

Technical data:

Power supply:	12-32V DC	Power supply for gateway
	Power consumption (at 24V): 11 mA (full consumption depends on how many DALI devices are on DALi bus)	
	16-18V DC	Power supply for the DALI bus
	DC overvoltage protection:	±50 V
	Wrong wiring polarity protection	
Interface:	CAN FT	1
	DALI output	1
DALI interface	Maximum count of Ballasts per one CANx DALI gateway	64
Clamps:	Power supply:	1.5mm ²
	DALI output:	1.5mm ²
	CAN FT	CAN FT Connection Terminal
		0.8mm ²
	DALI power supply	1.5mm ²
Enclosure:	Material:	Polyamide
	Color:	Gray
	Dimensions:	36(W)x91(H)x56(L) mm
Protection:	IP20 according to EN 60529	
Usage temperature:	-5C ... +55C	
Storage temperature:	-20C ... +70C	
Net weight:	61g	
Gross weight:	73g	



Caution

Security advice

The installation and assembly of electrical equipment may only be performed by skilled electrician. The devices must not be used in any relation with equipment that supports, directly or indirectly, human health or life or with application that can result danger of people, animals or real value

Mounting advice

The devices are supplied in operational status. The cables connections included can be clamped to the housing if required.

Electrical connection

The devices are constructed for the operation of protective low voltage (SELV). Grounding of device not needed. When switching the power supply on or off, power surges must be avoided.

Default settings

Line ID: 0

Node ID: 1

Max. number of group addresses per object : 16

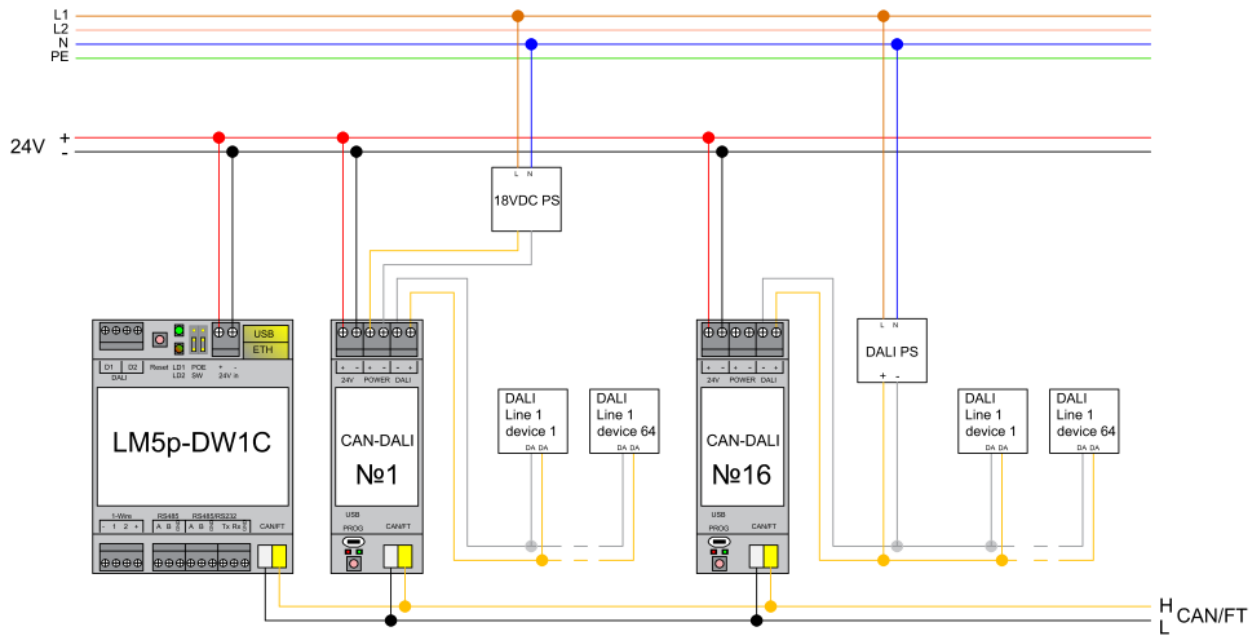
Reset to defaults

Press programming button for 5 seconds, the RED LED blinks 2 times, then release button - GREEN lights up shortly.

Programming physical address

Press *Tools* → *Write device address* from CANx app. Choose address and press *Write*. Then press programming button shortly on the device, GREEN LED lights up shortly. The LED is switched off automatically in 1 second which means address is written.

Connection diagram



Software configuration

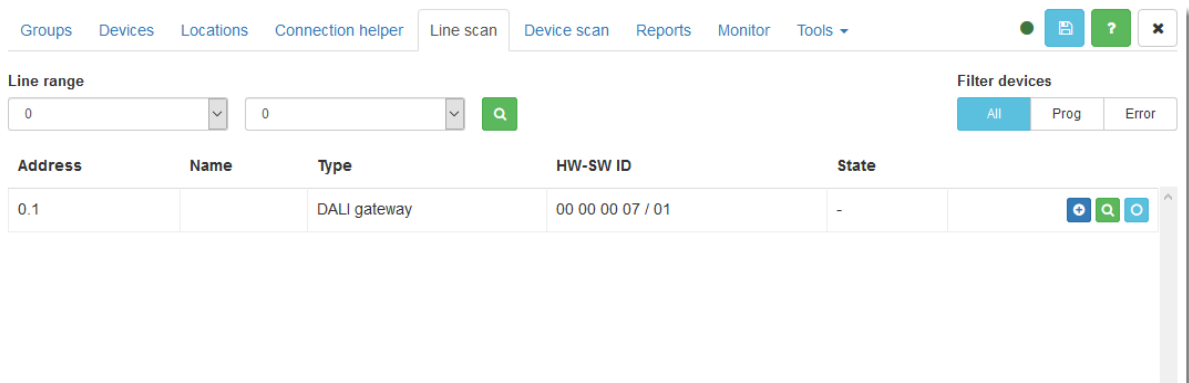
Two apps are mandatory for the CANx DALI Gateway:

- CANx app
- CANx DALI gateway

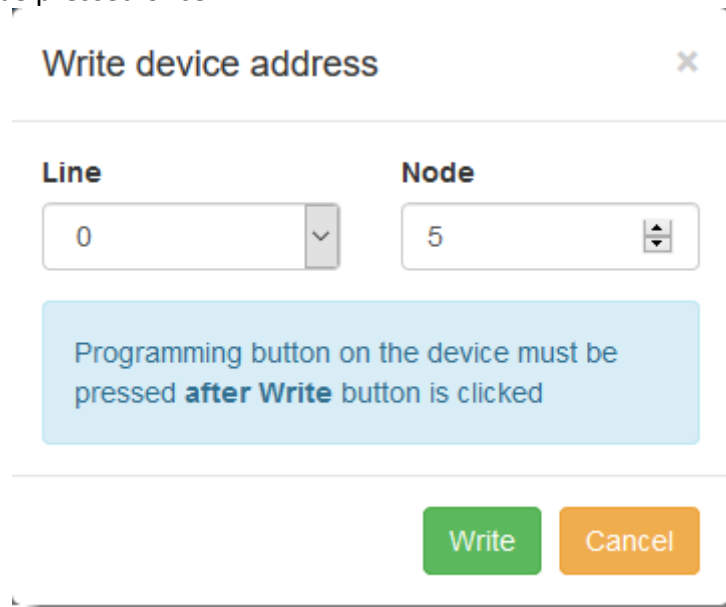
1. CANx app



As first step CANx DALI gateway must be configured in CANx app

- Open the app and scan the line under **Line scan** tab. The gateway will be found as 0.1 default address.



- Readdress the gateway to desire number by selecting **Tools -> Write device address**. Set the node number and press **Write**. After this procedure programming button on the DALI gateway has to be pressed once.



- Add device to Project by pressing 
- Go to Devices tab and select Configure 
- Set the **optional** association to DALI power supply short circiue status.

All Enabled Disabled
DALI

DALI power supply short circuit status (read-only) ✔

DALI power supply short circuit status (read-only)

Enabled v

Flags

F T R W

Group addresses + Add 1 bit (boolean)

x 0/0/1 DALI gateway - DALI power supply short circuit status (read-only)

Q

Tags

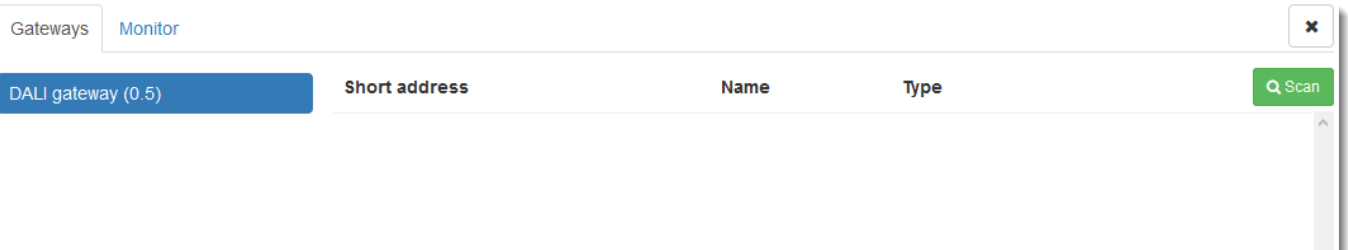
Q No tags set

↓ Save and write to device

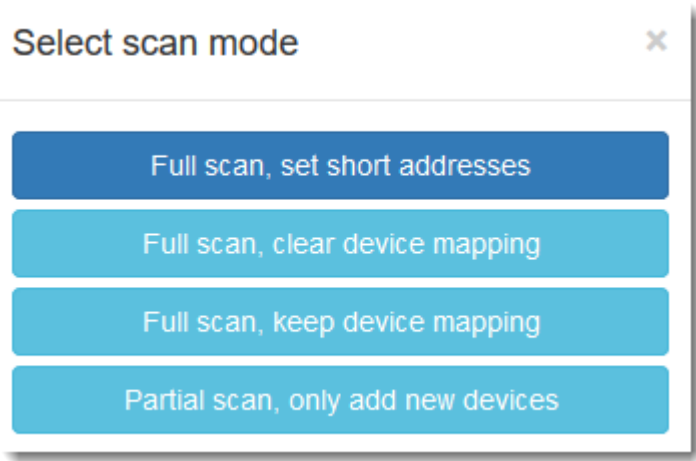
- Once done press

2. CANx DALI gateway app

- Open the CANx DALI gateway app and the configured DALI gateway will be available to select.





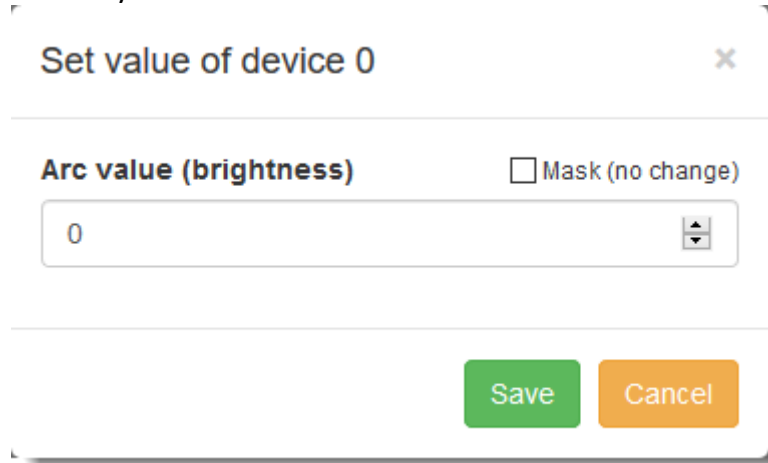
- Press **Scan** to search for connected slaves. Four options are possible



- Select desire option and scan will proceed. Found slaves will be available as list.

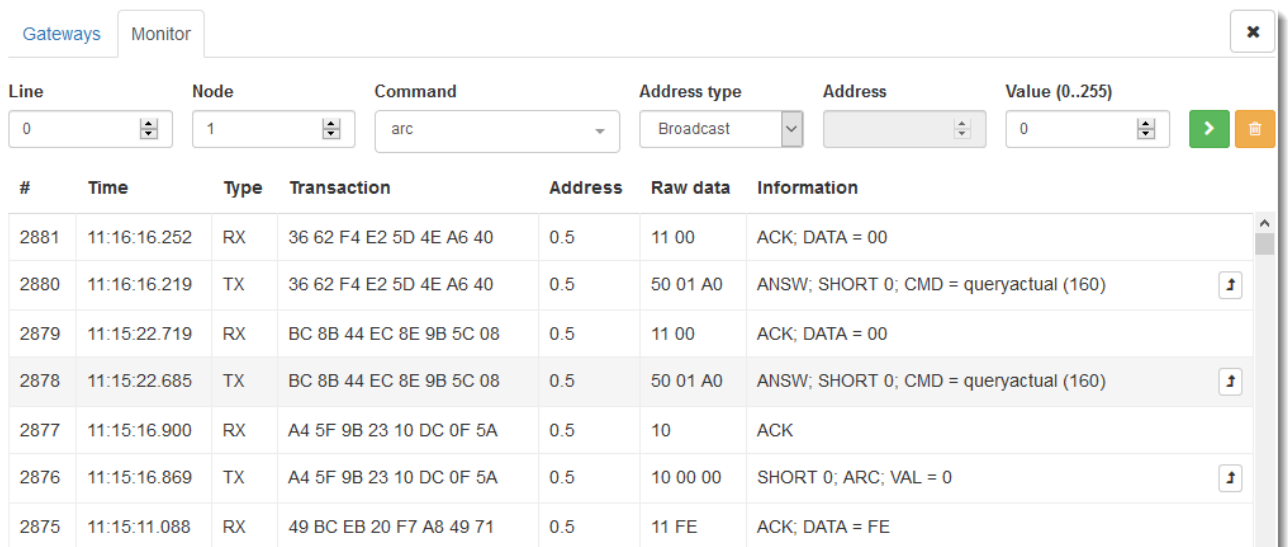


- Select  to assign group addresses to slave.
- Give a name to the slave, Select or create new group address for the **On/Off** and **Arc value**. On preset leve is the value which will be sent to the salve on **On** command. 254 is equal to 100%. Mapped group addresses are available both in CANx app objects list as well as in LogicMachine objects list.
- Press  to manually control the slave arc level.



Monitor

Monitor tool is used to monitor and send all commands on the DALI network.



- To send a command select DALI gateway line and a node. Select command, address type, address and respective value.
- All the commands correspond to DALI specifications.

DALI control commands from scripts

canxdali = require('applib.canxdali')

canxdali.sendcmds(req)

Sends single or multiple DALI commands to the given gateway.
Returns number of bytes sent or nil plus error message.
This is completely asynchronous function, it adds commands to gateway queue without waiting for returned results.

req table:

lineid - gateway line ID (number, required)
nodeid - gateway node ID (number, required)

command table:

cmd - command name (string, required)
value - command value (number, required for commands with a value)
address - DALI address (string or number, required)
addrtype - address type (string, required if address is a number)

address format:

address can be a string with following format:
s0..s63 - short address, from 0 to 63
g0..g15 - group, from 0 to 15
b - broadcast

if address is a number then *addrtype* is required, it can be either:

short
group
broadcast

Examples:

Send arc with value 0 to DALI short address 15 using gateway 0.1:

```
canxdali = require('applib.canxdali')
```

```
canxdali.sendcmds({
  lineid = 0,
  nodeid = 1,
  cmd = 'arc',
  address = 's15',
```

```
    value = 0,  
  })
```

Send multiple arc commands using gateway 1.42:

```
canxdali = require('applibs.canxdali')
```

```
canxdali.sendcmds({  
  lineid = 1,  
  nodeid = 42,  
  cmds = {  
    { cmd = 'arc', address = 's0', value = 50 },  
    { cmd = 'arc', address = 's4', value = 10 },  
  }  
})
```

canxdali.syncsendcmds(req)

Similar to `canxdali.sendcmds` but waits for each command to complete. On success returns Lua table with each command result, nil plus error message otherwise.

canxdali.sendqueries(req)

Similar to `canxdali.syncsendcmds` but checks for each command result, returns a table of values only for query type commands when all commands were successful. Useful for querying DALI device statuses.

canxdali.sethandler(type, fn)

Sets a callback to execute on a specific event. Callback is executed for each command inside data frame separately.

type - event type (string, required):

bus - all commands coming from bus side

busdata - only "bus data" type commands (from other master devices)

all - all commands coming to/from bus

fn - function to execute, or nil to remove callback (function or nil, required)

canxdali.step()

Waits for a frame or timeout, whichever happens first. Returns frame or nil plus error message on timeout. Frame can contain multiple commands when sent to bus.

Example (resident script):

```
if not canxdali then  
  function callback(frame)  
    log(frame)  
  end
```

```
  canxdali = require('applibs.canxdali')
```

```
    canxdali.sethandler('bus', callback)
end

canxdali.step()
```