



zigbee

Overview

Warning

Beta driver. It is under active development: behavior and settings may change between releases, and some devices may not yet work reliably. Use at your own risk and report any issues to driver-support@finitelabs.com.

The Zigbee 3 Device driver brings **Zigbee 3.0** devices into Control4, with a single driver that covers every supported model. Join a device to the controller's Zigbee 3.0 network and the driver recognizes its model, adds the connections and variables it needs, shows its readings, and lets you configure it from the **Settings** tab.

Index

- [System Requirements](#)
- [Included Drivers](#)
 - [Zigbee 3 Device](#)
 - [Zigbee 3 Light](#)
 - [Zigbee 3 Lock](#)
- [Device Compatibility](#)
- [Installer Setup](#)
 - [DriverCentral Cloud Setup](#)
 - [Driver Installation](#)
 - [Joining a Device](#)
 - [Driver Setup](#)
 - [Driver Tabs](#)
 - [Settings](#)
 - [Driver Properties](#)
 - [Cloud Settings](#)
 - [Driver Settings](#)
 - [Connections](#)

- [Programming](#)
 - [Events](#)
 - [Conditionals](#)
 - [Variables](#)
 - [Commands](#)
- [Troubleshooting](#)
- [Known Issues](#)
- [Developer Information](#)
- [Support](#)
- [Changelog](#)

System Requirements

- Control4 OS 4.3.0 or later
- A controller running the **Zigbee 3.0** stack with an open network
- A Zigbee 3.0 device on the [supported devices](#) list ([Zigbee 3 Supported Devices.pdf](#))

Included Drivers

Zigbee 3 Device

Add one for each joined device. It recognizes the model and creates the connections, readings and settings that model needs. It is the only driver you add for sensors, switches, plugs, remotes, and any other device that needs no light or lock proxy.

Key features:

- Thousands of Zigbee 3.0 devices, with no hub or bridge
- Switches, relays and plugs as Control4 relays, one per gang
- Light and lock connections for the companion drivers below
- Remote and scene controller buttons as programming events and Button Links
- Readings as variables, and contact, temperature and humidity connections
- Security sensors enrolled automatically, with Battery Low and Trouble events
- Offline Warning and Back Online events
- Device settings and actions from the **Settings** tab and from programming

Zigbee 3 Light

The light companion. It presents a standard Control4 light proxy for a Zigbee 3.0 bulb or fixture. Add one for each light and bind its **Zigbee Light** connection to the matching Zigbee 3 Device.

Key features:

- On, off, dimming, color temperature and full color, as the light supports them
- Fades at the Control4 rates, Max and Min On Level, and Dim-To-Warm
- Advanced Lighting Scenes
- On, Toggle and Off Button Links for keypad buttons

Zigbee 3 Lock

The lock companion. It presents a standard Control4 lock proxy for a Zigbee 3.0 deadbolt or lever. Add one for each lock and bind its **Zigbee Lock** connection to the matching Zigbee 3 Device.

Key features:

- Lock, unlock and toggle from Navigators and programming
- Status and history that show how the lock was operated, such as at the keypad
- Battery status with Warning and Critical levels
- Optional lock code for remote operation

Each companion driver includes its own documentation accessible from within Composer Pro.

Device Compatibility

Every supported model, with what the driver controls and reads on each, is in the [Supported Devices](#) list. It opens from this driver's documentation in Composer Pro, and comes as `Zigbee 3 Supported Devices.pdf` with the documentation.

The driver adds a light, lock or relay only where the paired device reports one as it joins, so a device can show fewer controls than its model's row lists.

If a listed device does not work, or an unlisted one does, please email driver-support@finitelabs.com.

A device that will not join, a setting a device refuses, and the limits of this beta are covered in [Troubleshooting](#) and [Known Issues](#).

Installer Setup

DriverCentral Cloud Setup

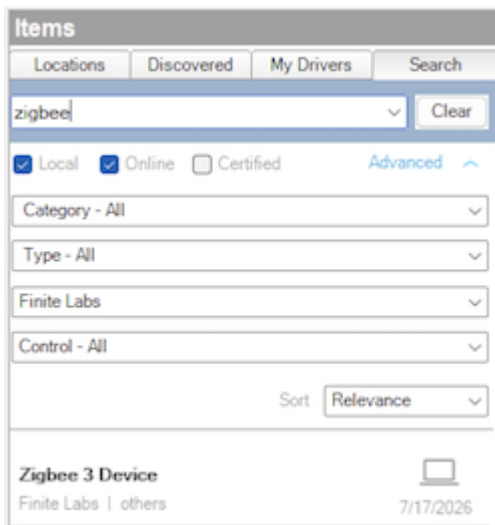
If you already have the [DriverCentral Cloud driver](#) installed in your project you can continue to [Driver Installation](#).

This driver relies on the DriverCentral Cloud driver to manage licensing and automatic updates. If you are new to using DriverCentral you can refer to their [Cloud Driver](#) documentation for setting it up.

Driver Installation

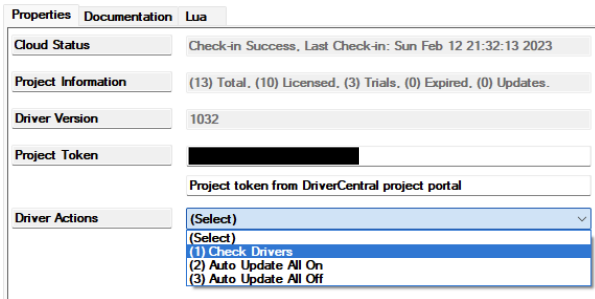
Driver installation and setup are similar to most other drivers. Below is an outline of the basic steps for your convenience.

1. Download the latest `control4-zigbee3.zip` from [DriverCentral](#).
2. Extract and [install](#) all `.c4z` files.
3. Use the "Search" tab to find the "Zigbee 3 Device" driver and add one to your project for each physical device you intend to join.



4. Select the newly added driver in the "System Design" tab. You will notice that the `Cloud Status` reflects the license state. If you have purchased a license it will show `License Activated`, otherwise `Trial Running` and the remaining trial duration.

5. You can refresh license status by selecting the "DriverCentral Cloud" driver in the "System Design" tab and performing the "Check Drivers" action.



The screenshot shows a software interface with three tabs: 'Properties', 'Documentation', and 'Lua'. The 'Properties' tab is active and contains several sections: 'Cloud Status' (Check-in Success, Last Check-in: Sun Feb 12 21:32:13 2023), 'Project Information' ((13) Total, (10) Licensed, (3) Trials, (0) Expired, (0) Updates), 'Driver Version' (1032), 'Project Token' (a redacted black box), and 'Driver Actions'. The 'Driver Actions' section has a dropdown menu open, showing options: '(Select)', '(1) Check Drivers', '(2) Auto Update All On', and '(3) Auto Update All Off'. The 'Project Token' field has a text input with the placeholder 'Project token from DriverCentral project portal'.

6. Continue to [Joining a Device](#).

Joining a Device

Join each device to the controller's Zigbee 3.0 network:

1. Confirm the controller is running the **Zigbee 3.0** stack.
2. Open the **Connections** view and select the **Network** tab.
3. Select **Zigbee Network**, then [double-click](#) this driver in the **Zigbee Network Connections** list (or highlight it and click **Identify**). This opens the identify window and starts a 60 second pairing timer.
4. Within that window, put the device into pairing mode (usually a factory reset or a press of the device's Zigbee button, see the device's manual). On success its address fills in and it shows online in the **Zigbee Mesh Routing Tree**.
5. The driver recognizes the device's model once it reports in, and the **Settings** tab fills in with its readings, settings and actions.

If the device does not join, or a battery device's settings stay *Applying*, see [Troubleshooting](#).

Driver Setup

Driver Tabs

While in System Design mode, the driver adds a **Settings** tab next to the "Properties", "Actions", and "Lua" tabs.

Settings

The **Settings** tab is where you view and configure the joined device. It has up to three views, switched with the tabs across the top: **Device** always, **Exposes** once the driver recognizes the device's model, and **Settings** when the device has any settings. The Device view names clusters and attributes the way open-source Zigbee tools do, such as `genPowerCfg` , so you can look a term up or quote it in a support request.

Device

What this device is and how it reports, plus the driver's own options.

Demo Multi-Sensor

Example Devices DEMO-MS1

1 Applying (tap the device's button): Detection interval, Temperature

2 Vendor	Example Devices
Model	DEMO-MS1
Description	Demo multi-sensor + valve
Model identifier	DEMO-MS1-Z
3 IEEE address	02000000DEC0DE01
4 Power source	Battery
Firmware	20230627 (application 0x11)
5 Last seen	2026-08-25 14:32:10
Enrollment	Enrolled

6 ENDPOINTS, BINDS AND REPORTING

7 Collapse all

8 Coordinator 02000000C0DE0001 Sync synced Bound 6 of 6 Checked 2026-08-25 14:32:10

Every cluster this device uses, whether it pushes to the coordinator, and the cadence of any reading it reports. Open a cluster with a caret for its attributes.

ENDPOINT 1

genBasic 0x0000

on demand

genPowerCfg 0x0001

9

bound

1 reporting

ATTRIBUTE	MIN	MAX	CHANGE	NEXT	STATE	11
batteryPercentageRemaining 0x0021	-	-	-	-	auto	
batteryVoltage 0x0020	Not supported by this device on firmware 20230627 (application 0x11)				not supported	

msIlluminanceMeasurement 0x0400

bound

on change

msTemperatureMeasurement 0x0402

bound

1 reporting

ATTRIBUTE	MIN	MAX	CHANGE	NEXT	STATE	12
measuredValue 0x0000	10s	1h	0.5 °C	-	applying <input type="button" value="override"/>	

msRelativeHumidity 0x0405

bound

1 reporting

ATTRIBUTE	MIN	MAX	CHANGE	NEXT	STATE	
measuredValue 0x0000	10s	1h	1 %	overdue	stale	

ssIasZone 0x0500

bound

reports alarms

ENDPOINT 2

genOnOff 0x0006	bound	on change
-----------------	-------	-----------

13 ? What every state means

14 PROPERTIES

The same properties as the native Properties tab, shown here so the web UI is a one-stop shop. Editing either side syncs.

DRIVER

Driver status	Driver version	Log level	Log mode
Applying (tap the device's butt	20260825	3 - Info	Off

CLOUD

Cloud status	Automatic updates
License Activated	On

1. **Status** - the same text as the [Driver Status](#) property; see the table there for what each value means.
2. **Vendor / Model / Description / Model identifier** - the device's identity. **Model** is the model as the [supported devices](#) list names it; **Model identifier** is the string the device itself reports.
3. **IEEE address** - the device's permanent Zigbee address.
4. **Power source / Firmware** - whether the device runs on mains or battery, and the firmware it reports.
5. **Last seen / Enrollment** - when the device last spoke and, for security sensors (door, water, some motion), whether it has enrolled with the controller, which it must do before it sends alarms. **Enrollment** reads **Enrolled** once the sensor accepts, **Reporting** once it has sent its status (or an alarm) without accepting, which shows its messages arrive, or **Not enrolled** (see [Troubleshooting](#)).
6. **Endpoints, binds and reporting** - every cluster the device uses, grouped by endpoint, with how each one reaches the driver and the cadence of any reading it reports.
7. **Collapse all / Expand all** - folds or opens every cluster with a caret, the ones that list reporting attributes.
8. **Coordinator / Sync / Bound / Checked** - the controller's Zigbee address, whether the device's bindings are in place, how many of the clusters it can bind are bound (one that reads **polled** or **refused** is not counted), and when that was last checked. **Sync** reads **synced** once they are in place; **pending** while the controller places them; **disabled** when the device refused one, which then reads **polled** or **refused**; or **unknown** after the controller's Zigbee service restarted. A battery device takes its bindings only while it is awake, and on OS 4.3 the ones after the first can take hours, which a later Control4 OS release will fix (see [Known Issues](#)).
9. **bound** - how the cluster reaches the driver (see **Transport** below), and beside it what it carries: how many readings report, **on change**, or **on poll** when the driver reads it on a timer. A security sensor's zone reads **reports alarms** once it has enrolled or sent an alarm, **reports status** once it has sent only its status, or **not enrolled**.
10. **Attribute / Min / Max / Change / Next** - one row per reading the cluster reports. The device reports no more often than **Min**, at least every **Max**, and whenever the value moves by **Change**. **Next** is when the next report is due; **overdue** means it has not arrived, so check that the device is powered and in range.
11. **State** - whether the device agreed to that cadence (see **Reporting state** below). A row the device refused reads **Not supported by this device** with its firmware. A cadence you set in place of the driver's carries an **override** tag and shows in **Status** as Applying until the device confirms it.

12. ⚙ - opens the cadence editor for that attribute; **Reset to default** returns it to the driver's cadence. Faster reports drain a battery sensor sooner.
13. **What every state means** - the same vocabulary, on the page.
14. **Properties** - the driver's own options: driver status and version (read-only), log level, log mode, and the update settings. They are the same properties as the native [Cloud](#) and [Driver](#) groups; editing either side syncs.

The driver asks the controller to bind each cluster it needs, so readings and button presses arrive as they happen. A cluster that is not bound is read on a timer instead: its readings still update, less promptly, but a timed read never fires a button press or a detection such as motion. A binding a device holds straight to another device, such as a remote bound to a bulb, is kept only if the device had it when the driver first set it up; one made later is removed the next time the controller places the device's bindings, such as after a controller reboot.

Transport	Meaning
bound	Sends its readings as they happen
sent	Binding queued; a battery device takes it after it wakes. On OS 4.3 its bindings after the first can take hours, which a later Control4 OS release will fix
awaiting	Not bound yet; its readings still update in the meantime
checking	The driver is checking the device's bindings
polled	The device refused the binding, or the cluster cannot be bound; its readings still update without it
refused	The device refused the binding and the driver does not read it; any readings it has arrive on another cluster
on demand	Not bound; the driver reads it only when needed

Reporting state	Meaning
confirmed	The device acknowledged the cadence and is reporting on it
applying	Re-sent on each wake until the device confirms
default	The driver's cadence; the device has not reported on it yet
stale	Was confirmed, but the device has gone quiet
auto	The device reports on its own cadence and declined ours

Reporting state	Meaning
not supported	The device refused it and does not report it
override	A value you set in place of the default (the device's own value for a setting, the driver's cadence for reporting)

Exposes view

Device

Exposes

Settings

Exposes

Everything this device reports or accepts.

1

SENSOR VALUES (4)

Numeric readings the device measures.

Battery

87 %

Humidity

48.5 %

Illuminance

42 lux

2

Temperature

21.5 °C · 70.7 °F

3

BINARY STATES (2)

On or off conditions the device detects.

4

ENDPOINT 1

Occupancy

Detected

ENDPOINT 2

5

Water leak

-

6

CONTROLS (1)

Things the device can be told to do; their state is shown here.

Valve

Off

7

BUTTONS 3 actions

Presses the device sends; each fires a programming event.

8

Last press

1 Single · 2026-08-25 14:31:02

9

1 Double

0x0008 cmd 0x02 stepmode = 1

x0

1 Hold

0x0008 cmd 0x01 movemode = 0

x1

1 Single

0x0006 cmd 0x02

x2

Press counts are since the driver last started. Press a button on the remote to watch it register here.

10

ACTIONS (3)

Momentary commands sent to the device. Each fires once.

ENDPOINT 1

11

blink

Effect

Sensor calibration

ENDPOINT 2

Open valve

Readings are also published as variables for programming.

1. **Sensor values** - the numeric readings the device measures, each with its label, current value, and unit. Every reading here is also published as a Control4 variable for programming.

2. **Temperature** - a reading. A label with a dotted underline carries its description as a tooltip, and a smaller grey figure after the value is the same reading in its alternate unit.
3. **Binary states** - on or off conditions the device detects, worded for the sensor type, such as **Detected** / **Clear** or **Open** / **Closed**.
4. **Endpoint 1 / Endpoint 2** - readings are grouped under their endpoint when the items in that card span more than one endpoint.
5. **Water leak** - a reading the device declares but has not reported yet shows a muted dash rather than being hidden.
6. **Controls** - the things the device can be told to do, such as a switch, light, lock, or valve, with their current state. They are operated through the driver's [connections](#), not from this view.
7. **Buttons** - shown for a remote or scene controller: the presses the device can send. Each fires its own programming event and is also offered as a **Button Link** connection. A lock shows **Lock events** here instead, the wrong codes and code changes it reports, each a programming event with no Button Link, and its **Last event** in place of **Last press**.
8. **Last press** - the most recent press the driver saw and, for presses this session, when. **before restart** means only the press itself was kept from before the driver last started; until a press arrives it reads **None yet this session**.
9. **1 Double** - one row per press the remote can send, named as its programming event and Button Link connection are, with a count of presses since the driver last started. The row flashes as a press arrives, so press each button to find its row. The grey line beneath is the Zigbee command the press arrives as, useful in a support request.
10. **Actions** - one-time commands the device accepts, such as a self-test, a calibration, a light effect or a factory reset, grouped by endpoint. A battery device gets one on its next wake. A factory reset can make the device leave the Zigbee network (see [Troubleshooting](#)). **Identify** is not listed here; run it from the **Execute Action** programming command.
11. **Effect / Sensor calibration** - click a button to send its action; its description is its tooltip. An action with several options, such as a light's **Effect**, has a list before its button to pick one from. The same actions are available to programming through the **Execute Action** command, an action with options once per option (**Effect: blink**). A factory reset, such as **Factory reset** or **Reset to factory settings**, is the exception: it asks you to confirm before it is sent, with the same warning as **Reset to factory defaults** on the Settings view below, and programming does not offer it.

Settings view

Device Exposes **Settings**

Settings

Options stored on the device. A change is re-sent until the device confirms it.

1 SETTINGS 5 on this device

2 Flat By cluster

3 Reset to factory defaults

4 Discard changes

5 ENDPOINT 1

6 SETTING	VALUE	STATE 7
8 Detection interval 2 to 255 s	60 s	● applying override
9 Led brightness	Not supported by this device on firmware 20230627 (application 0x11)	● not supported
10 Motion sensitivity	medium	● confirmed
11 Occupancy timeout	90 s READ-ONLY	● read-only

ENDPOINT 2

SETTING	VALUE	STATE
12 Valve auto-close	☒	● confirmed

13 ? What every state means

⚠ **Reset to factory defaults** sends the Zigbee factory-reset command to the device. It clears all of the device's own settings, not just the ones shown here. Some devices ignore it, and some also leave the Zigbee network and must then be joined again (see [Joining a Device](#)). Your changes here are dropped too: the rows then show what the device reports, and reporting cadences return to the driver's defaults.

Some devices also have a factory reset of their own, a setting such as **Factory reset** or an action on the Exposes view. It carries the same warning, is sent only once you confirm it, and is not offered to programming. Unlike your other changes, it is sent that one time and is not kept or sent again.

1. **Settings** - how many settings this device exposes.
2. **Flat / By cluster** - **Flat** lists every setting under its endpoint; **By cluster** nests them under the Zigbee cluster they write to, each cluster a group you can fold, with **Expand all / Collapse all** in the header.
3. **Reset to factory defaults** - sends the Zigbee factory-reset command after asking you to confirm; see the warning above. A battery device takes it when it next wakes, and until then **Discard changes** cancels it and gives your changes back.
4. **Discard changes** - cancels the edits the device has not confirmed yet, along with any action or factory reset still waiting for a battery device, and clears **refused** marks. Your earlier values and anything the device already confirmed stay in force. Reporting cadences on the Device view are not affected; use their **Reset to default**.

5. **Endpoint 1 / Endpoint 2** - settings are grouped under their endpoint when they span more than one.
6. **Setting / Value** - the setting's name, with its description as a tooltip when it has one (dotted underline), and the control that edits it.
7. **State** - whether the device has taken the value (see **Setting state** below). A setting not reported yet, such as a battery device's before it first wakes, shows a dash in its field or its toggle at neither on nor off.
8. **Detection interval** - a numeric setting. The allowed range is under the name and the unit is inside the field; a number outside the range is not sent. A value with a name of its own, such as a start-up color temperature of *previous*, shows by that name. This one is still **applying**: it is re-sent until the device confirms it.
9. **Led brightness** - **Not supported by this device on firmware 20230627 (application 0x11)**: the driver offers the control, but this unit refused it. The firmware is the build and application version the device reports.
10. **Motion sensitivity** - an enum setting, edited with a dropdown.
11. **Occupancy timeout** - a **read-only** setting: the device reports it but offers no way to set it.
12. **Valve auto-close** - an on/off setting, edited with a toggle.
13. **What every state means** - the legend for the state column, the same as the table below.

Setting state	Meaning
confirmed	The device is using this value
not reported	The device has not reported its value yet
applying	Re-sent on each wake until the device confirms
read-only	The device reports it, but offers no way to set it
unavailable	Waits until another setting has a value, or a given one
not supported	Offered by the driver, but the device refused it
refused	The device refused the value you picked and keeps the one shown
override	A value you set in place of the default (the device's own value for a setting, the driver's cadence for reporting)

Your changes are kept across driver updates and controller reboots, and re-sent until the device confirms them. If the device refuses a value, the row reads **refused** and shows the device's own value; pick another value or use **Discard changes**. A setting that depends on

another one, such as a valve's timer temperature in timer mode, reads **unavailable** until the other has the value it needs, and its tooltip names the condition. Where the device takes two settings together, such as a plug's power threshold and its breaker, the first one set reads **applying** until the other is set too. A value the device reports that its row cannot set, such as a number past the row's range or an option its dropdown does not offer, is shown as the device reports it and never sent back; in a dropdown it is greyed out. A setting made of on/off parts, such as an Aqara FP300's detection range in quarter-meter zones, shows how many parts are on, with a checkbox for each part under its row. Its checkboxes work once the device has reported the setting.

Driver Properties

Cloud Settings

Cloud Status (read-only)

Displays the DriverCentral cloud license status.

Automatic Updates [Off | On]

Enables or disables automatic driver updates via DriverCentral.

Driver Settings

Driver Status (read-only)

Displays the device's current state, the same text as **Status** on the Settings tab's Device view.

Status	Meaning
Initializing	The driver is starting, which takes under a second
Waiting for device	The device has not reported in yet, or not since a controller or Zigbee Server restart. Check that it is joined and powered. A battery device also reads it while offline, until it has been silent for twice its check-in period, or 48 hours if it names none
Online	The device is responding, and every change you made is in place
Offline	A mains device reads Offline once it has not been heard from for about half an hour; check its power and range. A battery device reads it once silent for twice its check-in period, or 48 hours if it names none

Status	Meaning
Applying: <settings>	Changes the device has not confirmed yet. On a battery device it reads Applying (tap the device's button): <settings> , so press the device's button to wake it
Refused: <settings>	The device refused the value you set. Pick another value, or use Discard changes on the Settings view. While other changes are still applying it follows them as ; refused: <settings>
Not supported	The device reported a model, or a version of one, that this driver does not support. Check the supported devices list

The readings, connections and events of an earlier model are removed once the device answers as one this driver does not support, as after a re-pair as another model (see [Connections](#)).

Without an active DriverCentral license, Driver Status reads No active license . See [Troubleshooting](#).

Driver Version (read-only)

Displays the current version of the driver.

Log Level [0 - Fatal | 1 - Error | 2 - Warning | 3 - Info | 4 - Debug | 5 - Trace | 6 - Ultra]

Sets the logging level. Default is 3 – Info .

Log Mode [Off | Print | Log | Print and Log]

Sets the logging mode. Default is Off . Any other mode returns to Off after 3 hours, and setting Off puts Log Level back to 3 – Info .

Connections

ZIGBEE (consumer)

Bind this to the controller's Zigbee Server so the driver can reach the physical device. This is the only static connection.

The driver has no fixed proxy. When it recognizes a device it adds only the connections that device needs, each named for the light, lock, sensor, relay, or button action it carries. A connection the device no longer has, such as after it is re-paired as another model, is removed; one still bound stays until you remove its binding and the driver next loads. A **Zigbee 3 Light** or **Zigbee 3 Lock** bound to one that stays reads *Disconnected*.

Light (provider, dynamic)

One `ZIGBEE_LIGHT` connection per light the device exposes; a multi-channel dimmer publishes one per channel. Bind a [Zigbee 3 Light](#) companion to it.

Lock (provider, dynamic)

One `ZIGBEE_LOCK` connection per lock endpoint. Bind a [Zigbee 3 Lock](#) companion to it.

Relay (provider, dynamic)

One `RELAY` control connection per switch gang, and one for a fan's on and off. While the device is offline a relay ignores commands and keeps its state. A controller restart, a driver update or a new connection runs no open or close programming and adds no history entry; a relay switched while the driver was not running runs it once, when the device reports its state.

Contact Sensor (provider, dynamic)

One `CONTACT_SENSOR` connection per binary sensor (motion, contact, leak, smoke, and so on). A controller restart, a driver update or a new connection runs no open or close programming and adds no history entry; a sensor that changed while the driver was not running runs it when the sensor reports the change.

Temperature / Humidity (provider, dynamic)

One `TEMPERATURE_VALUE` or `HUMIDITY_VALUE` connection per temperature or humidity reading. These are the only readings with a value connection; the rest (air quality, device temperature, and so on) are published only as variables. A bound thermostat or display is sent a reading when it changes, and when it asks for one. A driver update sends it nothing; a controller restart sends it the last reading at once, since it starts again without one. A new connection sends the reading once, and the thermostats and displays already on that reading receive it too.

Button Link (consumer, dynamic)

One `BUTTON_LINK` connection per button action a remote or scene controller can send. Wire one to a light's own button link to drive it directly; a tap clicks, and a held button ramps until it is released.

Programming

Events

Event	Description
Battery Low	Fires when a security sensor (door, window, water, motion) reports low battery
Trouble	Fires when a security sensor (door, window, water, motion) reports a fault
Offline Warning	Fires when the device has failed to report and is considered offline
Back Online	Fires when the device reports again after having gone offline

Offline Warning is not immediate: a mains device is marked offline about half an hour after it was last heard from, and a battery device once it has been silent for twice its own check-in period, or 48 hours if it names none. For a device not heard from since a controller restart, that time counts from the restart. Commands sent to a mains device in that half hour are not delivered. Back Online fires only after an Offline Warning. A controller reboot or Director restart fires neither event for a device that stays up, and a device already offline when the driver was added fires neither.

For remotes and scene controllers the driver adds one event per button action, such as *1 Single* or *2 Hold*, once it recognizes the model. A scene recall that names its scene, such as *Recall 12*, appears only after the remote first sends it, so press that button once before programming it. Some remotes send two actions for one press (a short hold can send a hold and then a single), and each fires its own event.

A lock adds one event for each wrong code or code change it reports, such as *Unlock Failure Invalid Pin Or Id* or *Pin Code Added*, with no Button Link. A code change appears only after the lock first reports one. Program a lock's locks and unlocks on the **Zigbee 3 Lock's** Locked and Unlocked events.

An event the device no longer sends is removed, but one used in programming stays until that programming is removed.

Conditionals

Conditional	Type	Description
DEVICE_ONLINE	BOOL	Tests whether the device is <code>Online</code> or <code>Offline</code> . It reads <code>Offline</code> while Driver Status reads <code>Offline</code> , so a battery device reads <code>Offline</code> only once Offline Warning has fired

Variables

Variable	Type	Description
<code>{Reading}</code>	NUMBER	One per reading the device measures, such as <code>Humidity</code> , <code>Illuminance</code> , <code>Power</code> , or <code>Battery</code>
<code>{Reading} C</code> / <code>{Reading} F</code>	NUMBER	A temperature reading, such as <code>Temperature</code> or <code>Device Temperature</code> , in Celsius and in Fahrenheit
<code>{Name} State</code>	BOOL	One per binary sensor, true while it detects (Detected , Open , and so on), and one per relay, true while on
<code>Battery State</code>	STRING	<code>Low</code> or <code>OK</code> , from a security sensor's low-battery flag. Other battery devices report their level in the <code>Battery</code> reading instead
<code>Last Action</code>	STRING	The last button action a remote sent or event a lock reported, such as <code>1_single</code> ; it changes only when the action differs
<code>Last Seen</code>	STRING	When the device last sent anything, as <code>YYYY-MM-DD HH:MM:SS</code>
<code>Zigbee Coordinator EUI</code>	STRING	The Zigbee address of the controller whose network the device is joined to, shared with this driver's other devices on that network
<code>Zigbee Coordinator Controller</code>	INT	The device ID of that controller, published with its Zigbee address

Note: {Reading} and {Name} are the names the **Exposes** view shows, such as Battery , Contact State , or Relay State . A device with the same reading or relay on several endpoints names each by its endpoint, such as Power (1) or Relay (L2) State , as its settings and actions are named. Which readings a device publishes depends on the device; its Exposes view lists them.

Commands

Command	Parameters	Description
Set Configuration	Setting	Sets one of the device's settings to a value
Set Numeric Configuration	Setting, Value	Sets one of the device's numeric settings to a value
Execute Action	Action	Runs a device action such as Identify or a self-test

Note: Setting and Action fill in once the driver recognizes the device. A setting is listed only after the device has reported it or accepted a value, so a battery device's settings appear after its first wake. **Set Configuration** lists each setting with each of its values, an on/off setting as Setting: On and Setting: Off , and each part of a setting made of on/off parts the same way, such as

Detection range 0.00m – 0.25m: On . **Set Numeric Configuration** lists each numeric setting with its range, such as Detection interval (2 to 255 s) ; Value is the number to set, and a number outside the range is not sent. Action lists **Identify**, which blinks the device so you can find it, and the actions on the Exposes view, such as a smoke detector's self-test. None of these commands offers a factory reset, the device's own or **Reset to factory defaults**: some devices leave the Zigbee network when reset, so send one from the Settings tab, where you confirm it.

Troubleshooting

A device drops off as soon as it pairs The controller's Zigbee 3.0 network keeps only Zigbee 3.0 devices, and removes an older Zigbee device the moment it joins. Its address never fills in, it never shows in the **Zigbee Mesh Routing Tree**, and it may keep joining and leaving on its own. First-generation Xiaomi and Aqara battery sensors are a common example; their current models pair. A factory reset, pairing it again, or pairing it near a different router does not help, and no driver can change it. Use a device whose specifications say Zigbee 3.0.

A Third Reality 3RSP02028BZ plug or 3RVS01031Z vibration sensor never finishes joining These are Zigbee 3.0 devices, but the firmware on the units we tested did not complete the security check the controller requires of a joining device. Other Third Reality models we tested join normally.

A battery device's setting stays *Applying* Battery devices sleep to save power, so a change is queued and delivered the next time the device wakes, and Driver Status reads *Applying* (tap the device's button) until then. Press the device's pairing button to wake it. Some devices need several presses a few seconds apart before every queued setting is delivered.

Sync stays pending on a battery device The controller places a battery device's bindings only while the device is awake, and on OS 4.3 it can place them one at a time, 15 minutes or more apart, which a later Control4 OS release will fix. A device that sends readings on its own needs nothing, and its readings keep updating in the meantime. A device that wakes only when it is used, such as a remote, can need a press of its button about every 15 minutes until **Sync** reads **synced**. Pressing it more often does not speed this up.

A remote bound straight to a bulb stops controlling it The controller keeps a binding a device holds straight to another device only if the device had it when the driver first set it up. One made later, such as a remote bound to a bulb, is removed the next time the controller places the device's bindings, such as after a controller reboot. Link the remote to the light in Control4 instead, with its [Button Link](#) connections.

A device reads *Waiting for device* After a controller or Zigbee Server restart the controller takes each device back only once it hears from it, and it can mark a battery device offline while it sleeps. The driver reads the device *Waiting for device* until it reports. A mains device reads *Offline* if it has not reported within about half an hour of the restart, and a battery device if it has not reported within twice its check-in period, or 48 hours if it names none (see [Events](#)). Press a battery device's button to wake it sooner.

A device that lost power still reads **Online** A mains device reads *Offline*, and Offline Warning fires, about half an hour after it was last heard from. See [Events](#).

A security sensor's Enrollment reads **Not enrolled** The sensor has not accepted the controller's enrollment, and some sensors never do. The driver enrolls it up to five times, then stops until the sensor restarts, as after a battery change, or is joined again. Many sensors send their status anyway: once a status message arrives, **Enrollment** reads **Reporting**, which shows the sensor's messages reach the controller. To confirm its alarms do too, trigger it once (open the door, walk in front of it, or wet a leak sensor's probe) and watch its state change on the **Exposes** view. If it stays **Not enrolled** and its state does not change, factory reset the sensor and join it again (see [Joining a Device](#)).

A setting reads Not supported by this device The driver is set up for each device's latest firmware. A device on older firmware may refuse a setting or a reading's reporting; the row then reads **Not supported by this device** with the firmware the device reports. Update the device's firmware with its maker's hub or app, and the driver asks again once the device reports the new version.

A device stops responding after a factory reset Some devices leave the Zigbee network when they are reset, from **Reset to factory defaults** on the Settings view or from the device's own reset action or setting. Join it again to the same Zigbee 3 Device driver (see [Joining a Device](#)).

An IKEA TRETAKT or INSPELNING plug's LED setting works the other way round On firmware older than 2.4.25 the plug applies its LED setting the other way round, and the driver does not correct for it. Update the plug's firmware before relying on that setting.

An Aqara FP300 or FP310 shows no Target Distance The sensor reports how far away the person it detects is only after its **Track target distance** action has run. Run it from the Exposes view, or with the **Execute Action** programming command. The `Target Distance` variable appears with the sensor's first report.

Set Configuration lists none of a battery device's settings Press the device's pairing button to wake it; its settings appear once it reports them. See [Commands](#).

A scene button's event is missing from programming Press that button once so the remote sends it, then program it. See [Events](#).

Driver Status reads No active license The driver has no active DriverCentral license, so it has stopped communicating with the device and shows its lights and locks offline, at the level or status they had. Check its **Cloud Status**, and refresh the license with the **DriverCentral Cloud** driver's **Check Drivers** action. The driver checks every 10 seconds and starts again once a license is active.

Known Issues

Programming across updates While the driver is in beta, an update can change or remove an event, variable, connection or command that programming uses. Each such change is called out in the [Changelog](#).

Battery devices on OS 4.3 After a battery device is paired, or reset and paired again, the controller can take several hours to place all of its bindings, and a day or more for a device that rarely wakes. Until then **Sync** reads **pending** and some clusters read **sent**. Readings still update when the device wakes, but a button press, or a reading the device sends as it

happens, can be missed on a cluster that reads **sent**. For a device that wakes only when it is used, see [Troubleshooting](#). A later Control4 OS release will fix the wait for the bindings after the first; a battery device will still take each one only while it is awake.

Locks The **Zigbee 3 Lock** driver does not yet manage users or PIN codes or their schedules, report which user operated the lock, or read the event log the lock keeps. A lock's own settings, such as **Auto relock time**, are on its Zigbee 3 Device's **Settings** tab where the lock offers them.

Fans A fan is controlled as an on and off relay. Its speed and mode are not controlled yet.

Thermostats, radiator valves and window coverings There is no thermostat or blinds proxy yet. Such a device works through the readings, settings and actions the [supported devices](#) list shows for it.

Aqara FP300 and FP310 presence sensors Their **Led disabled night** setting turns the LED off at night, but the night window's start and end times cannot be set yet.

Futurehome Charge EV charger and DIY AirWick aerosol dispenser The charger's session energy and charging duration, and the dispenser's last and next spray times, are not shown.

Device firmware The driver does not update a device's firmware. Update it with the device maker's own hub or app. Coming soon: Zigbee firmware updates (OTA) for supported devices, installed from the driver.

A device that refuses a binding On OS 4.3, when a device refuses one of the bindings the driver asks for, the controller stops placing the rest and **Sync** reads **disabled**; a later Control4 OS release will fix this. The driver finds the refused binding, which reads **polled** or **refused**, and asks for the others again: a mains device is through it in seconds, and a battery device takes a wake for each step. Until the others are placed, their readings still update, but a button press, or a reading the device sends as it happens, can be missed. A device whose Device view shows no **Coordinator** can stay **disabled** until another Zigbee 3 Device on the same controller reads **synced**, then moves on within about 15 minutes, or at a later wake on a battery device. If it is the only Zigbee 3 Device on that controller, join another device there.

A device that cannot report its bindings The controller reads a device's bindings before it places any. On OS 4.3, a device that cannot answer that read never has its bindings placed, and **Sync** never reads **synced**; a later Control4 OS release will fix this. Its readings still update, but a button press, or a reading the device sends as it happens, can be missed. Where those matter, use another device.

Developer Information



FINITE LABS

Copyright © 2026 Finite Labs LLC

All information contained herein is, and remains the property of Finite Labs LLC and its suppliers, if any. The intellectual and technical concepts contained herein are proprietary to Finite Labs LLC and its suppliers and may be covered by U.S. and Foreign Patents, patents in process, and are protected by trade secret or copyright law. Dissemination of this information or reproduction of this material is strictly forbidden unless prior written permission is obtained from Finite Labs LLC. For the latest information, please visit <https://drivercentral.io/platforms/control4-drivers/utility/zigbee3>

The driver's device definitions and Zigbee Cluster Library tables are generated from zigbee-herdsman and zigbee-herdsman-converters, open-source projects distributed under the MIT License. Their notices are in the `NOTICE` file in the `docs` folder of `control4-zigbee3.zip`.

Support

If you have any questions, supported-device requests, or issues integrating this driver with Control4, you can contact us at driver-support@finitelabs.com or call/text us at +1 (949) 371-5805.

Changelog

v20261029 - 2026-10-29

Added

- Added the **Zigbee 3 Device** driver for Zigbee 3.0 devices joined to the controller's own Zigbee network, with no hub or bridge, covering the models in the supported devices list. It requires Control4 OS 4.3.0 or later.

- Added switches, relays and plugs as Control4 relays, one per gang, and energy readings for metering plugs and switches.
- Added sensor readings as Control4 variables, contact, temperature and humidity sensor connections, and automatic enrollment of security sensors so they send alarms.
- Added a programming event and a Button Link connection for each button action on remotes and scene controllers.
- Added a programming event for each wrong code or code change a lock reports.
- Added Battery Low, Trouble, Offline Warning and Back Online events.
- Added the **Settings** tab, which shows each device's readings, bindings and reporting, and changes its settings and reporting intervals.
- Added device actions, such as Identify and a self-test, and programming commands that change a device's settings and run its actions.
- Added the **Zigbee 3 Light** companion driver, which presents a joined bulb or fixture as a standard Control4 light with dimming, color temperature and full color as the light supports them, fades at the Control4 rates, Dim-To-Warm, Advanced Lighting Scenes and On, Toggle and Off Button Links.
- Added the **Zigbee 3 Lock** companion driver, which presents a joined lock as a standard Control4 lock with lock, unlock and toggle, how the lock was operated, battery status and an optional lock code.
- Added DriverCentral licensing and automatic updates.
- Added a Supported Devices page to the **Zigbee 3 Device** documentation, which lists every supported model with what the driver controls and reads on it.
- Added a NOTICE file with the MIT License notices of zigbee-herdsman and zigbee-herdsman-converter, which the device definitions and Zigbee Cluster Library tables are generated from.