MCC DIN8 MANUAL - DESCRIPTION OF API COMMANDS

USER
MANUAL

MCC DIN8 MANUAL

The structure of any packet sent to and from the LCC module is as follows:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucSubCmd;
uint8_t aucData[N];
uint8_t ucChecksum;
}
```

Example of a lamp switching command for channel 0:

ucHeader	usLength (2)	ucChannel	ucCommand	ucSubCmd	aucData (3)	ucChecksum
0x00	0x00, 0x0A	0x00	0xDC	0x00	0x00, 0x00, 0x01	0xE7

ucHeader – header, always 0x00

usLength – length of the entire packet in bytes

ucChannel – the channel number to which the packet is destined, from 0 to 7. Addressing of PLD DIN 1 modules is in accordance with section **Address setting on PLD DIN 1 modules.**

In addition, some of the packets can be directed to the MCC DIN 8 control module, in which case the ucChannel field takes the value 255 (0xff).

ucCommand – command from the command list. Values outside this list are ignored. Depending on the value of this field, the length and interpretation of the DATA field changes.

List of available commands:

CMD_APP_CHECKSUM_GET = 0x08,	CMD_DC_AMPS_READ = 0xB6,
CMD_APP_VERSION_GET = 0x09,	CMD_DC_AMPS_SET = 0xB7,
CMD_CONTROLLER_INFO = 0x0A,	CMD_DC_MAX_AMPS_READ = 0xBA,
CMD_JUMP_TO_APP = 0x0C,	CMD_DC_MAX_AMPS_SET = 0xBB,
CMD_FLASH_ERASE = 0x0D,	CMD_STROBE_AMPS_READ = 0xBE,
CMD_FLASH_WRITE = 0x0E,	CMD_STROBE_AMPS_SET = 0xBF,
CMD_JUMP_CONTROLLER_TO_BOOT = 0x0F,	CMD_STROBE_MAX_AMPS_READ = 0xC2,
CMD_JUMP_DRIVER_TO_BOOT = 0x1F,	CMD_STROBE_MAX_AMPS_SET = 0xC3,
CMD_DRIVER_INFO = 0x2A,	CMD_STROBE_PULSEWIDTH_READ = 0xC6,
CMD_UID_SERIAL = 0x0B,	CMD_STROBE_PULSEWIDTH_SET = 0xC7,
CMD_ACK = 0x61,	CMD_STROBE_MAX_PULSEWIDTH_READ = 0xCA,

List of available commands:

CMD_STROBE_MAX_PULSEWIDTH_SET = 0xCB,	CMD_GPO_PULSEWIDTH_READ = 0xAA,
CMD_STROBE_DELAY_READ = 0xCE,	CMD_GPO_PULSEWIDTH_SET = 0xAB,
CMD_STROBE_DELAY_SET = 0xCF,	CMD_GPO_DELAY_READ = 0xAE,
CMD_STROBE_PERIOD_READ = 0xD2,	CMD_GPO_DELAY_SET = 0xAF,
CMD_STROBE_PERIOD_SET = 0xD3,	CMD_STATUS = 0xF0,
CMD_STROBE_MIN_PERIOD_READ = 0xD6,	CMD_NETWORK_INFO_GET = 0xFF,
CMD_STROBE_MIN_PERIOD_SET = 0xD7,	CMD_NETWORK_IP_ADDRESS_SET = 0xF1,
CMD_OUTPUT_READ = 0xDE,	CMD_NETWORK_DHCP_SET = 0xF2,
CMD_OUTPUT_SET = 0xDC,	CMD_NETWORK_MAC_SET = 0xF3,
CMD_MODE_READ = 0xE1,	CMD_RTC_TIMEDATE_SET = 0xF5,
CMD_MODE_SET = 0xDF,	CMD_DRIVER_STATUS_GET = 0x20,
CMD_TRIGGER_SOURCE_READ = 0xE4,	CMD_DRIVER_EEPROM_WRITE = 0x2E,
CMD_TRIGGER_SOURCE_SET = 0xE2,	CMD_DRIVER_PARAM_SET = 0xF4,
CMD_TRIGGER_DC_ENABLE_SET = 0xE3,	CMD_DRIVER_LOCK_SET = 0xFA,
CMD_TRIGGER_POLARITY_READ = 0xEA,	CMD_DRIVER_DIAGNOSTICS_GET = 0xFB,
CMD_TRIGGER_POLARITY_SET = 0xE8,	CMD_DRIVER_CALIBRATION_SET = 0xFC,
CMD_KEYPAD_READ = 0xE7,	CMD_TEMP_WARNING_READ = 0x21,
CMD_KEYPAD_SET = 0xE5,	CMD_TEMP_WARNING_SET = 0x22,
CMD_GPO_STATUS = 0xA0,	CMD_TEMP_ALARM_READ = 0x23,
CMD_GPO_SOURCE_READ = 0xA2,	CMD_TEMP_ALARM_SET = 0x24,
CMD_GPO_SOURCE_SET = 0xA3,	CMD_TEMPERATURE_READ = 0x25,
CMD_GPO_OUTPUT_READ = 0xAD,	CMD_PRINTSCREEN_GET = 0xF6,
CMD_GPO_OUTPUT_SET = 0xAC,	CMD_CONTROLLER_FTP_FILE_SET = 0xF7,
CMD_GPO_POLARITY_READ = 0xA6,	CMD_CONTROLLER_PARAM_SET = 0xF8
CMD_GPO_POLARITY_SET = 0xA7,	

ucSubCmd / ucReserved – p subcommand, usually 0, used when a command needs to be specified, e.g. selection of the parameter to which the command refers.

aucData – data field, the length and structure of this field depends on the selected command. In most cases, the length of the data field is 3 and should be interpreted as one 24-bit value stored in big-endian format. In more complex cases, the data field should be interpreted as a command-specific structure.

ucChecksum – checksum of the packet, the 1-byte arithmetic sum of all bytes in the packet (without the ucChecksum field)

Most commands, both queries and responses, are 10 bytes long (3 bytes of aucData).

Simple commands, that set a single parameter, occur in pairs, e.g.:

CMD_DC_AMPS_READ = 0xB6,

CMD_DC_AMPS_SET = 0xB7.

The CMD_*_READ commands are used to read, while the CMD_*_SET commands are used to set the parameter value. In response to CMD_*_READ commands, the module sends back a frame with the parameter value in the aucData field.

In the case of commands to read several parameters simultaneously,
e.g: **CMD_STATUS = 0xF0, CMD_DRIVER_STATUS_GET = 0x20,**

Commands can be divided into 3 types:

- parameter request
- response to parameter request
- setting a paramater

Finding a device in the network

Using the command **CMD_NETWORK_INFO_GET = 0xFF**, it is possible to find all MCC modules on a given subnetwork.

CMD_NETWORK_INFO_GET command

ucHeader	usLength	ucChannel	ucCommand	ucSubCmd	aucData [3]	ucChecksum
0x00	0x000A	0xFF	0xFF	0x00	0x00, 0x00, 0x00	0x08

should be sent to the broadcast address of the given subnet x.y.z.255 on port 5000 (the default UDP port of the MCC module).

Upon receiving such a command, all available modules will respond with their network parameters:

ucHeader	usLength	ucChannel	ucCommand	ucSubCmd	aucData [19]	ucChecksum
0x00	0x001A	0xFF	0xFF	0x00	MAC[6], DHCP[1], IP[4], MASK[4], GATE[4]	0x??

e.g.: 0x00 0x00 0x1A 0xFF 0xFF 0x00 0x8C 0x1F 0x64 0x13 0x80 0x00 0x00 0xC0 0xA8 0x01 0x06 0xFF 0xFF 0xFF 0x00 0xC0 0xA8 0x01 0x01 0x90

is the response from the module:

- MAC address = 8C:1F:64:13:80:00
- DHCP = 0 (0 - means that the IP address is set in static mode, 1 - obtained from a DHCP server)
- IP address = 192.168.1.6
- netmask = 255.255.255.0
- network gateway = 192.168.1.1

Knowing the IP address of the MCC module, further commands can be sent directly to the module in UDP mode on port 5000 (default UDP port), or a TCP connection can be established on port 5001 (default TCP port).

Changing the default UDP and TCP ports is possible from the network settings menu of the MCC module. However, please note that after changing the default UDP port, finding such a module using the **CMD_NETWORK_INFO_GET** command will only be possible on the new UDP port.

CMD_APP_CHECKSUM_GET
- query packet structure:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucReserved;
uint8_t ucData[3];
uint8_t ucChecksum;
}
```

CMD_APP_CHECKSUM_GET
- structure of the response packet:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint16_t usChecksum;
uint8_t ucChecksum;
}
```

CMD_CONTROLLER_INFO, CMD_DRIVER_INFO
- structure of the response packet:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucSubCmd;
uint8_t ucExecuting; // 0 - bootloader, 1 application
uint8_t ucHardwareVersion[2];
uint8_t ucBootloaderVersion;
uint8_t ucApplicationVersion[2];
uint8_t ucApplicationChecksum;
uint8_t ucGitVersion[4];
uint8_t ucChecksum;
}
```

CMD_UID_SERIAL
- structure of the response packet:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucSubCmd;
uint8_t aucUID[16];
uint8_t aucSerial[16];
uint8_t ucChecksum;
}
```

CMD_ACK
- structure of the response packet:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucSubCmd; // komenda, której tyczy się ramka ACK
uint8_t ucACK; // status wykonania ACK = 0x06, NAK = 0x15
uint8_t ucIndex; // additional information, e.g. operation/
page number for commands with multiple members
uint8_t ucChecksum;
}
```

CMD_APP_VERSION_GET
- structure of the response packet:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint16_t usVersion;
uint8_t ucChecksum;
}
```

CMD_STATUS
- structure of the response packet:

```
{
uint8_t ucHeader
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucSubCmd;
uint8_t ucOutputEnable;
uint8_t ucOutputMode;
uint8_t ucTriggerEnable;
uint8_t ucTriggerPolarity;
uint16_t usDCCurrent;
uint16_t usDCCurrentLimit;
uint16_t usStrobeCurrent;
uint16_t usStrobeCurrentLimit;
uint16_t usStrobePulsewidth;
uint16_t usStrobePulsewidthLimit;
uint16_t usStrobeTriggerPeriod;
uint16_t usStrobeTriggerPeriodLimit;
uint16_t usStrobeTriggerDelay;
uint16_t usOutputVoltage;
uint16_t usOutputVoltageLimit;
uint16_t usUnknown0 = 0x0164;
uint8_t ucLocked;
uint8_t ucDeviceType;
uint8_t ucError[3];
uint8_t ucTriggerControlsDC;
uint8_t ucChecksum;
}
```

From version 0.09 CMD_STATUS
- structure of the response packet:

```
{
uint8_t ucHeader
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucSubCmd = 0x01;
uint8_t ucOutputEnable;
uint8_t ucOutputMode;
uint8_t ucTriggerEnable;
uint8_t ucTriggerPolarity;
uint16_t usDCCurrent;
uint16_t usDCCurrentLimit;
uint16_t usStrobeCurrent;
uint16_t usStrobeCurrentLimit;
uint24_t uiStrobePulsewidth;
uint24_t uiStrobePulsewidthLimit;
uint24_t uiStrobeTriggerPeriod;
uint24_t uiStrobeTriggerPeriodLimit;
uint24_t uiStrobeTriggerDelay;
uint8_t ucReserved0 = 0x00;
uint8_t ucLocked;
uint8_t ucDeviceType;
uint8_t aucError[3];
uint8_t ucTriggerControlsDC;
uint8_t ucChecksum;
}
```

Meaning of the fields:

ucOutputEnable - specifies whether the output is enabled (in DC mode)

0 - output enable

1 - output disable

ucOutputMode - channel operation mode

3 - DC mode

4 - Strobe mode

ucTriggerEnable - trigger source, DC and Strobe modes have a separate trigger source setting and the value depends on the ucOutputMode field

0 - local trigger

1 - global trigger

ucTriggerPolarity - trigger polarity

0 - falling edge

1 - rising Edge

usDCCurrent - current in DC mode, expressed in mA

usDCCurrentLimit - maximum allowed current in DC mode, expressed in mA

usStrobeCurrent - current in Strobe mode, expressed in mA

usStrobeCurrentLimit - maximum allowable current in Strobe mode, expressed in mA

usStrobePulsewidth / uiStrobePulsewidth - pulse length in Strobe mode, expressed in µs,

up to version 0.08, the field is 16 bits long, minimum value 2µs, maximum value 60000µs (60ms),

since version 0.09 field is 24 bits long, minimum value 2µs, maximum value 4000000µs (4000ms)

usStrobePulsewidthLimit / uiStrobePulsewidthLimit - maximum pulse length in Strobe mode, range of values as above

usStrobeTriggerPeriod / uiStrobeTriggerPeriod - gating period in Strobe mode between individual trigger pulses, expressed in µs,

up to version 0.08 the field is 16 bits long, minimum value 20µs, maximum value 60000µs (60ms),

since version 0.09 field is 24 bits long, minimum value 20µs, maximum value 4000000µs (4000ms)

usStrobeTriggerPeriodLimit / uiStrobeTriggerPeriodLimit - minimum gating period length, range of values as above

usStrobeTriggerDelay / uiStrobeTriggerDelay - delay time between the trigger pulse and the switching on of the lamp pulse. Range of values as for usStrobePulsewidth / uiStrobePulsewidth.

usOutputVoltage, usOutputVoltageLimit - unused fields.

ucLocked - determines whether the module allows special operations such as calibration, by default the module is locked

0 - module unlocked

1 - module locked

ucDeviceType - device type, always value 0x01

aucError[3] - error bit field

ucTriggerControlsDC - determines if in DC mode

0 - trigger is not supported

1 - input level of the trigger controls lighting of the lamp

CMD_TRIGGER_SOURCE_SET - packet structure:

```
{
uint8_t ucHeader ;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucSubCmd; // 0 - trigger for DC mode, 1 - trigger for Strobe mode
uint8_t aucData[3]; // aucData[3] == 0 – local, aucData[3] == 1 – global
uint8_t ucChecksum;
}
```

For the MCC module (channel 255), the **CMD_TRIGGER_SOURCE_SET** command controls whether the trigger is active:

CMD_TRIGGER_SOURCE_SET - packet structure:

```
{
uint8_t ucHeader ;
uint16_t usLength;
uint8_t ucChannel == 255;
uint8_t ucCommand;
uint8_t ucSubCmd;
uint8_t aucData[3]; // aucData[3] == 0 – disable, aucData[3] == 1 – enable
uint8_t ucChecksum;
}
```

CMD_TRIGGER_POLARITY_SET - packet structure:

```
{
uint8_t ucHeader ;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucSubCmd ; // 0 – polarity for DC mode, 1 – polarity for Strobe mode
uint8_t aucData[3]; // aucData[3] == 0 – falling, aucData[3] == 1 - rising
uint8_t ucChecksum;
}
```

For the MCC module (channel 255), the usSubCmd field is not used

CMD_NETWORK_IP_ADDRESS_SET - packet structure:

```
{
uint8_t ucHeader ;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucSubCmd ; // IP type: 0 – IP address, 1 – subnet mask, 2 – gateway address
uint8_t aucData[4];
uint8_t ucChecksum;
}
```

CMD_RTC_TIMEDATE_SET - packet structure:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucReserved;
uint8_t ucHours;
uint8_t ucMinutes;
uint8_t ucSeconds;
uint8_t ucDay;
uint8_t ucMonth;
uint8_t ucYear; // 2000 + ucYear
uint8_t ucChecksum;
}
```

CMD_DRIVER_STATUS_GET - packet structure:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucReserved;
uint16_t usCurrent;
uint16_t usVoltage;
int16_t ssTemperature;
int16_t ssLampTemperature;
uint8_t ucTrigger;
uint8_t ucChecksum;
}
```

CMD_DRIVER_PARAM_SET - packet structure:

{

uint8_t ucHeader;

uint16_t usLength;

uint8_t ucChannel;

uint8_t ucCommand;

uint8_t ucSubCmd;

uint8_t aucData[4];

uint8_t ucChecksum;

}

or

{

uint8_t ucHeader;

uint16_t usLength;

uint8_t ucChannel;

uint8_t ucCommand;

uint8_t ucSubCmd;

uint8_t aucData[16];

uint8_t ucChecksum;

}

The ucSubCmd field is used to select the parameter referenced by the **CMD_DRIVER_PARAM_SET** command and can take the following values:

PARAM_DEVICE_TYPE = 0,

PARAM_SERIAL_NUMBER = 1,

PARAM_DATE_OF_PRODUCTION = 2,

PARAM_EEPROM_VERSION = 3,

PARAM_CALIBRATION_VALID = 4,

PARAM_NAME = 5,

PARAM_LOCK = 6,

PARAM_PI

N = 7

For PARAM_NAME the aucData field is 16 bytes long and contains the module name displayed in the top bar instead of „Channel N", for the other values the aucData field is 4 bytes long and should be interpreted as a 32-bit value.

CMD_TEMP_WARNING_READ, CMD_TEMP_WARNING_SET, CMD_TEMP_ALARM_READ, CMD_TEMP_ALARM_SET

{

uint8_t ucHeader;

uint16_t usLength;

uint8_t ucChannel;

uint8_t ucCommand;

uint8_t ucSubCmd; // 0 – driver, 1 - lampa

uint8_t aucData[3];

uint8_t ucChecksum;

}

The aucData field contains the temperature to the nearest 0.1°C, in the range 25.0 ... 85.0, stored with a multiplier of 10. The WARNING temperature must not be higher than ALARM. If the newly set ALARM value is lower than WARNING, WARNING will be automatically lowered.

The ucSubCmd field selects whether the parameter relates to the module temperature - value 0, or the lamp temperature - value 1. The lamp temperature is taken into account if an external temperature sensor is detected.

If the WARNING value is exceeded, a small thermometer symbol will appear on the screen next to the temperature thathas been exceeded and the TEMPERATURE_NEAR_LIMIT flag will be lit in the module status field.

Exceeding the ALARM value will result in a large thermometer symbol appearing in the center of the screen, setting the TEMPERATURE_TOO_HOT flag in the module's status field, a change in the state of the GPO line if ALARM is selected as the GPO source, and an audible signal.

Further temperature rise above the CRITICAL value (90°C) will cause the lamp to be automatically switched off. Attempts to switch on the lamp will be ignored until the module (and lamp) temperature falls below the ALARM value.

CMD_TEMPERATURE_READ - structure of the response packet:

{

uint8_t ucHeader;

uint16_t usLength;

uint8_t ucChannel;

uint8_t ucCommand;

uint8_t ucReserved;

uint16_t usTemperature;

uint16_t usTemperatureWarning;

uint16_t usTemperatureError;

uint16_t usLampTemperature;

uint16_t usLampTemperatureWarning;

uint16_t usLampTemperatureError;

uint16_t usCPUTemperature;

uint16_t usCPUTemperatureWarning;

uint16_t usCPUTemperatureError;

uint8_t ucChecksum;

}

CMD_GPO_STATUS - structure of the response packet:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucReserved = 0;
uint8_t ucOutState;
uint8_t ucSource;
uint8_t ucPolarity;
uint16_t usPulsewidth;
uint16_t usDelay;
uint8_t ucChecksum;
}
```

0.09 CMD_GPO_STATUS - structure of the response packet:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucReserved = 1;
uint8_t ucOutState;
uint8_t ucSource;
uint8_t ucPolarity;
uint24_t uiPulsewidth;
uint24_t uiDelay;
uint8_t ucChecksum;
}
```

CMD_CONTROLLER_PARAM_SET - structure of packet:

```
{
uint8_t ucHeader;
uint16_t usLength;
uint8_t ucChannel;
uint8_t ucCommand;
uint8_t ucSubCmd;
uint8_t aucData[4];
uint8_t ucChecksum;
}
```

The ucSubCmd field is used to select the parameter referenced by the **CMD_CONTROLLER_PARAM_SET** commandand can take the following values:

PARAM_SCREEN_CONTRAST = 0,

PARAM_SCREEN_DISPLAY_TIME = 1,

PARAM_SCREEN_ROTATION = 2,

PARAM_KEY_LOCK = 3,

PARAM_KEY_SOUND = 4,

PARAM_KEY_PIN = 5,

PARAM_UPTIME = 6

The aucData field is 4 bytes long and should be interpreted as a 32-bit value:

PARAM_SCREEN_CONTRAST - display contrast, 0 to 100, in increments of 10,

PARAM_SCREEN_DISPLAY_TIME - display turn-on time, 0 - always, 30 - 30 seconds, 60 - 60 seconds

PARAM_UPTIME - time in seconds since switching on, read only

sales@airob.com

airob.com