# 2 System, Status, Versions

### *2.1 LAN_GET_SERIAL_NUMBER*

Reading the serial number of the Z21.

Request to Z21:

| DataLen | | Header | | Data |
|---|---|---|---|---|
| 0x04 | 0x00 | **0x10** | 0x00 | - |

Reply from Z21:

| DataLen | | Header | | Data |
|---|---|---|---|---|
| 0x08 | 0x00 | **0x10** | 0x00 | 32 Bits Serial number (little endian) |

### *2.2 LAN_LOGOFF*

Logging off the client from the Z21.

Request to Z21:

| DataLen | | Header | | Data |
|---|---|---|---|---|
| 0x04 | 0x00 | **0x30** | 0x00 | - |

Reply from Z21:
none

Use the same port number when logging out as when logging in.

**Note:** the login is implicitly done with the first command of the client (e.g. LAN_SYSTEM_STATE_GETDATA, ...).

### *2.3 LAN_X_GET_VERSION*

The X-Bus version of the Z21 can be read out with the following command.

Request to Z21:

| DataLen | | Header | | Data | | |
|---|---|---|---|---|---|---|
| | | | | X-Header | DB0 | XOR-Byte |
| 0x07 | 0x00 | **0x40** | 0x00 | **0x21** | **0x21** | 0x00 |

Reply from Z21:

| DataLen | | Header | | Data | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | X-Header | DB0 | DB1 | DB2 | XOR-Byte |
| 0x09 | 0x00 | **0x40** | 0x00 | 0x63 | 0x21 | **XBUS_VER** | **CMDST_ID** | 0x60 |

**XBUS_VER** X-Bus protocol version (0x30 = V3.0, 0x36 = V3.6, 0x40 = V4.0, … )
**CMDST_ID** Command station ID (0x12 = Z21 device family)

### *2.4 LAN_X_GET_STATUS*

This command can be used to request the Z21 status.

Request to Z21:

| DataLen | | Header | | Data | | |
|---|---|---|---|---|---|---|
| | | | | X-Header | DB0 | XOR-Byte |
| 0x07 | 0x00 | **0x40** | 0x00 | **0x21** | **0x24** | 0x05 |

Reply from Z21:
see *2.12* LAN_X_STATUS_CHANGED

This command station status is identical to the CentralState, which is delivered in the system status, see *2.18* LAN_SYSTEMSTATE_DATACHANGED.

### *2.5 LAN_X_SET_TRACK_POWER_OFF*

This command switches off the track voltage.

Request to Z21:

| DataLen | | Header | | Data | | |
|---|---|---|---|---|---|---|
| | | | | X-Header | DB0 | XOR-Byte |
| 0x07 | 0x00 | **0x40** | 0x00 | **0x21** | **0x80** | 0xa1 |

Reply from Z21:
see *2.7* LAN_X_BC_TRACK_POWER_OFF

### *2.6 LAN_X_SET_TRACK_POWER_ON*

This command switches on the track voltage, or terminates either the emergency stop or the programming mode.

Request to Z21:

| DataLen | | Header | | Data | | |
|---|---|---|---|---|---|---|
| | | | | X-Header | DB0 | XOR-Byte |
| 0x07 | 0x00 | **0x40** | 0x00 | **0x21** | **0x81** | 0xa0 |

Reply from Z21:
see *2.8* LAN_X_BC_TRACK_POWER_ON

### *2.7 LAN_X_BC_TRACK_POWER_OFF*

The following packet is sent from the Z21 to the registered clients when
- a client has sent command *2.5* LAN_X_SET_TRACK_POWER_OFF.
- or the track voltage has been switched off by some input device (multiMaus).
- and the relevant client has activated the corresponding broadcast, see ***2.16*** LAN_SET_BROADCASTFLAGS, Flag 0x00000001

Z21 to Client:

| **DataLen** | | **Header** | | **Data** | | |
|---|---|---|---|---|---|---|
| | | | | **X-Header** | **DB0** | **XOR-Byte** |
| 0x07 | 0x00 | **0x40** | 0x00 | **0x61** | **0x00** | 0x61 |

### *2.8 LAN_X_BC_TRACK_POWER_ON*

The following packet is sent from the Z21 to the registered clients when
- a client has sent command *2.6* LAN_X_SET_TRACK_POWER_ON.
- or the track voltage has been switched on by some input device (multiMaus).
- and the relevant client has activated the corresponding broadcast, see ***2.16*** LAN_SET_BROADCASTFLAGS, Flag 0x00000001

Z21 to Client:

| **DataLen** | | **Header** | | **Data** | | |
|---|---|---|---|---|---|---|
| | | | | **X-Header** | **DB0** | **XOR-Byte** |
| 0x07 | 0x00 | **0x40** | 0x00 | **0x61** | **0x01** | 0x60 |

### *2.9 LAN_X_BC_PROGRAMMING_MODE*

The following packet is sent from the Z21 to the registered clients if the Z21 has been put into CV programming mode by ***6.1*** LAN_X_CV_READ or ***6.2*** LAN_X_CV_WRITE and the respective client has activated the corresponding broadcast, see ***2.16*** LAN_SET_BROADCASTFLAGS, Flag 0x00000001

Z21 to Client:

| **DataLen** | | **Header** | | **Data** | | |
|---|---|---|---|---|---|---|
| | | | | **X-Header** | **DB0** | **XOR-Byte** |
| 0x07 | 0x00 | **0x40** | 0x00 | **0x61** | **0x02** | 0x63 |

### *2.10 LAN_X_BC_TRACK_SHORT_CIRCUIT*

The following packet is sent from the Z21 to the registered clients if a short circuit has occurred and the relevant client has activated the corresponding broadcast, see ***2.16*** LAN_SET_BROADCASTFLAGS, Flag 0x00000001

Z21 to Client:

| **DataLen** | | **Header** | | **Data** | | |
|---|---|---|---|---|---|---|
| | | | | **X-Header** | **DB0** | **XOR-Byte** |
| 0x07 | 0x00 | **0x40** | 0x00 | **0x61** | **0x08** | 0x69 |

### *2.11 LAN_X_UNKNOWN_COMMAND*

The following packet is sent from the Z21 to the client in response to an invalid request.

Z21 to Client:

| DataLen | | Header | | Data | | |
|---|---|---|---|---|---|---|
| | | | | X-Header | DB0 | XOR-Byte |
| 0x07 | 0x00 | **0x40** | 0x00 | **0x61** | **0x82** | E3 |

### *2.12 LAN_X_STATUS_CHANGED*

The following packet is sent from the Z21 to the client if the client explicitly sets the status to *2.4* LAN_X_GET_STATUS.

Z21 to Client:

| DataLen | | Header | | Data | | | |
|---|---|---|---|---|---|---|---|
| | | | | X-Header | DB0 | DB1 | XOR-Byte |
| 0x08 | 0x00 | **0x40** | 0x00 | **0x62** | **0x22** | **Status** | XOR-Byte |

DB1 … command station status

Bitmask for command station status:

```
#define csEmergencyStop          0x01   // The emergency stop is switched on
#define csTrackVoltageOff        0x02   // The track voltage is switched off.
#define csShortCircuit           0x04   // Short-circuit
#define csProgrammingModeActive  0x20   // The programming mode is active
```

This command station status is identical to the SystemState.CentralState, see *2.18* LAN_SYSTEMSTATE_DATACHANGED.

### *2.13 LAN_X_SET_STOP*

With this command the emergency stop is activated, i.e. the locomotives are stopped but the track voltage remains switched on.

Request to Z21:

| DataLen | | Header | | Data | |
|---|---|---|---|---|---|
| | | | | X-Header | XOR-Byte |
| 0x06 | 0x00 | **0x40** | 0x00 | **0x80** | 0x80 |

Reply from Z21:
see *2.14* LAN_X_BC_STOPPED

### *2.14 LAN_X_BC_STOPPED*

The following packet is sent from the Z21 to the registered clients when

- a client has sent command *2.13* LAN_X_SET_STOP.
- or the emergency stop was triggered by some input device (multiMaus).
- and the relevant client has activated the corresponding broadcast, see ***2.16*** LAN_SET_BROADCASTFLAGS, Flag 0x00000001

Z21 to Client:

| DataLen | | Header | | Data | | |
|---|---|---|---|---|---|---|
| | | | | X-Header | DB0 | XOR-Byte |
| 0x07 | 0x00 | **0x40** | 0x00 | **0x81** | 0x00 | 0x81 |

### *2.15 LAN_X_GET_FIRMWARE_VERSION*

The firmware version of the Z21 can be read with this command.

Request to Z21:

| DataLen | | Header | | Data | | |
|---|---|---|---|---|---|---|
| | | | | X-Header | DB0 | XOR-Byte |
| 0x07 | 0x00 | **0x40** | 0x00 | **0xF1** | 0x0A | 0xFB |

Reply from Z21:

| DataLen | | Header | | Data | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | X-Header | DB0 | DB1 | DB2 | XOR-Byte |
| 0x09 | 0x00 | **0x40** | 0x00 | 0xF3 | 0x0A | **V_MSB** | **V_LSB** | XOR-Byte |

**DB1** … MSB of the Firmware version
**DB2** … LSB of the Firmware version

The version is specified in BCD format.
**Example**:
`0x09 0x00 0x40 0x00 0xf3 0x0a` **`0x01 0x23`** `0xdb` … means: „Firmware Version **1.23**“

### *2.16 LAN_SET_BROADCASTFLAGS*

Set the broadcast flags in the Z21. These flags are set per client (i.e. per IP + port number) and must be set again the next time you log on.

Request to Z21:

| DataLen | | Header | | Data |
|---|---|---|---|---|
| 0x08 | 0x00 | **0x50** | 0x00 | 32 Bits Broadcast-Flags (little endian) |

Broadcast flags are an OR-combination of the following values:

0x00000001 Broadcasts and info messages concerning driving and switching are delivered to the registered clients automatically.
The following messages are concerned:
***2.7*** LAN_X_BC_TRACK_POWER_OFF
***2.8*** LAN_X_BC_TRACK_POWER_ON
***2.9*** LAN_X_BC_PROGRAMMING_MODE
***2.10*** LAN_X_BC_TRACK_SHORT_CIRCUIT
***2.14*** LAN_X_BC_STOPPED
***4.4*** LAN_X_LOCO_INFO (loco address must be subscribed too)
***5.3*** LAN_X_TURNOUT_INFO

0x00000002 Changes of the feedback devices on the R-Bus are sent automatically.
Z21 Broadcast messages see ***7.1*** LAN_RMBUS_DATACHANGED

0x00000004 Changes of RailCom data of subscribed locomotives are sent automatically.
Z21 Broadcast messages see **8.1** LAN_RAILCOM_DATACHANGED

0x00000100 Changes of the Z21 system status are sent automatically.
Z21 Broadcast messages see ***2.18*** LAN_SYSTEMSTATE_DATACHANGED

**From Z21 FW Version 1.20:**

0x00010000 Extends flag 0x00000001; client now gets LAN_X_LOCO_INFO LAN_X_LOCO_INFO without having to subscribe to the corresponding locomotive addresses, i.e. for all controlled locomotives!
Due to the high network traffic, this flag may only be used by adequate PC railroad automation software and **is NOT intended for mobile hand controllers** under any circumstances.
From FW V1.20 bis V1.23: LAN_X_LOCO_INFO is sent for **all** locomotives.
From **FW V1.24**: LAN_X_LOCO_INFO is sent for **all modified** locomotives.

0x01000000 Forwarding messages from LocoNet bus to LAN client without locos and switches.

0x02000000 Forwarding locomotive-specific LocoNet messages to LAN Client:
OPC_LOCO_SPD, OPC_LOCO_DIRF, OPC_LOCO_SND, OPC_LOCO_F912, OPC_EXP_CMD

0x04000000 Forwarding switch-specific LocoNet messages to LAN client:
OPC_SW_REQ, OPC_SW_REP, OPC_SW_ACK, OPC_SW_STATE

See also chapter ***9*** LocoNet.

**From Z21 FW Version 1.22:**

0x08000000 Sending status changes of LocoNet track occupancy detectors to the LAN client.
See **9.5** LAN_LOCONET_DETECTOR

**From Z21 FW Version 1.29:**

0x00040000 Sending changes of RailCom data to the LAN Client.
Client gets LAN_RAILCOM_DATACHANGED without having to subscribe to the corresponding locomotive addresses, i.e. for all controlled locomotives! Due to the high network traffic, this flag may only be used by adequate PC railroad automation software and is NOT intended for mobile hand controllers under any circumstances.
Z21 Broadcast messages see **8.1** LAN_RAILCOM_DATACHANGED

**From Z21 FW Version 1.30:**
0x00080000 Sending status changes of CAN-Bus track occupancy detectors to the LAN client.
See **10.1** LAN_CAN_DETECTOR

**From Z21 FW Version 1.41:**
0x00020000 Forward CAN-Bus booster status messages to LAN Client.
See **10.2.3**LAN_CAN_BOOSTER_SYSTEMSTATE_CHGD

**From Z21 FW Version 1.43:**
0x00000010 Send fast clock time messages to LAN client.
See **12.2** LAN_FAST_CLOCK_DATA

Reply from Z21:
none

**When preparing the settings for the broadcast flags, always consider the effects on the network load. This applies in particular to the broadcast flags 0x00010000, 0x00040000, 0x02000000 and 0x04000000!** The IP packets may be deleted by the router in case of overload and UDP does not offer any detection mechanisms for this! For example, before using flag 0x00000100 (system status) it is worth considering whether 0x00000001 with the corresponding LAN_X_BC_xxx broadcast messages would be a more suitable alternative. Not every application needs to be regularly informed in detail about the latest voltage, current and temperature values of the Z21.

### *2.17 LAN_GET_BROADCASTFLAGS*

Reading the broadcast flags in the Z21.

Request to Z21:

| DataLen | | Header | | Data |
|---|---|---|---|---|
| 0x04 | 0x00 | **0x51** | 0x00 | - |

Reply from Z21:

| DataLen | | Header | | Data |
|---|---|---|---|---|
| 0x08 | 0x00 | **0x51** | 0x00 | Broadcast-Flags 32 Bit (little endian) |

Broadcast-Flags see above.

### *2.18 LAN_SYSTEMSTATE_DATACHANGED*

Reports a change in the system status from the Z21 to the client.

This message is asynchronously reported to the client by the Z21 when the client

- activated the corresponding broadcast, see ***2.16*** LAN_SET_BROADCASTFLAGS, Flag 0x00000100.
- explicitly requested the system status, see ***2.19*** LAN_SYSTEMSTATE_GETDATA.

Z21 to Client:

| DataLen | | Header | | Data |
|---|---|---|---|---|
| 0x14 | 0x00 | **0x84** | 0x00 | **SystemState** (16 Bytes) |

**SystemState** is structured as follows (the 16-bit values are little endian):

| Byte Offset | Typ | Name | | |
|---|---|---|---|---|
| 0 | INT16 | MainCurrent | mA | Current on the main track |
| 2 | INT16 | ProgCurrent | mA | Current on programming track |
| 4 | INT16 | FilteredMainCurrent | mA | smoothed current on the main track |
| 6 | INT16 | Temperature | °C | command station internal temperature |
| 8 | UINT16 | SupplyVoltage | mV | supply voltage |
| 10 | UINT16 | VCCVoltage | mV | internal voltage, identical to track voltage |
| 12 | UINT8 | CentralState | bitmask | see below |
| 13 | UINT8 | CentralStateEx | bitmask | see below |
| 14 | UINT8 | reserved | | |
| 15 | UINT8 | Capabilities | bitmask | see below, from Z21 FW Version 1.42 |

Bitmask for CentralState:

```
#define csEmergencyStop          0x01   // The emergency stop is switched on
#define csTrackVoltageOff        0x02   // The track voltage is switched off
#define csShortCircuit           0x04   // Short-circuit
#define csProgrammingModeActive  0x20   // The programming mode is active
```

Bitmask for CentralStateEx:

```
#define cseHighTemperature       0x01   // temperature too high
#define csePowerLost             0x02   // Input voltage too low
#define cseShortCircuitExternal  0x04   // S.C. at the external booster output
#define cseShortCircuitInternal  0x08   // S.C. at the main track or programming track
```

**From Z21 FW Version 1.42:**

```
#define cseRCN213                0x20   // turnout addresses according to RCN-213
```

**From Z21 FW Version 1.42:**

Bitmask for Capabilities:

```
#define capDCC                   0x01   // capable of DCC
#define capMM                    0x02   // capable of MM
//#define capReserved            0x04   // reserved for future development
#define capRailCom               0x08   // RailCom is activated
#define capLocoCmds              0x10   // accepts LAN commands for locomotive decoders
#define capAccessoryCmds         0x20   // accepts LAN commands for accessory decoders
#define capDetectorCmds          0x40   // accepts LAN commands for detectors
#define capNeedsUnlockCode       0x80   // device needs activate code (z21start)
```

SystemState.Capabilities provides an overview of the device's range of features.
If SystemState.Capabilities == 0, then it can be assumed that the device has an older firmware version.
SystemState.Capabilities should not be evaluated when using older firmware versions!

### *2.19 LAN_SYSTEMSTATE_GETDATA*

Request the current system status.

Request to Z21:

| DataLen | | Header | | Data |
|---|---|---|---|---|
| 0x04 | 0x00 | **0x85** | 0x00 | - |

Reply from Z21:
see above ***2.18*** LAN_SYSTEMSTATE_DATACHANGED

### *2.20 LAN_GET_HWINFO*

**From Z21 FW Version 1.20 and SmartRail FW Version V1.13.**

Read the hardware type and the firmware version of the Z21.

Request to Z21:

| DataLen | | Header | | Data |
|---|---|---|---|---|
| 0x04 | 0x00 | **0x1A** | 0x00 | - |

Reply from Z21:

| DataLen | | Header | | Data | |
|---|---|---|---|---|---|
| 0x0C | 0x00 | **0x1A** | 0x00 | HwType 32 Bit (little endian) | FW Version 32 Bit (little endian) |

**HwType**:

```
#define D_HWT_Z21_OLD      0x00000200   // „black Z21“ (hardware variant from 2012)
#define D_HWT_Z21_NEW      0x00000201   // „black Z21“(hardware variant from 2013)
#define D_HWT_SMARTRAIL    0x00000202   // SmartRail (from 2012)
#define D_HWT_z21_SMALL    0x00000203   // „white z21“ starter set variant (from 2013)
#define D_HWT_z21_START    0x00000204   // „z21 start“ starter set variant (from 2016)

#define D_HWT_SINGLE_BOOSTER     0x00000205   // 10806 „Z21 Single Booster“ (zLink)
#define D_HWT_DUAL_BOOSTER       0x00000206   // 10807 „Z21 Dual Booster“ (zLink)

#define D_HWT_Z21_XL       0x00000211   // 10870 „Z21 XL Series“  (from 2020)
#define D_HWT_XL_BOOSTER   0x00000212   // 10869 „Z21 XL Booster“ (from 2021, zLink)

#define D_HWT_Z21_SWITCH_DECODER 0x00000301   // 10836 „Z21 SwitchDecoder“ (zLink)
#define D_HWT_Z21_SIGNAL_DECODER 0x00000302   // 10836 „Z21 SignalDecoder“ (zLink)
```

The **FW version** is specified in BCD format.

Example:
**0x0C 0x00 0x1A 0x00 0x00 0x02 0x00 0x00 0x20 0x01 0x00 0x00**
means: „Hardware Type **0x200,** Firmware Version **1.20**“

To read out the version of an older firmware, use the alternative command
***2.15*** LAN_X_GET_FIRMWARE_VERSION. Apply following rules for older firmware versions:

- V1.10 ... Z21 (hardware variant from 2012)
- V1.11 ... Z21 (hardware variant from 2012)
- V1.12 ... SmartRail (from 2012)

### *2.21 LAN_GET_CODE*

Read the software feature scope of the Z21 (and z21 or z21start of course).

This command is of particular interest for the hardware variant "z21 start", in order to be able to check whether driving and switching via LAN is blocked or permitted.

Request to Z21:

| DataLen | | Header | | Data |
|---|---|---|---|---|
| 0x04 | 0x00 | **0x18** | 0x00 | - |

Reply from Z21:

| DataLen | | Header | | Data |
|---|---|---|---|---|
| 0x05 | 0x00 | **0x18** | 0x00 | Code (8 Bit) |

**Code**:

```
#define Z21_NO_LOCK         0x00  // all features permitted
#define z21_START_LOCKED    0x01  // „z21 start”: driving and switching is blocked
#define z21_START_UNLOCKED 0x02  // „z21 start”: driving and switching is permitted
```