](#) die Parameter der Schnittstelle zu setzen.

1.1 (CUX1900xxx) 1. Szenentaster

Mit diesem virtuellen Gerät können auf dem KLF200 konfigurierte Szenen (Programme) über einen virtuellen Taster direkt aktiviert werden.


CUxD Gerätetyp: (19) KLF200 Geräte ▼

Gerät: 1. Szenentaster ▼

Seriennummer: (numerisch max. 3 Stellen)

Name: (leer = wird autom. generiert)

Geräte-Icon:
 ▼



Gerät auf CCU erzeugen !

Konfigurationsparameter:

Parameter
CHANNELS <input style="width: 100px;" type="text" value="4"/> (1-16)

CHANNELS - Anzahl der konfigurierten Kanäle zum logischen Zusammenfassen mehrerer Szenen in einem CCU Gerät.

Kanal	Parameter
Ch.: 1	<div>KEY SCENE <input style="width: 150px;" type="text" value="4: Dachrollos zu"/> ▼</div> <div>KEY SPEED <input style="width: 100px;" type="text" value="DEFAULT"/> ▼</div> <div>KEY PRIORITY <input style="width: 150px;" type="text" value="5: Comfort Level 2"/> ▼</div>

SCENE - Auswahl einer auf dem KLF200 zuvor konfigurierten Szene aus Liste

SPEED - Geschwindigkeit (Default, Silent, Fast)

PRIORITY - Priorität (Seite 59 API)

0: Human Protection

1: Environment Protection

2: User Level 1

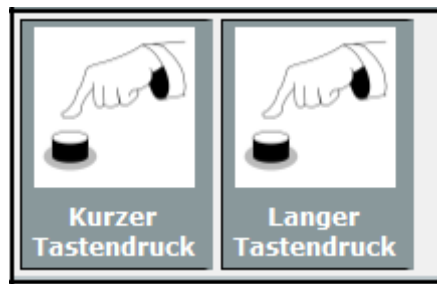
3: User Level 2

4: Comfort Level 1

5: Comfort Level 2 (Default)

6: Comfort Level 3

7: Comfort Level 4



Kanaltypen:

Kanaltyp	Kanalnummer
KEY	1..16

Kanaltyp KEY:

DP-Name	Typ	Zugriff	Beschreibung
PRESS_SHORT	action	schreibend	konfigurierte Szene aktivieren
PRESS_LONG	action	schreibend	konfigurierte Szene stoppen
WORKING	boolean	lesend	TRUE während die Szene abgearbeitet wird
STATUS	integer	lesend	StatusReply nach Ausführung des Befehls

StatusReply Codes (Seite 63 API):

Stat	TAG	Beschreibung
0	UNKNOWN_STATUS_REPLY	Used to indicate unknown reply.
1	COMMAND_COMPLETED_OK	Indicates no errors detected.
2	NO_CONTACT	Indicates no communication to node.
3	MANUALLY_OPERATED	Indicates manually operated by a user.
4	BLOCKED	Indicates node has been blocked by an object.
5	WRONG_SYSTEMKEY	Indicates the node contains a wrong system key.
6	PRIORITY_LEVEL_LOCKED	Indicates the node is locked on this priority level.
7	REACHED_WRONG_POSITION	Indicates node has stopped in another position than expected.
8	ERROR_DURING_EXECUTION	Indicates an error has occurred during execution of command.
9	NO_EXECUTION	Indicates no movement of the node parameter.
10	CALIBRATING	Indicates the node is calibrating the parameters.
11	POWER_CONSUMPTION_TOO_HIGH	Indicates the node power consumption is too high.
12	POWER_CONSUMPTION_TOO_LOW	Indicates the node power consumption is too low.
13	LOCK_POSITION_OPEN	Indicates door lock errors. (Door open during lock command)
14	MOTION_TIME_TOO_LONG_COMMUNICATION_ENDED	Indicates the target was not reached in time.
15	THERMAL_PROTECTION	Indicates the node has gone into thermal protection mode.
16	PRODUCT_NOT_OPERATIONAL	Indicates the node is not currently operational.
17	FILTER_MAINTENANCE_NEEDED	Indicates the filter needs maintenance.
18	BATTERY_LEVEL	Indicates the battery level is low.
19	TARGET_MODIFIED	Indicates the node has modified the target value of the command.
20	MODE_NOT_IMPLEMENTED	Indicates this node does not support the mode received.
21	COMMAND_INCOMPATIBLE_TO_MOVEMENT	Indicates the node is unable to move in the right direction.
22	USER_ACTION	Indicates dead bolt is manually locked during unlock command.
23	DEAD_BOLT_ERROR	Indicates dead bolt error.
24	AUTOMATIC_CYCLE_ENGAGED	Indicates the node has gone into automatic cycle mode.
25	WRONG_LOAD_CONNECTED	Indicates wrong load on node.
26	COLOUR_NOT_REACHABLE	Indicates that node is unable to reach received colour code.
27	TARGET_NOT_REACHABLE	Indicates the node is unable to reach received target position.
28	BAD_INDEX_RECEIVED	Indicates io-protocol has received an invalid index.
29	COMMAND_OVERRULED	Indicates that the command was overruled by a new command.
30	NODE_WAITING_FOR_POWER	Indicates that the node reported waiting for power.
223	INFORMATION_CODE	Indicates an unknown error code received. (Hex code is shown on display)
224	PARAMETER_LIMITED	Indicates the parameter was limited by an unknown device. (Same as LIMITATION_BY_UNKNOWN_DEVICE)
225	LIMITATION_BY_LOCAL_USER	Indicates the parameter was limited by local button.
226	LIMITATION_BY_USER	Indicates the parameter was limited by a remote control.
227	LIMITATION_BY_RAIN	Indicates the parameter was limited by a rain sensor.
228	LIMITATION_BY_TIMER	Indicates the parameter was limited by a timer.
230	LIMITATION_BY_UPS	Indicates the parameter was limited by a power supply.
231	LIMITATION_BY_UNKNOWN_DEVICE	Indicates the parameter was limited by an unknown device. (Same as PARAMETER_LIMITED)
234	LIMITATION_BY_SAAC	Indicates the parameter was limited by a standalone automatic controller.
235	LIMITATION_BY_WIND	Indicates the parameter was limited by a wind sensor.
236	LIMITATION_BY_MYSELF	Indicates the parameter was limited by the node itself.
237	LIMITATION_BY_AUTOMATIC_CYCLE	Indicates the parameter was limited by an automatic cycle.
238	LIMITATION_BY_EMERGENCY	Indicates the parameter was limited by an emergency.

1.2 (CUX1901xxx) 2. Rolladen

Mit diesem virtuellen Gerät können mit dem KLF200 gekoppelte Rolläden über die CCU gesteuert werden.

CUxD Gerätetyp: (19) KLF200 Geräte ▼

Gerät: 2. Rolladen ▼

Seriennummer: 1 (numerisch max. 3 Stellen)

Name: (leer = wird autom. generiert)

Geräte-Icon:
Jalousieaktor 1fach Unterputz ▼

Gerät auf CCU erzeugen !

Konfigurationsparameter:

Parameter	
NODE	2: [080] Treppe Rolladen ▼
SPEED	DEFAULT ▼
PRIORITY	5: Comfort Level 2 ▼
PLI	33333333,30 PLI,LockTime
Zyklische Statusmeldung <input checked="" type="checkbox"/>	

NODE - Auswahl aus Liste aller mit dem KLF200 gekoppelten Aktoren. Die hexadezimale Zahl in der eckigen Klammer beschreibt zusätzlich den Type+SubType des Gerätes, Seite 104 API). Der Status dieses Aktors wird für Rückmeldungen ausgewertet.

SET_NODES - Beschreibung s. **SET_NODES** Datenpunkt (optional!)

SPEED - Bewegungsgeschwindigkeit (Default, Silent, Medium, Fast)

PRIORITY - Priorität (Seite 59 API)

- 0: Human Protection
- 1: Environment Protection
- 2: User Level 1
- 3: User Level 2
- 4: Comfort Level 1
- 5: Comfort Level 2 (Default)**
- 6: Comfort Level 3
- 7: Comfort Level 4

PLI - PrioritylevelLockInformation,LockTime
(Default = 33333333,xxx = Deaktiviert)
Für jeden der 8 **Prioritätslevel** von 0 (links) bis 7 (rechts) können 4 Werte definiert werden. (Seite 61 API)

PLI	Name	Beschreibung
0	Disable priority	Disable the priority related to the Master
1	Enable	Enable the priority related to the Master
2	Enable all	Enable all pool entry for the specified priority level Must be used with caution!
3	Keep current	Do not make any action. When used, the priority setting for the specific level will be kept in its current state.

Als **LockTime** sind Zeiten von **30** bis **7650** Sekunden im Abstand von 30s und **u** für Unbegrenzt erlaubt.

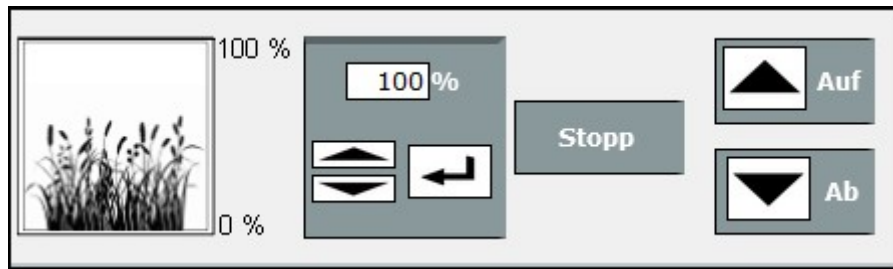
CYCLIC_INFO_MSG - Prüfen, ob sich das Gerät mindestens ein Mal pro Stunde meldet. Ansonsten wird eine Servicemeldung erzeugt.

WORKING_LEVEL_UPDATE - Aktualisierung von LEVEL-Änderungen während der Bewegung des Antriebs.

Kanal	Parameter
Ch.: 1	WINMATIC INVERT <input type="checkbox"/> WINMATIC SPEED_PARAMETER 1 (0-4)

INVERT - [x] Status dieses Gerätes invertieren

SPEED_PARAMETER - (Default = 1) ID des Funktionsparameters für die Geschwindigkeit. Nicht ändern!



Kanaltypen:

Kanaltyp	Kanalnummer
BLIND	1

Kanaltyp BLIND:

DP-Name	Typ	Zugriff	Beschreibung
LEVEL	float	lesend schreibend	Rolladen öffnen, schließen bzw. anfahren einer bestimmten Position
STOP	action	schreibend	Rolladenbewegung anhalten
WORKING	boolean	lesend	kennzeichnet Aktivität (Bewegung)
DIRECTION	enum	lesend	Rolladenbewegung (0..keine, 1..hoch, 2..runter). Wird eine Bewegung erkannt, dann erfolgt alle 5s ein LEVEL -Update.
ON_TIME	integer	schreibend	nach Ablauf der angegebenen Sekunden wird nach einer LEVEL -Änderung die <u>letzte Position</u> oder ein zuvor gesetzter ON_TIME_LEVEL wieder hergestellt. (z.B. als Beschattungszeit anwendbar). Ist gerade ein Timer aktiv, dann wird die Zeit entsprechend verlängert und die gespeicherte <u>letzte Position</u> bleibt unverändert. Eine manuelle Bedienung während dieser Zeit deaktiviert den Timer.
ON_TIME_LEVEL	float	schreibend	LEVEL nach Ablauf von ON_TIME überschreiben
SPEED	integer	lesend schreibend	Bewegungsgeschwindigkeit linear setzen. 0 =Default, 1 =Minimal ... 51200 =Maximal


Allgemeine Kanaldatenpunkte:

DP-Name	Typ	Zugriff	Beschreibung
STATUS	integer	lesend	StatusReply nach Ausführung eines Sendebefehls
MY_STATUS_SET	integer	lesend schreibend	MY_STATUS Wert für Triggerung nach nächstem Sendebefehl durch eine Programmverknüpfung setzen.
MY_STATUS	integer	lesend	Ausgabe von MY_STATUS_SET , wenn Ausführung des letzten Sendebefehls erfolgreich war, ansonsten -MY_STATUS_SET
SET_LIMIT	string	schreibend	<i>ParameterID,LimitMin,LimitMax,LimitTime</i> Min- und Max-Limits im Aktor für die definierte Zeit (<i>LimitTime</i>) setzen (Seite 75 API). Wenn <i>LimitMax</i> gesetzt ist, muss auch <i>LimitMin</i> gesetzt bzw. vorher mittels GET_LIMIT abgefragt werden! Als <i>LimitTime</i> sind Zeiten von 30 bis 7590 Sekunden, u für „Unbegrenzt“, m für „clear Master“ und a für „clear All“ erlaubt
GET_LIMIT	string	schreibend	<i>ParameterID,MinMax</i> Limit Abfrage starten. Ergebnis im xx_MIN/MAX DP ParameterID: 0 =MP, 1 =FP1, 2 =FP2, 3 =FP3, 4 =FP4 MinMax: 0 =Min (Default), 1 =Max, 2 =Min & Max
GET_STATE	boolean	schreibend	Statusabfrage per Funk mit Update von LEVEL / MP / FP bei Änderung (FALSE) bzw. immer (TRUE)
WINK	integer	schreibend	<i>WinkTime</i> (Seite 74 API) Gerät zum Identifizieren für <i>WinkTime</i> (1..253) Sekunden oder (255) unendlich aktivieren. Mit (0) STOP kann der Prozess vorzeitig abgebrochen werden.
Expertenparameter			
PRIORITY	integer	lesend schreibend	Priorität (s. Geräteparameter) temporär für den nächsten Sendebefehl setzen
PLI	string	lesend schreibend	Steuerung der Prioritätslevel (s. Geräteparameter) temporär für den nächsten Sendebefehl setzen
SET_NODES	string	lesend, schreibend	<Node1>,<Node2>,... oder G <GroupID> Liste mit bis zu 20 NodeIDs oder eine GroupID (nur <i>LEVEL</i> , <i>STOP</i> , <i>STATE</i>) temporär für den nächsten Sendebefehl setzen. Siehe CONF auf der CUxD-Statusseite für die verfügbaren Node- und Group-IDs.
SET_PARAMETER	string	schreibend	<i>MP,FP1,FP2,...,FP16</i> Alle angegebenen Geräteparameter an dieses bzw. alle zuvor mittels SET_NODES definierten NodeIDs senden
GET_PARAMETER	string	lesend	aktuelle Geräteparameter per Funk von diesem Gerät abrufen <i>MP,FP1,FP2,...,FP16</i>
MODE	string	schreibend	<i>ModeNumber,ModeParameter</i> (Seite 80 API)
REBOOT	action	schreibend	KLF200 neu starten

DP-Name	Typ	Zugriff	Beschreibung
RAW Rückgabewerte (0..51200)			
MP	integer	lesend	Hauptparameter (MP) (Seite 106 API)
MP_MIN	integer	lesend	Hauptparameter Min Limit
MP_MAX	integer	lesend	Hauptparameter Max Limit
FP1..4	integer	lesend	FunktionsParameter1..4 (Seite 104 API)
FP1..4_MIN	integer	lesend	FunktionsParameter1..4 Min Limit
FP1..4_MAX	integer	lesend	FunktionsParameter1..4 Max Limit

1.3 (CUX1902xxx) 3. Fensteröffner

Mit diesem virtuellen Gerät können mit dem KLF200 gekoppelte Fensteröffner über die CCU gesteuert werden.

CUxD Gerätetyp:	(19) KLF200 Geräte
Gerät:	3. Fensteröffner
Seriennummer:	1 (numerisch max. 3 Stellen)
Name:	(leer = wird autom. generiert)
Geräte-Icon:	WinMatic
	
<input type="button" value="Gerät auf CCU erzeugen !"/>	

Konfigurationsparameter:

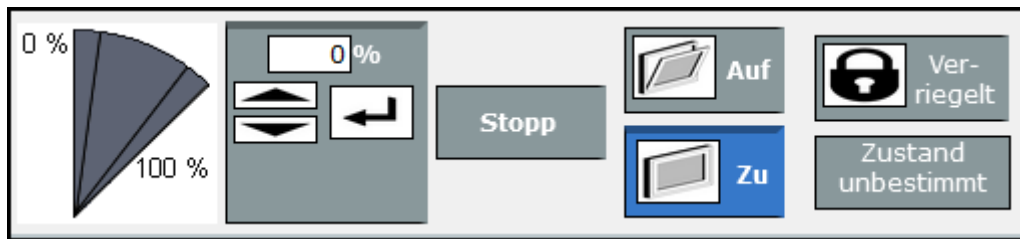
Parameter	
NODE	0: [101] Treppe Fenstermotor
SPEED	DEFAULT
PRIORITY	5: Comfort Level 2
PLI	33333333,30 PLI,LockTime
Zyklische Statusmeldung	<input checked="" type="checkbox"/>
POLL_MP_MIN	0 min (0-60)

Beschreibung wie Konfigurationsparameter, zusätzlich:

POLL_MP_MIN - Abfrageintervall in Minuten vom **MP_MIN** Geräteparameter zur Regenerkennung (Default = 0, Deaktiviert)

Kanal	Parameter
Ch.: 1	WINMATIC INVERT <input type="checkbox"/>
	WINMATIC SPEED_PARAMETER 1 (0-4)

siehe Konfigurationsparameter



Kanaltypen:

Kanaltyp	Kanalnummer
WINMATIC	1

Kanaltyp WINMATIC:

DP-Name	Typ	Zugriff	Beschreibung
LEVEL	float	lesend schreibend	Fenster öffnen, schließen bzw. anfahren einer bestimmten Position
STOP	action	schreibend	Fensterbewegung anhalten
WORKING	boolean	lesend	kennzeichnet Aktivität (Bewegung)
DIRECTION	enum	lesend	Fensterbewegung (0..keine, 1..öffnend, 2..schließend). Wird eine Bewegung erkannt, dann erfolgt alle 5s ein LEVEL -Update.
ON_TIME	integer	schreibend	nach Ablauf der angegebenen Sekunden wird nach einer LEVEL -Änderung die <u>letzte Position</u> oder ein zuvor gesetzter ON_TIME_LEVEL wieder hergestellt. (z.B. als Lüftungszeit anwendbar). Ist gerade ein Timer aktiv, dann wird die Zeit entsprechend verlängert und die gespeicherte <u>letzte Position</u> bleibt unverändert. Eine manuelle Bedienung während dieser Zeit deaktiviert den Timer.
ON_TIME_LEVEL	float	schreibend	LEVEL nach Ablauf von ON_TIME überschreiben (z.B. Fenster schließen unabhängig von der <u>letzten Position</u>).
SPEED	integer	lesend schreibend	Bewegungsgeschwindigkeit linear setzen. 0 =Default, 1 =Minimal ... 51200 =Maximal

Allgemeine Kanaldatenpunkte siehe Kanaldatenpunkte.

Regenerkennung:

Bei Fenstermotoren mit internem Regensensor wird die Fensteröffnung bei Regenerkennung per Default automatisch begrenzt, indem **MP_MIN** bei Regen auf einen vorkonfigurierten festen Wert gesetzt wird. Dieser Wert kann sich pro Fenster unterscheiden.

Mittels **POLL_MP_MIN=5** Geräteparameter wird der aktuelle Begrenzungswert vom CUxD z.B. alle **5** Minuten per Funk vom Fenstermotor abgefragt. Bei meinem Velux **KMG100** beträgt er z.B. 45596 und bei meinem Velux **KSX100** z.B. 45568.

Die folgende Programmverknüpfung triggert die Regenerkennungen eines Fenstermotors und setzt eine Systemvariable:

Name	Beschreibung	Bedingung (Wenn...)	Aktivität (Dann..., Sonst...)	Aktion
Regensensor		Kanalzustand: Velux KMG100:1 bei MP_MIN im Wertebereich / mit Wert größer als 45500 bei Änderung auslösen	Systemzustand: Regen sofort auf ist wahr setzen	<input type="checkbox"/> systemintern

Bedingung: Wenn...

Geräteauswahl bei im Wertebereich / mit Wert bei Änderung auslösen

Aktivität: Dann... ☒ Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern).

Systemzustand sofort

Aktivität: Sonst... ☒ Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern).

Systemzustand sofort

Zum Prüfen und Aufzeichnen der empfangenen Datenpunkte inkl. **MP_MIN** im CUxD DEVICELOG kann folgende Ergänzung im CUxD Setup eingetragen werden:

```
DEVLOGFILE=/tmp/devlog.txt
DEVLOGSIZE=1000000
LOGIT=CUX1902
```

Regensensor bei langem Tastendruck für 15 Minuten (900s) deaktivieren und Fenster 70% öffnen:

Bedingung: Wenn...

Geräteauswahl bei

Aktivität: Dann... ☒ Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern).

Geräteauswahl	Velux KMG100:1	sofort	SET_LIMIT	0,0
Geräteauswahl	Velux KMG100:1	sofort	PLI	3033333,900
Geräteauswahl	Velux KMG100:1	sofort	Kippstellung	auf 70.00 %

Stoßlüftung bei kurzem Tastendruck für 20 Minuten (1200s):

Bedingung: Wenn...

Geräteauswahl bei

Aktivität: Dann... ☒ Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern).

Geräteauswahl	Velux KMG100:1	sofort	Einschaltdauer	auf 1200 s
Geräteauswahl	Velux KMG100:1	sofort	Kippstellung	auf 100.00 %

Aktionen hintereinander ausführen:

Sind an einem Dachfenster z.B. ein Fensteröffner und ein Rolladen verbaut, dann ist es mit Hilfe der im CUxD implementierten **MY_STATUS** Datenpunkte einfach möglich, eine Programmverknüpfung zu erstellen, die den Dachrolladen erst dann herunterfährt, wenn das Dachfenster auch wirklich geschlossen ist. So können Schäden am Rolladen beim Anstoßen an das geöffnete Fenster verhindert werden.

Dazu muss eine Programmverknüpfung zum Schließen des Fensters und eine weitere Programmverknüpfung zum Herunterfahren des Rolladens konfiguriert werden. In der ersten Programmverknüpfung wird beim Schließen des Fensters (LEVEL=0) mittels **MY_STATUS_SET** ein eigener Statuswert definiert. Dieser Wert kann nach erfolgreicher Abarbeitung des Befehls genutzt werden, um mittels **MY_STATUS** Datenpunkt einmalig die zweite Programmverknüpfung zu triggern und den Rolladen herunterfahren.

Beispielkonfiguration:

1. Programmverknüpfung (Schließen des Fensters)

Bedingung: Wenn...			
Zeitsteuerung	Täglich nachts beginnend am 01.01.2022	zu Zeitpunkten auslösen	
+ ODER			
+ UND			
Aktivität: Dann... <input checked="" type="checkbox"/> Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern).			
Geräteauswahl	SZ Dachfenster Motor:1	sofort	Kippstellung auf 0.00 %
Geräteauswahl	SZ Dachfenster Motor:1	sofort	MY_STATUS_SET auf 123
+			
Aktivität: Sonst... <input type="checkbox"/> Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern).			
+			

2. Programmverknüpfung (Herunterfahren des Rolladens)

Bedingung: Wenn...			
Geräteauswahl	SZ Dachfenster Motor:1	bei MY_STATUS	im Wertebereich / mit Wert gleich 123 bei Aktualisierung auslösen
+ UND			
+ ODER			
Aktivität: Dann... <input checked="" type="checkbox"/> Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern).			
Geräteauswahl	SZ Dachfenster Blind:1	sofort	Behanghöhe auf 0.00 %
+			
Aktivität: Sonst... <input type="checkbox"/> Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern).			
+			

1.4 (CUX1903xxx) 4. Schalter

Mit diesem virtuellen Gerät können mit dem KLF200 gekoppelte Schalter über die CCU gesteuert werden.


CUxD Gerätetyp: (19) KLF200 Geräte ▼

Gerät: 4. Schalter ▼

Seriennummer: (numerisch max. 3 Stellen)

Name: (leer = wird autom. generiert)

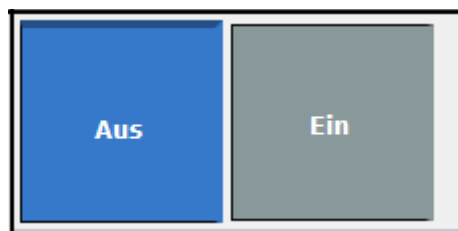
Geräte-Icon:
 ▼



Gerät auf CCU erzeugen !

Konfigurationsparameter:

siehe Konfigurationsparameter



Kanaltypen:

Kanaltyp	Kanalnummer
SWITCH	1

Kanaltyp SWITCH:

DP-Name	Typ	Zugriff	Beschreibung
STATE	boolean	lesend schreibend	Gerät Ein-/Ausschalten
TOGGLE	action	schreibend	Schaltzustand umschalten
WORKING	boolean	lesend	kennzeichnet Aktivität
ON_TIME	integer	schreibend	Einschaltzeit, eine manuelle Statusänderung während dieser Zeit deaktiviert den Timer.

Allgemeine Kanaldatenpunkte siehe Kanaldatenpunkte.

1.5 (CUX1904xxx) 5. Dimmer


Mit diesem virtuellen Gerät können mit dem KLF200 gekoppelte Lampen über die CCU gesteuert werden.

CUxD Gerätetyp: (19) KLF200 Geräte ▼

Gerät: 5. Dimmer ▼

Seriennummer: 1 (numerisch max. 3 Stellen)

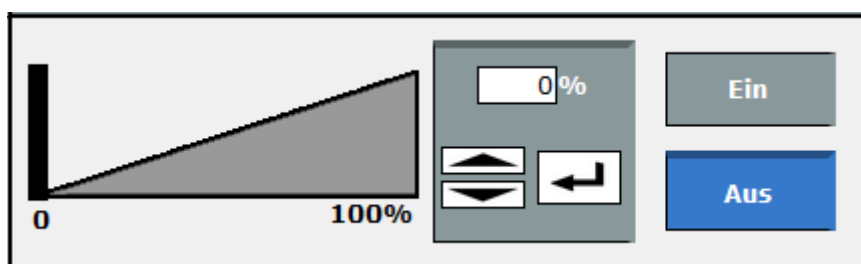
Name: (leer = wird autom. generiert)

Geräte-Icon: Dimmaktor 1fach Unterputz ▼ 

Gerät auf CCU erzeugen !

Konfigurationsparameter:

siehe Konfigurationsparameter



Kanaltypen:

Kanaltyp	Kanalnummer
DIMMER	1

Kanaltyp DIMMER:

DP-Name	Typ	Zugriff	Beschreibung
LEVEL	float	lesend schreibend	Dimmlevel setzen
OLD_LEVEL	action	schreibend	Dimmer einschalten
TOGGLE	action	schreibend	Dimmzustand umschalten
WORKING	boolean	lesend	kennzeichnet Aktivität
ON_TIME	integer	schreibend	Einschaltzeit, eine manuelle Level-Änderung während dieser Zeit deaktiviert den Timer.

Allgemeine Kanaldatenpunkte siehe Kanaldatenpunkte.

2 KLF200 Befehlsfolgen

Details zu den Parametern sind in der KLF200 API beschrieben.

Beispiele per HM-Script:

```
! MIN-Limit vom Rolladen (MP-Param) für 1 Stunde auf 50%=25600 setzen:
dom.GetObject("CUxD.CUX1901001:1.SET_LIMIT").State("0,25600,,3600");
```

```
! (MP-Param) alle Limits löschen „clear All“:
dom.GetObject("CUxD.CUX1901001:1.SET_LIMIT").State("0,,,a");
```

```
! Rolladen leise=1 auf 90% öffnen:
dom.GetObject("CUxD.CUX1901001:1.SPEED").State("1");
dom.GetObject("CUxD.CUX1901001:1.LEVEL").State("0.9");
```

```
! Rolläden schnell=51200 schließen:
dom.GetObject("CUxD.CUX1901001:1.SPEED").State("51200");
dom.GetObject("CUxD.CUX1901001:1.LEVEL").State("0");
```

```
! Rolläden mit den Node-ID's 0,3,4 gleichzeitig 100% öffnen:
dom.GetObject("CUxD.CUX1901001:1.SET_NODES").State("0,3,4");
dom.GetObject("CUxD.CUX1901001:1.LEVEL").State("1");
```

```
! die LEVEL aller Geräte in User-Gruppe 50 auf 35% setzen:
dom.GetObject("CUxD.CUX1901001:1.SET_NODES").State("G50");
dom.GetObject("CUxD.CUX1901001:1.LEVEL").State("0.35");
```

oder per Befehlszeilenaufruf:

MP_MIN Abfrage vom Fensteröffner manuell starten:

```
/usr/local/addons/cuxd/extra/timer.tcl CUxD.CUX1902001:1.GET_LIMIT 0
```

MIN-Limit vom Rolladen für 10 Minuten auf 70%=15360 setzen:

```
/usr/local/addons/cuxd/extra/timer.tcl CUxD.CUX1901001:1.SET_LIMIT 0,15360,,600
```

Identifizierung des Rolladens für 30s aktivieren:

```
/usr/local/addons/cuxd/extra/timer.tcl CUxD.CUX1901001:1.WINK 30
```

Rolladen mit mittlerer Geschwindigkeit=25600 öffnen:

```
/usr/local/addons/cuxd/extra/timer.tcl CUxD.CUX1901001:1.SET_PARAMETER 0,25600
```

Raffstore mit mittl. Geschwindigkeit schließen, danach Lamellen mit mittl. Geschwindigkeit öffnen:

```
.../extra/timer.tcl CUxD.CUX1901001:1.SET_PARAMETER 51200,25600,25600,30000
```

Per CUxD-Terminal bzw. CUxD Gerät **(40) Universalsteuerung** können alle in der API dokumentierten Datentelegramme ohne Einschränkung gesendet werden. Mehrere Datentelegramme können, durch Doppelpunkt getrennt, hintereinander in einem String übergeben werden. Der Datenaufbau ist folgendermaßen:

<COMMAND>|<Data1..n>:<COMMAND>|<Data1..n>:...

GW_GET_ALL_NODES_INFORMATION_REQ (0202) zum Aktualisieren der Node-Listen:

0202

GW_GET_SCENE_LIST_REQ (040C) zum Aktualisieren der Szenen-Listen:

040C

GW_GET_ALL_GROUPS_INFORMATION_REQ (0229) zum Aktualisieren der Gruppen-Listen:

0229|00 00

GW_GET_NODE_INFORMATION_REQ (0200) zum Aktualisieren von Node 0 und Node 1:

0200|00:0200|01

GW_HOUSE_STATUS_MONITOR_ENABLE_REQ (0240) HSM aktivieren (Default):

0240

GW_HOUSE_STATUS_MONITOR_DISABLE_REQ (0242) HSM deaktivieren:

0242

3 KLF200 Verbindungsprobleme und Restart

Es kommt manchmal vor, daß sich das KLF200 Gateway nach längerer Betriebszeit oder wenn mehrere Prozesse gleichzeitig darauf zugreifen, aufhängt und nicht mehr reagiert. Danach ist in der Regel kein neuer Verbindungsaufbau zum Gateway möglich.

Vorbeugend wäre hier mittels CUxD ein periodischer KLF200-Neustart über den REBOOT Datenpunkt und eine Programmverknüpfung möglich.

Zuverlässiger ist es jedoch, die KLF200-Verbindung mittels CUxD (40) Universalsteuerung zu überwachen und nur bei echten Verbindungsfehlern über ein Funk-Schaltmodul o.ä. durch kurzzeitiges Trennen der Versorgungsspannung einen Neustart auszulösen.

Konfiguration der CUxD (40) Universalsteuerung:

Name	Kanal	Parameter
KLF200_CONN:4 Taster	Ch.: 4	<div>KEY ACTIVE <input checked="" type="checkbox"/></div> <div>KEY REPEAT <input type="text" value="0"/> (0-2)</div> <div>KEY DEVICE <input type="text" value="ttySSL0"/></div> <div>KEY REG_MATCH <input type="checkbox"/></div> <div>KEY RCV_SHORT <input type="text"/></div> <div>KEY RCV_LONG <input type="text"/></div> <div>KEY CMD_EXEC <input checked="" type="checkbox"/></div> <div>KEY CMD_SHORT <input type="text"/></div> <div>KEY CMD_LONG <input type="text"/></div> <div>KEY CMD_CONNECT <input type="text" value="if ['\$DP\$' ~3d 'DISCONNECT']"/></div>

CMD_CONNECT:

```
if [ '$DP$' ~3d 'DISCONNECT' -a '$VALUE$' ~3d '3' ]; then extra/timer.tcl BidCos-RF.IEQ0274319:1.STATE 1 0 0 10;
sleep 1; extra/pty2tcp reconnect ttySSL0; fi
```

Nach 3 fehlerhaften Verbindungsversuchen (if ['\$DP\$' ~3d 'DISCONNECT' -a '\$VALUE\$' ~3d '3']) wird die Stromversorgung für 10s unterbrochen (extra/timer.tcl BidCos-RF.IEQ0274319:1.STATE 1 0 0 10) und die SSL Verbindung neu gestartet (extra/pty2tcp reconnect ttySSL0).