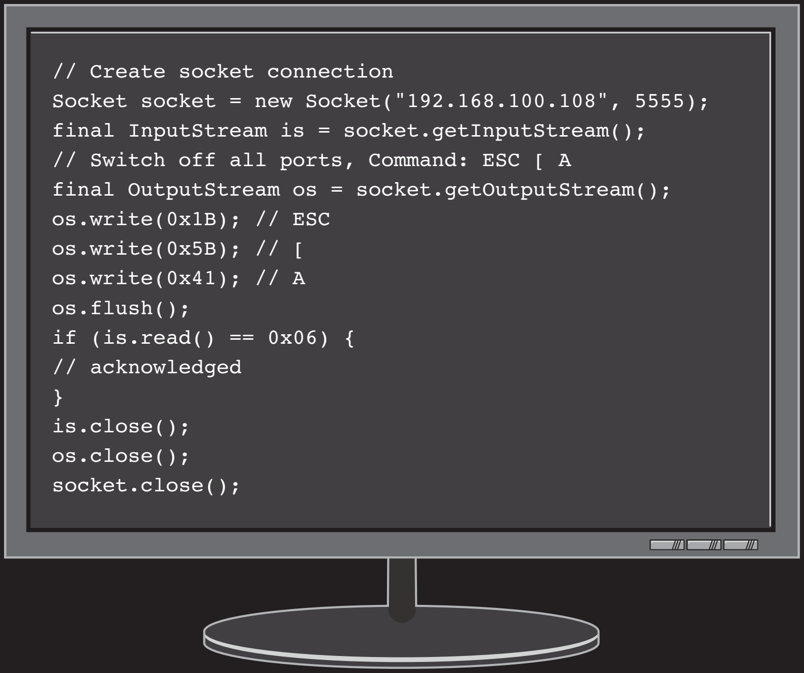
APPLICATION PROGRAM INTERFACE MANUAL

DKM-API

DKM KVM SWITCHES API MANUAL

24/7 TECHNICAL SUPPORT AT 1.877.877.2269 OR VISIT BLACKBOX.COM



```
// Create socket connection
Socket socket = new Socket("192.168.100.108", 5555);
final InputStream is = socket.getInputStream();
// Switch off all ports, Command: ESC [ A
final OutputStream os = socket.getOutputStream();
os.write(0x1B); // ESC
os.write(0x5B); // [
os.write(0x41); // A
os.flush();
if (is.read() == 0x06) {
// acknowledged
}
is.close();
os.close();
socket.close();
```

BLACK BOX

TABLE OF CONTENTS

CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

1. INTRODUCTION

1.1 ABOUT THIS MANUAL

This manual describes how to install your DKM Switch API, how to operate it, and how to perform troubleshooting.

1.2 SAFETY INSTRUCTIONS

To ensure reliable and safe long-term operation of your DKM Switch, note the following guidelines.

1.2.1 INSTALLATION

- Only use the device according to this User Manual. Failure to follow these procedures could result in damage to the equipment or injury to the user or installer.
- Only use in dry, indoor environments.
- The DKM Switch and the power supply units can get warm. Do not install components in an enclosed space without any airflow.
- Do not obscure ventilation holes.
- Only use power supplies originally supplied with the product or manufacturer-approved replacements. Do not use a power supply if it appears to be defective or has a damaged chassis.
- Connect all power supplies to grounded outlets. In each case, ensure that the ground connection is maintained from the outlet socket through to the power supply's AC power input.
- Do not connect the link interface to any other equipment, particularly network or telecommunications equipment.
- Only connect devices to the serial interface that are protected against short circuit currents and incorrect voltages at the serial interface.
- To disconnect the DKM Switch from the power supply, remove the power cord cables of all power supply units or switch supplies off.
- Take any required ESD precautions.
- To disconnect the device completely from the electric circuit, all power cables have to be removed.

1.2.2 REPAIR

- Do not attempt to open or repair a power supply unit.
- Do not attempt to open or repair the DKM Switch. There are no user-serviceable parts inside.
- Contact Black Box Technical Support at 877-877-2269 or info@blackbox.com if there is a fault.

2. DESCRIPTION

2.1 APPLICATION

The DKM Switch API is used to control the matrix externally by serial commands via a serial (RS-232) or network (TCP/IP) connection.

The DKM Switch API has been successfully implemented with various common media control systems.

The DKM Switch API provides the full scope of switching functionality. It does not support the configuration of a DKM Switch system.

2.2 ACCESS OPTIONS

You have the following options to access the DKM Switch for external serial control:

ACCESS OPTION	SYMBOL
Serial interface	RS232
TCP/IP interface	TCP/IP

Both serial interface and TCP/IP interface use the same commands for the operation of the DKM Switch matrix.

2.3 SYSTEM OVERVIEW

A DKM Switch matrix system consists of a DKM Switch matrix and, for KVM applications, one or more CPU Units/CON Units. The DKM Switch matrix is connected to the CPU Units/CON Units by interconnect cables or directly to the video devices where used as a video matrix.

CPU Units are connected directly to the sources (computer, CPU) by the provided cables.

Monitor(s), keyboard and mouse are connected to the CON Units.

Communication between the DKM Switch matrix and the CPU Units/CON Units occurs over the respective interconnect cables.

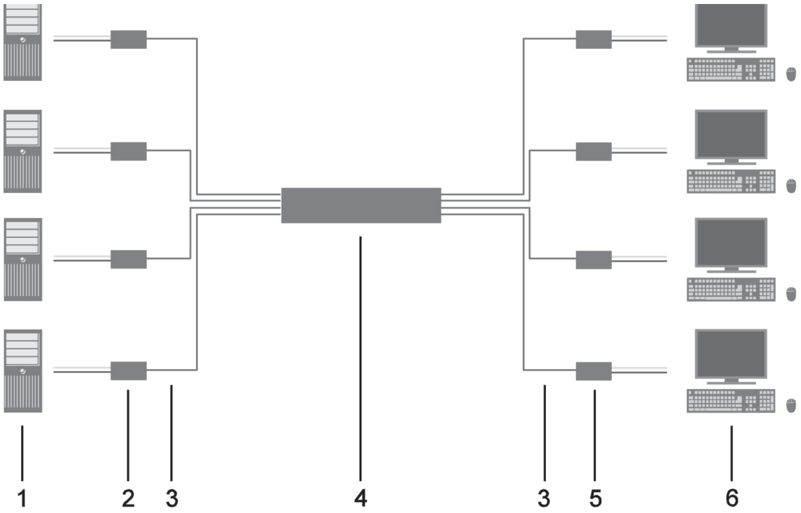


FIGURE 2-1.

System Overview (example)

1 Source (computer, CPU)

2 CPU Units

3 Interconnect cable

4 DKM Switch matrix

5 CON Units

6 Console (monitor, keyboard, mouse)

2.4 SYSTEM OVERVIEW EXTERNAL CONTROL

The DKM Switch matrix can be connected to an external serial control via the CPU board and its connectors.

The CPU board provides the possibility for both serial and TCP/IP connections.

The serial connection to an external serial control is established by using a serial cable with DB9 connectors or a DB9 to RJ-45 adapter cable (DKM Switch Compact).

The TCP/IP connection is established by using a CAtx network cable.

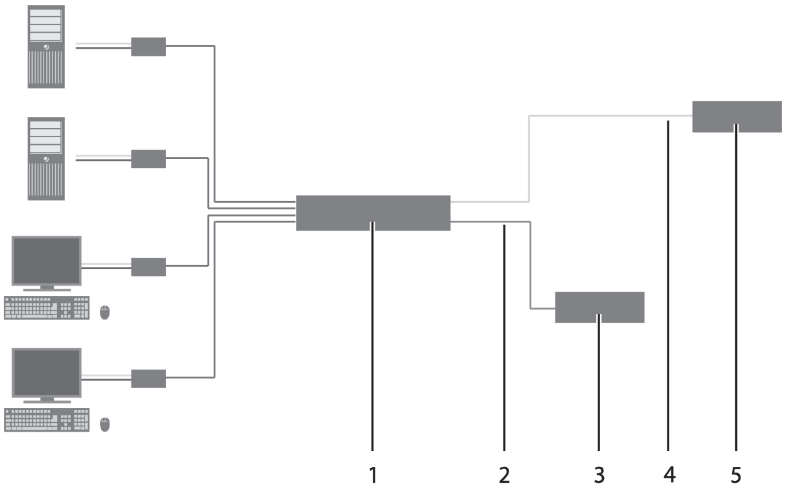


FIGURE 2-2.

System Overview (example)

1 DKM Switch matrix

2 Serial connection cable (D-Sub 9) or D-Sub 9 to RJ45 adapter cable

3 External serial control (RS232, option 1)

4 Network connection cable (Cat X)

5 External serial control (TCP/IP, option 2)

2.5 PRODUCT RANGE

PART NUMBER	DESCRIPTION
DKM-API	DKM Switch matrix application programming interface (API)

2.6 DIAGNOSTICS AND STATUS

2.6.1 STATUS LEDS CPU BOARD (ENTERPRISE)

The DKM Switch CPU board is fitted with the following LEDs for overall status indication:

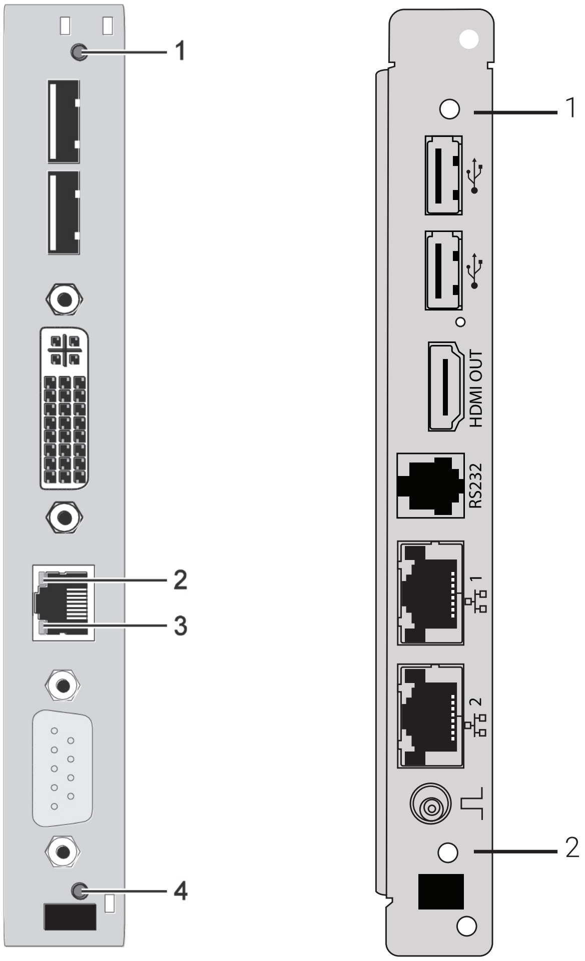


FIGURE 2-3. CPU BOARD, FRONT VIEW (LEFT: ACX288-R2-CTL; RIGHT: ACX288-R2-ADCTL)

STATUS LEDS ON CPU BOARD

POSITION	LED	STATUS	DESCRIPTION
1	Status 1	White	CPU board is in registration process
		Blue flashing	Registration at the matrix is started
		Red flashing	Registration in progress
		Green flashing	Operating condition
		Green	CPU board de-registered
2 (ACX288-R2-CTL)	TCP/IP Status 1	Red	Operating condition
		Off	No Connection
3 (ACX288-R2-CTL)	TCP/IP Status 2	Green flashing	Active data traffic
		Off	No active data traffic
4 (ACX288-R2-CTL) 2 (ACX288-R2-ADCTL)	Status 2	White	CPU board is in registration process
		Red flashing	Registration at the matrix is started
		Off	Operating condition

NOTE: Due to variations in LED type “white” might also appear as light purple or light blue.

2.6.2 STATUS LEDS CPU BOARD (COMPACT)

DKM Switch components are fitted with the following LEDs for overall status indication:

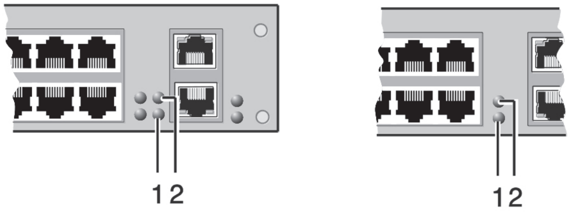


FIGURE 2-4.

CPU

Front View		Front View DKM Switch 8 port	
1	Status LED 2	1	Status LED 2
2	Status LED 1	2	Status LED 1

STATUS LEDS ON CPU BOARD (COMPACT)

POSITION	LED	STATUS	DESCRIPTION
1	Status 2	White	CPU board is in registration process
		Red flashing	Registration at the matrix is started
		Off	Operating condition
2	Status 1	White	CPU board is in registration process
		Blue flashing	Registration at the matrix is started
		Red flashing	Registration in progress
		Green flashing	Operating condition
		Green	CPU board de-registered

NOTE: Due to variations in LED type “white” might also appear as light purple or light blue.

CHAPTER 3: INSTALLATION

3. INSTALLATION

3.1 API DOWNLOAD

The API package is downloaded from blackbox.com. Contact Black Box Technical Support at 877-877-2269 or info@blackbox.com for details.

3.2 SYSTEM SETUP

We recommend that first-time users set up the system in the same room as a test setup. This will allow you to identify and solve any cabling problems, and experiment with your system more conveniently.

Setup of the external control

1. Install the CPU and I/O boards.
2. Connect keyboard, mouse, and monitor to the CPU board of the matrix.
3. Connect the matrix to the power supply.
4. Open the OSD via a hotkey and log in with administrator rights in the main menu.
5. Configure initially as requested.
6. Connect the external control either via RS-232 or TCP/IP to the matrix.

4. CONFIGURATION

4.1 GENERAL REMARKS

The DKM Switch API provides all commands that are necessary to switch the DKM Switch matrix.

4.2 DKM SWITCH CONFIGURATION

To operate the DKM Switch matrix, it has to be configured appropriately.

In the following section, all relevant chapters from the DKM Switch manual are described. For a detailed explanation, refer to the main manual.

4.2.1 SYSTEM DATA

The DKM Switch API relevant system configuration is set in this menu.

You have the following possibilities to access the menu:

ACCESS OPTION	SYMBOL
OSD icon	OSD
Java icon	JAVA

You can select between the following DKM Switch API relevant settings:

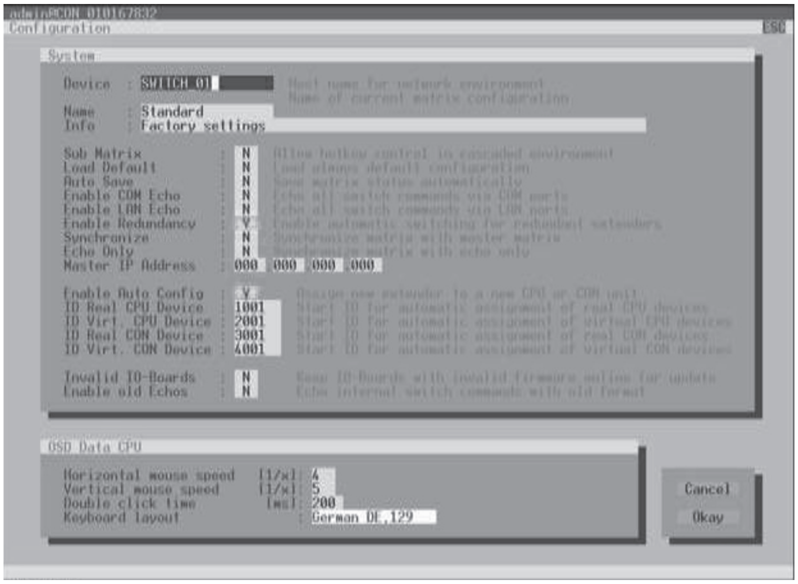
DKM SWITCH API SETTINGS

FIELD	SELECTION	DESCRIPTION
Enable COM Echo	activated	Send all performed switching commands in the matrix as an echo via serial interface. NOTE: This function should be enabled when using a media control via serial interface.
	deactivated	Function not active (default)
Enable LAN Echo	activated	Send all performed switching commands in the matrix as an echo via LAN connection. NOTE: This function should be enabled when using a media control via TCP/IP connection.
	deactivated	Function not active (default)

OSD

Select Configuration > System in the main menu.

NOTE: The serial interface can be blocked while OSD has been opened.



```
admin@CON 010167832
Configuration                                                          ESC

  System
    Device    : SWITCH 01          Host name for network environment
                                   Name of current matrix configuration
    Name      : Standard
    Info      : Factory settings

    Sub Matrix          : N   Allow hotkey control in cascaded environment
    Load Default        : N   Load always default configuration
    Auto Save           : N   Save matrix status automatically
    Enable COM Echo     : N   Echo all switch commands via COM ports
    Enable LAN Echo     : N   Echo all switch commands via LAN ports
    Enable Redundancy   : Y   Enable automatic switching for redundant extenders
    Synchronize         : N   Synchronize matrix with master matrix
    Echo Only           : N   Synchronize matrix with echo only
    Master IP Address   : 000.000.000.000

    Enable Auto Config  : Y      Assign new extender to a new CPU or CON unit
    ID Real CPU Device  : 1001   Start ID for automatic assignment of real CPU devices
    ID Virt. CPU Device : 2001   Start ID for automatic assignment of virtual CPU devices
    ID Real CON Device  : 3001   Start ID for automatic assignment of real CON devices
    ID Virt. CON Device : 4001   Start ID for automatic assignment of virtual CON devices

    Invalid IO-Boards   : N      Keep IO-Boards with invalid firmware online for update
    Enable old Echos    : N      Echo internal switch commands with old format

  OSD Data CPU
    Horizontal mouse speed   [1/x]: 4
    Vertical mouse speed     [1/x]: 5                             Cancel
    Double click time         [ms]: 200
    Keyboard layout               : German DE,129                 Okay
```

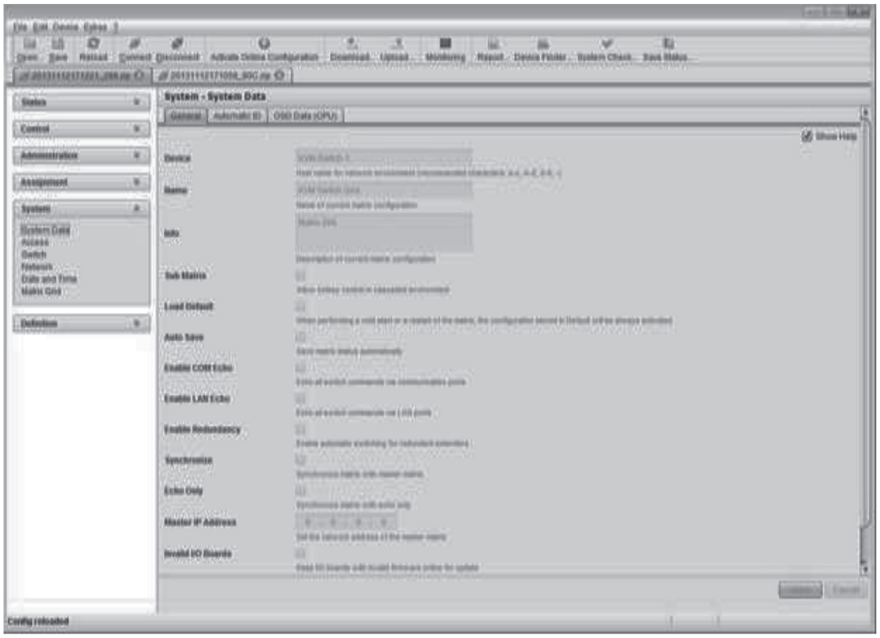
FIGURE 4-1. MENU CONFIGURATION – SYSTEM

You can select between the following buttons:

BUTTON	FUNCTION
Cancel	Reject changes
Save	Save changes

Java Tool

Select System > System Data in the main menu.


File Edit Device Extras ?
Open Save Reload Connect Disconnect Activate Online Configuration Download Upload Monitoring Report Device Finder System Check Save Status
Status
Control
Administration
Assignment
System
System Data
Access
Switch
Network
Date and Time
Matrix Grid
Definition
General
Automatic ID
OSD Data (CPU)
☑ Show Help
Apply
Cancel

System - System Data

Device
Host name for network environment (recommended characters: a-z, A-Z, 0-9, -)

Name
Name of current matrix configuration

Info
Description of current matrix configuration

Sub Matrix □
Allow hotkey control in cascaded environment

Load Default □
When performing a cold start or a restart of the matrix, the configuration stored in Default will be always activated

Auto Save □
Save matrix status automatically

Enable COM Echo □
Echo all switch commands via communication ports

Enable LAN Echo □
Echo all switch commands via LAN ports

Enable Redundancy □
Enable automatic switching for redundant extenders

Synchronize □
Synchronize matrix with master matrix

Echo Only □
Synchronize matrix with echo only

Master IP Address 0 . 0 . 0 . 0
Set the network address of the master matrix

Invalid I/O Boards □
Keep I/O boards with invalid firmware online for update

Config reloaded

FIGURE 4-2. MENU SYSTEM – SYSTEM DATA

4.2.2 NETWORK

The DKM Switch API relevant network configuration is set in this menu.

You have the following possibilities to access the menu:

ACCESS OPTION	SYMBOL
OSD icon	OSD
Java icon	JAVA

You can select between the following DKM Switch API relevant settings:

DKM SWITCH API SETTINGS

FIELD	SELECTION	DESCRIPTION
DHCP	activated	The network settings are automatically supplied by a DNS server (default)
	deactivated	Function not active
IP address	Byte	Input of the IP address in the form “192.168.1.1”, if DHCP is not active
Subnet Mask	Byte	Input of the subnet mask in the form “255.255.255.0”, if DHCP is not active (default: 255.255.255.0)
Gateway	Byte	Input of the subnet mask in the form “192.168.1.1”, if DHCP is not active
Service	activated	LAN interface at the DKM Switch activated for access via Java tool (service port 5555)
	deactivated	Function not active
FTP Server	activated	FTP server for transmission of configuration files activated.
	deactivated	Function not active

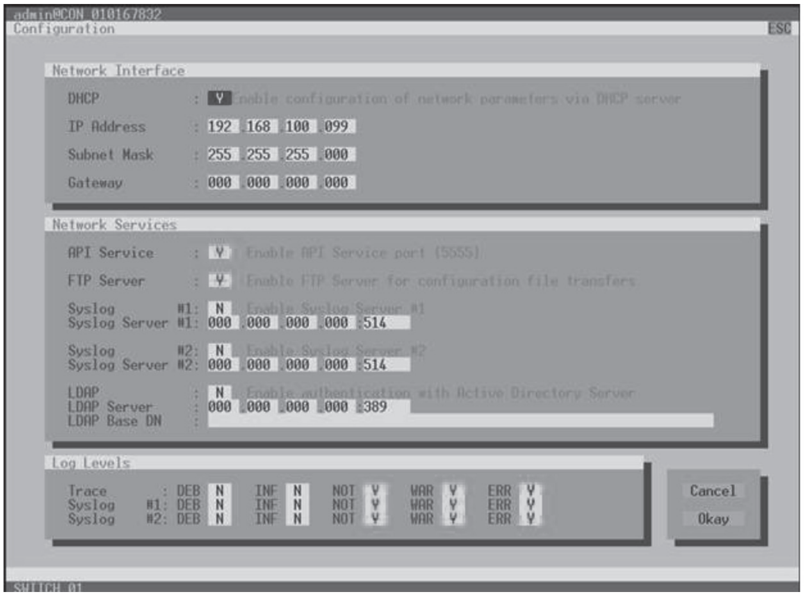
NOTE: Activate the modified network parameters by doing a restart.

CAUTION: Consult your system administrator before modifying the network parameters. Otherwise, unexpected results and failures can occur in combination with the network.

OSD

Select Configuration > Network in the main menu.

NOTE: The serial interface can be blocked while OSD has been opened.



```
admin@CON 010167832
Configuration                                                      ESC

  Network Interface
    DHCP              : Y Enable configuration of network parameters via DHCP server
    IP Address        : 192 .168 .100 .099
    Subnet Mask       : 255 .255 .255 .000
    Gateway           : 000 .000 .000 .000

  Network Services
    API Service       : Y  Enable API Service port (5555)
    FTP Server        : Y  Enable FTP Server for configuration file transfers
    Syslog         #1: N  Enable Syslog Server #1
    Syslog Server  #1: 000 .000 .000 .000 :514
    Syslog         #2: N  Enable Syslog Server #2
    Syslog Server  #2: 000 .000 .000 .000 :514
    LDAP              : N  Enable authentication with Active Directory Server
    LDAP Server       : 000 .000 .000 .000 :389
    LDAP Base DN      :

  Log Levels
    Trace       : DEB N   INF N   NOT Y   WAR Y   ERR Y        Cancel
    Syslog   #1: DEB N   INF N   NOT Y   WAR Y   ERR Y
    Syslog   #2: DEB N   INF N   NOT Y   WAR Y   ERR Y        Okay

SWITCH 01
```

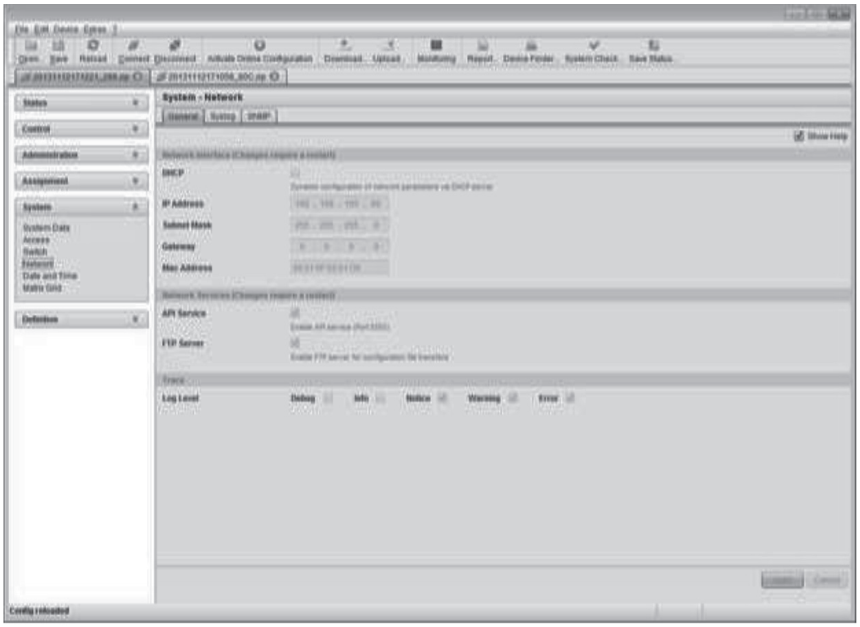
FIGURE 4-3. MENU CONFIGURATION – NETWORK

You can select between the following buttons:

BUTTON	FUNCTION
Cancel	Reject changes
Save	Save changes

Java Tool

Select System > Network in the task area.



Network Interface (Changes require a restart)

DHCP ☐ Dynamic configuration of network parameters via DHCP server

IP Address [illegible]

Subnet Mask [illegible]

Gateway 0 . 0 . 0 . 0

Mac Address [illegible]

Network Services (Changes require a restart)

API Service ☑ Enable API service (Port 5555)

FTP Server ☑ Enable FTP server for configuration file transfers

Trace

Log Level Debug ☐ Info ☐ Notice ☑ Warning ☑ Error ☑

FIGURE 4-4. MENU SYSTEM – NETWORK

4.3 COMMUNICATION SETUP

TCP/IP socket connection

To control the DKM Switch via TCP/IP socket connection the DKM Switch API Service has to be activated. See Network Status and Network in the DKM Switch matrices manual for more information.

NOTE: The DKM Switch matrix includes 16 network sockets. These sockets are kept open for 30 seconds. If there is no keep alive signal in between this period, the socket will be closed again.

Java code example

```
// Create socket connection
Socket socket = new Socket("192.168.100.108", 5555);
final InputStream is = socket.getInputStream();

// Switch off all ports, Command: ESC [ A
final OutputStream os = socket.getOutputStream();
os.write(0x1B); // ESC
os.write(0x5B); // [
os.write(0x41); // A
os.flush();

if (is.read() == 0x06) {
// acknowledged
}

is.close();
os.close();
socket.close();
```

Serial connection

To establish the serial communication to the DKM Switch, set the format for serial data transmission to the following parameters:

115.2K, 8, 1, NO

(115.2 KBAUD, 8 data bits, 1 stop bit, no parity)

4.4 TELEGRAM STRUCTURE

4.4.1 REQUEST

ESC <Server identification><Command> [<Size>, <Data>]

[ ] = Optional elements

4.4.2 RESPONSE

<ACK>, [<ECHO>]

or

ESC <Server identification><Command><Size><Data>

[ ] = Optional elements

<ACK> Acknowledge

<NAK> Negative Acknowledge

<ECHO> reports the matrix sequences solicited by a command and thus the new switching status of the matrix. The echo can be used to update user applications and to operate several matrices in parallel. See System Data in the DKM Switch manual to get more information about Echo Mode.

NOTE: Use the <ECHO> reports to verify that the switch commands have been executed as requested. Update the external switch status according to the <ECHO> reports rather than according to your commands.

4.5 CONSTRAINTS

- Maximum buffer size for data transfer is 8192 bytes.
- 16 sockets for TCP/IP communication over port 5555 are available. Make sure that there will be at least one socket left for the communication with the Java tool.
- Wait for a response before sending another request to the matrix.

5. OPERATION AND SPECIFICATIONS

The DKM Switch can be controlled via an RS-232 serial interface or a TCP/IP socket (port 5555).

TELEGRAM STRUCTURE

TYPE	BYTES	DESCRIPTION
Control character	1	Always: ESC (0x1B)
Server identification	1	Identification of service
Command	1	A special command
Size	2	Optional, if telegram size > 3
Data	n	Optional, n bytes of data

Byte Order: Little Endian

Example: `1012 -> 0xF4` 0x03 (not `0x03 0xF4!`)

(Special) characters:

ACK	0x06
NAK	0x15

Request

ESC <Server identification><Command> [<Size>, <Data>]

[ ] = Optional elements

Response

<ACK> , [<ECHO>]

or

ESC <Server identification><Command><Size><Data>

[ ] = Optional elements

<ECHO> reports the matrix sequences solicited by a command and thus the new switching status of the matrix. The echo can be used to update user applications and to operate several matrices in parallel.

SEQUENCE OF A DATA COMMUNICATION

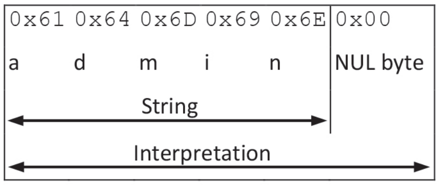
DKM SWITCH MATRIX	CONTROL CPU
–	Sending of a telegram
Acquiring of a command, processing of a command, blocking of further commands	–
a) Errors occurred: <NAK> b) No errors: <ACK> c) Optional: <ECHO> d) Optional: Response telegram with data	–
–	a) Repeat telegram b) Next telegram c) Receive and process response telegram

NOTE: The serial interface can be blocked while the OSD is open.

Strings

NOTE: All strings are NUL-terminated, e.g. the output of names that end with a NUL byte. After the NUL byte, the interpretation must be ended. All subsequent bytes are undefined and must not be interpreted.

Example:



5.1 SYSTEM REQUESTS

5.1.1 GET SYSTEM TIME

Request

Telegram

```
ESC ( S
```

General Description

Get system time

GET SYSTEM TIME

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
(	1	Server identification	0x28
S	1	Command	0x53

Example

Get system time

```
0x1B 0x28 0x53
```

Response

Telegram

```
ESC ) S Size Seconds Minutes Hours Day Date Month Year
```

General Description

Return system time

RETURN SYSTEM TIME

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
)	1	Server identification	0x29
S	1	Command	0x53
Size	2	Total length of telegram (12 Bytes)	0x0C 0x00
Seconds	1	Seconds (0 – 59)	0x00 – 0x59
Minutes	1	Minutes (0 – 59)	0x00 – 0x59
Hours	1	Hours (0 – 23)	0x00 – 0x23
Day	1	Day (1 – 7, Monday = 1)	0x01 – 0x07
Date	1	Date (1- 31)	0x01 – 0x31
Month	1	Month (1 – 12)	0x01 – 0x12
Year	1	Year (+2000)	e.g. 2012 = 0x12

Example

Return system time: Saturday 15:27:48 28.01.2012

```
0x1B 0x29 0x53 0x0C 0x00 0x48 0x27 0x15 0x06 0x28 0x01 0x12
```

NOTE: The system is encoded in the BCD format.

5.1.2 GET SYSTEM STATUS

Request

Telegram

```
ESC [ z
```

General Description

Get system status

GET SYSTEM TIME

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
(	1	Server identification	0x5B
z	1	Command	0x7A

Example

Get system status

```
0x1B 0x5B 0x7A
```

Response

Telegram

```
ESC ] z CRC bitset
```

General Description

Return system status

GET SYSTEM STATUS

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
]	1	Server identification	0x5D
z	1	Command	0x7A
Size	2	Total length of telegram (25 Bytes)	0x19 0x00
Bitset 1	1	Bit 00: taskMAIN active Bit 01: LANAPI active Bit 02: LANGRID active Bit 03: taskSWITCH active Bit 04: taskSYNC active Bit 05: taskUART active Bit 06: taskINT active Bit 07: taskOSD1 active	0x00 – 0xFF 00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000

GET SYSTEM STATUS (CONTINUED)

TYPE	BYTES	DESCRIPTION	HEX CODING
Bitset 2	1	Bit 08: Switch IC active Bit 09: Switch over-temp. Bit 10: Grid active Bit 11: Grid Master Bit 12: 576er Master CPU Bit 13: 576er Slave CPU Bit 14: Redundancy primary CPU Bit 15: Redundancy secondary CPU	0x00 – 0xFF 00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000
Bitset 3	1	Bit 16: PSU 1 available Bit 17: PSU 1 voltage ok Bit 18: PSU 1 error Bit 19: PSU 2 available Bit 20: PSU 2 voltage ok Bit 21: PSU 2 error Bit 22: PSU 3 available Bit 23: PSU 3 voltage ok	0x00 – 0xFF 00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000
Bitset 4	1	Bit 24: PSU 3 error Bit 25: PSU 4 available Bit 26: PSU 4 voltage ok Bit 27: PSU 4 error Bit 28: Fan 1 ok Bit 29: Fan 1 error Bit 30: Fan 2 ok Bit 31: Fan 2 error	0x00 – 0xFF 00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000
Bitset 5	1	Not in use	0x00
Bitset 6	1	Not in use	0x00
Bitset 7	1	Not in use	0x00
Bitset 8	1	Not in use	0x00
Bitset 9	4	GLActive Total number of Grid Lines in the system	e.g. 8 Grid Lines 0x08 0x00 0x00 0x00
Bitset 10	4	GLBusy	Number of Grid Lines in use e.g. 2 Grid Lines 0x02 0x00 0x00 0x00
Bitset 11	4	GLFree Number of unused Grid Lines [GLActive – GLBusy]	e.g. 6 Grid Lines 0x06 0x00 0x00 0x00

Example

Return system status

```
0x1B 0x5D 0x7A 0x19 0x00 0x2D (00101101) 0x0D (00001101)
0x00 (00000000) 0x00 (00000000) 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
```

NOTE: The system status is encoded in the CRC format.

5.2 SWITCH COMMANDS

5.2.1 SWITCH OFF ALL PORTS

Request

Telegram

```
ESC [ A
```

General Description

Switch off all ports

SWITCH OFF ALL PORTS

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
A	1	Command	0x41

Example

Switch off all ports

```
0x1B 0x5B 0x41
```

Response

<ACK> [<ECHO>] or <NAK>.

[ ] = Optional elements

5.2.2 GET CPU DEVICE CONNECTED TO CON DEVICE

Request

Telegram

`ESC [ H Size ConID`

General Description

Get CPU device (input) connected to CON device (output)

GET CPU DEVICE CONNECTED TO CON DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
H	1	Command	0x48
Size	2	Total length of telegram (7 Bytes)	0x07 0x00
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B

Example

Get CPU device connected to CON device (`ConID = 3017`)

`0x1B 0x5B 0x48 0x07 0x00 0xC9 0x0B`

Response

Telegram

`ESC ] H Size ConID CpuID`

General Description

Return CPU device (input) connected to CON device (output)

RETURN CPU DEVICE CONNECTED TO CON DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
	1	Server identification	0x5D
H	1	Command	0x48
Size	2	Total length of telegram (9 Bytes)	0x09 0x00
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B
CpuID	2	ID of CPU device	e.g. 1012 = 0xF4 0x03

Example

Return CPU device (`CpuID = 1012`) connected to CON device (`ConID = 3017`)

`0x1B 0x5D 0x48 0x09 0x00 0xC9 0x0B 0xF3 0x03`

or <NAK>

5.2.3 SET CONNECTION OF CPU DEVICE TO CON DEVICE

Request

Telegram

`ESC [ I Size ConID CpuID`

General Description

Set CPU device connection (input) to CON device (output)

Input data of CPU device (Video, USB, Audio, …) will be transmitted to CON device

SET CONNECTION OF CPU DEVICE TO CON DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
I	1	Command	0x49
Size	2	Total length of telegram (9 Bytes)	0x09 0x00
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B
CpuID	2	ID of CPU device	e.g. 1012 = 0xF4 0x03

Example

Set CPU device (`CpuID = 1012`) connection to CON device (`ConID = 3017`)

`0x1B 0x5B 0x49 0x09 0x00 0xC9 0x0B 0xF4 0x03`

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.4 GET CPU DEVICES CONNECTED TO CON DEVICES

Request

Telegram

`ESC [ J Size ConCnt ConID[1] … ConID[ConCnt]`

General Description

Get CPU devices (input) connected to CON device (output)

For `ConCnt = 0`, all CON devices will be returned

GET CPU DEVICES CONNECTED TO CON DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
[	1	Server identification	`0x5B`
J	1	Command	`0x4A`
Size	2	Total length of telegram (7 Bytes + data)	e.g. for `ConCnt = 3` `0x0D 0x00`
ConCnt	2	Number of CON devices	e.g. `3 = 0x03 0x00`
CpuID	2	ID of CON device	e.g. `3017 = 0xC9 0x0B`

Example

Return CPU devices connected to CON devices (`ConID = 3017, 3028, 3040`)

```
0x1B 0x5B 0x4A 0x0D 0x00 0x03 0x00 0xC9 0x0B 0xD4 0x0B 0xE0 0x0B
```

Response

Telegram

```
ESC ] J Size ConCnt <ConID, CpuID>[1] …<ConID, CpuID>[ConCnt]
```

General Description

Get CPU devices (input) connected to CON devices (output).

Returns a list of pairs of ConID, CpuID

GET CPU DEVICES CONNECTED TO CON DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
]	1	Server identification	`0x5D`
J	1	Command	`0x4A`
Size	2	Total length of telegram (7 Bytes + data)	e.g. for `ConCnt = 3` `0x13 0x00`
ConCnt	2	Number of CON devices	e.g. `3 = 0x03 0x00`
ConID	2	ID of CON device	e.g. `3017 = 0xC9 0x0B`
CpuID	2	ID of CPU device	e.g. `1012 = 0xF4 0x03`

Example

Get CPU devices connected to CON devices

```
CpuID[1] = 1012, ConID[1] = 3017;
CpuID[2] = 1013, ConID[2] = 3028;
CpuID[3] = 1020, ConID[3] = 3040;
0x1B 0x5D 0x4A 0x13 0x00 0x03 0x00 0xC9 0x0B 0xF4 0x03
0xD4 0x0B 0xF5 0x03 0x0E 0x0B 0xFC 0x03
```

or <NAK>

5.2.5 SET CONNECTIONS OF CPU DEVICES TO CON DEVICES

Request

Telegram

```
ESC [ K Size ConCnt <ConID, CpuID>[1] …<ConID, CpuID>[ConCnt]
```

General Description

Set connections of CPU devices (input) to CON devices (output).

Data of CPU (Video, USB, Audio, …) will be transmitted to CON device

SET CONNECTIONS OF CPU DEVICES TO CON DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
K	1	Command	0x4B
Size	2	Total length of telegram (7 Bytes + data)	e.g. for ConCnt = 3 0x13 0x00
ConCnt	2	Number of CONs	e.g. 3 = 0x03 0x00
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B
CpuID	2	ID of CPU device	e.g. 1012 = 0xF4 0x03

Example

Set connections of CPU devices to CON devices

```
ConID[1] = 3017, CpuID[1] = 1012;
ConID[2] = 3028, CpuID[2] = 3013;
ConID[3] = 3040, CpuID[3] = 1020;
0x1B 0x5B 0x4B 0x13 0x00 0x03 0x00 0xC9 0x0B 0xF4 0x03
0xD4 0x0B 0xF5 0x03 0x0E 0x0B 0xFC 0x03
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.6 GET CON DEVICE CONNECTED TO CPU DEVICE

Request

Telegram

`ESC [ L Size CpuID`

General Description

Get CON device (input) connected to CPU device (output)

GET CON DEVICE CONNECTED TO CPU DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
L	1	Command	0x4C
Size	2	Total length of telegram (7 Bytes)	0x07 0x00
CpuID	2	ID of CPU device	e.g. 1012 = 0xF4 0x03

Example

Get CON device connected to CPU device (`CpuID = 1012`)

`0x1B 0x5B 0x4C 0x07 0x00 0xF4 0x03`

Response

Telegram

`ESC ] L Size CpuID ConID`

General Description

Return CON device (input) connected to CPU device (output)

RETURN CON DEVICE (INPUT) CONNECTED TO CPU DEVICE (OUTPUT)

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
]	1	Server identification	0x5D
L	1	Command	0x4C
Size	2	Total length of telegram (9 Bytes)	0x09 0x00
CpuID	2	ID of CPU device	e.g. 1012 = 0xF4 0x03
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B

Example

Return CON device (`ConID = 3017`) connected to CPU device (`CpuID = 1012`)

```
0x1B 0x5D 0x4C 0x09 0x00 0xF4 0x03 0xC9 0x0B
```

or <NAK>

5.2.7 SET CONNECTION OF CON DEVICE TO CPU DEVICE

Request

Telegram

```
ESC [ M Size CpuID ConID
```

General Description

Set CON device (input) connection to CPU device (output)

Input data of CON device (USB, Audio) will be transmitted to CPU device

GET CON DEVICE CONNECTED TO CPU DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
M	1	Command	0x4D
Size	2	Total length of telegram (9 Bytes)	0x09 0x00
CpuID	2	ID of CPU device	e.g. 1012 = 0xF4 0x03
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B

Example

Set CON device (`ConID = 3017`) connection to CPU device (`CpuID = 1012`)

```
0x1B 0x5B 0x4D 0x09 0x00 0xF4 0x03 0xC9 0x0B
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.8 GET CON DEVICES CONNECTED TO CPU DEVICES

Request

Telegram

```
ESC [ N Size CpuCnt CpuID[1] … CpuID[CpuCnt]
```

General Description

Get CON devices (input) connected to CPU devices (output).

For CpuCnt = 0, all CPU devices will be returned

GET CON DEVICES CONNECTED TO CPU DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
N	1	Command	0x4E
Size	2	Total length of telegram (7 Bytes + data)	e.g. for CpuCnt = 3 0x0D 0x00
CpuCnt	2	Number of CPU devices	e.g. 3 = 0x03 0x00
CpuID	2	ID of CPU device	e. g. 1012 = 0xF4 0x03

Example

Get CON devices connected to CPU devices (CpuID = 1012, 1013, 1020)

```
0x1B 0x5B 0x4E 0x0D 0x00 0x03 0x00 0xF4 0x03 0xF5 0x03 0xFC 0x03
```

Response

Telegram

```
ESC ] N Size CpuCnt <CpuID, ConID>[1] … <CpuID, ConID>[CpuCnt]
```

General Description

Return CON devices (input) connected to CPU devices (output).

Returns a list of pairs of CpuID, ConID

CHAPTER 5: OPERATION AND SPECIFICATIONS

GET CON DEVICES CONNECTED TO CPU DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
]	1	Server identification	`0x5D`
N	1	Command	`0x4E`
Size	2	Total length of telegram (7 Bytes + data)	e.g. for `CpuCnt = 3` `0x13 0x00`
CpuCnt	2	Number of CPU devices	e.g. 3 = `0x03 0x00`
CpuID	2	ID of CPU device	e.g. `1012 = 0xF4 0x03`
ConID	2	ID of CON device	e.g. 3017 = `0xC9 0x0B`

Example

Return CON devices connected to CPU devices

```
CpuID[1] = 1012, ConID[1] = 3017;
CpuID[2] = 1013, ConID[2] = 3028;
CpuID[3] = 1020, ConID[3] = 3040;
0x1B 0x5D 0x4E 0x13 0x00 0x03 0x00 0xF4 0x03 0xC9 0x0B
0xF5 0x03 0xD4 0x0B 0xFC 0x03 0x0E 0x0B
```

or <NAK>

5.2.9 SET CONNECTION OF CON DEVICES TO CPU DEVICES

Request

Telegram

```
ESC [ O Size CpuCnt <CpuID, ConID>[1] …<CpuID, ConID>[CpuCnt]
```

General Description

Set connection CON devices (input) to CPU devices (output).

Data of CON device (USB, Audio) will be transmitted to CPU device

SET CONNECTION OF CON DEVICES TO CPU DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
]	1	Server identification	`0x5B`
O	1	Command	`0x4F`
Size	2	Total length of telegram (7 Bytes + data)	e.g. for `CpuCnt = 3` `0x13 0x00`
CpuCnt	2	Number of CPU devices	e.g. `3 = 0x03 0x00`
CpuID	2	ID of CPU device	e.g. `1012 = 0xF4 0x03`
ConID	2	ID of CON device	e.g. `3017 = 0xC9 0x0B`

Example

Set connection of CON devices to CPU devices.

```
CpuID[1] = 1012, ConID[1] = 3017;
CpuID[2] = 1013, ConID[2] = 3028;
CpuID[3] = 1020, ConID[3] = 3040;
0x1B 0x5B 0x4F 0x13 0x00 0x03 0x00 0xF4 0x03 0xC9 0x0B
0xF5 0x03 0xD4 0x0B 0xFC 0x03 0x0E 0x0B
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.10 SET CONNECTION OF CON DEVICE TO CPU DEVICE (BIDIRECTIONAL)

Request

Telegram

```
ESC [ P Size CpuID ConID
```

General Description

Set CON device (input) connection to CPU device (output) and CPU device (input) connection to CON device (output)

Data of CON device (USB, Audio, ...) will be transmitted to CPU device

Data of CPU device (Video, USB, Audio, ...) will be transmitted to CON device

SET CONNECTION OF CON DEVICE TO CPU DEVICE (BIDIRECTIONAL)

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
P	1	Command	0x50
Size	2	Total length of telegram (9 Bytes)	0x09 0x00
CpuID	2	ID of CPU device	e.g. 1012 = 0xF4 0x03
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B

Example

Set CON device (`ConID = 3017`) connection to CPU device (`CpuID = 1012`)

```
0x1B 0x5B 0x50 0x09 0x00 0xF4 0x03 0xC9 0x0B
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.11 SET CONNECTION OF CON DEVICES TO CPU DEVICES (BIDIRECTIONAL)

Request

Telegram

```
ESC [ Q Size Cnt <CpuID, ConID>[1] …<CpuID, ConID>[Cnt]
```

General Description

Set connection of CON devices (input) to CPU devices (output) and CPU devices (input) to CON devices (output)

Data of CON device (USB, Audio, …) will be transmitted to CPU device Data of CPU device (Video, USB, Audio, …) will be transmitted to CON device

SET CONNECTION OF CON DEVICES TO CPU DEVICES (BIDIRECTIONAL)

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
Q	1	Command	0x51
Size	2	Total length of telegram (7 Bytes + data)	e.g. for Cnt = 3 0x13 0x00
Cnt	2	Size of list	e.g. 3 = 0x03 0x00
CpuID	2	ID of CPU device	e.g. 1012 = 0xF4 0x03
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B

Example

Connect CONs with CPUs and CPUs with CONs

```
CpuID[1] = 1012, ConID[1] = 3017;
CpuID[2] = 1013, ConID[2] = 3028;
CpuID[3] = 1020, ConID[3] = 3040;
0x1B 0x5B 0x51 0x13 0x00 0x03 0x00 0xF4 0x03 0xC9 0x0B
0xF5 0x03 0xD4 0x0B 0xFC 0x03 0x0E 0x0B
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.12 GET CONNECTIONS FOR ALL CON DEVICES AND CPU DEVICES

Request

Telegram

```
ESC [ R
```

General Description

Get all CPU device – CON device connections

GET CONNECTIONS FOR ALL CON DEVICES AND CPU DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
R	1	Command	0x52

Example

Get all CPU device – CON device connections

```
0x1B 0x5B 0x52
```

Response

Telegram

```
ESC ] R Size CpuCnt ConCnt <CpuID, ConID>[1] …<CpuID, ConID>[ CpuCnt] <ConID, CpuID>[1] …<ConID, CpuID>[
ConCnt]
```

General Description

Return all CPU device – CON device connections in pairs.

For each defined CPU device, the ConID of the connected CON device will be added, or 0 if the CPU device is disconnected.

For each defined CON device, the CpuID of the connected CPU device will be added, or 0 if the CON device is disconnected.

GET CONNECTIONS FOR ALL CON DEVICES AND CPU DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
[	1	Server identification	`0x5D`
R	1	Command	`0x52`

Size	2	Total length of telegram (9 Bytes + data)	e.g. for `CpuCnt = 3`
			`ConCnt = 2`
			`0x15 0x00`

CpuCnt	2	Number of CPU device	e.g. `3 = 0x03 0x00`
ConCnt	2	Number of CON device	e.g. `2 = 0x02 0x00`
CpuID	2	ID of CPU device	e.g. `1012 = 0xF4 0x03`
ConID	2	ID of CON device	e.g. `3017 = 0xC9 0x0B`

Example

Return all CPU device – CON device connections in pairs

```
CpuID[1] = 1012, ConID[1] = 3017;
CpuID[2] = 1013, ConID[2] = 3028;
CpuID[3] = 1020, ConID[3] = 3040;
ConID[1] = 3017, CpuID[1] = 1012;
ConID[2] = 3028, CpuID[2] = 0;
0x1B 0x5D 0x52 0x15 0x00 0x03 0x00 0x02 0x00 0xF4 0x03
0xC9 0x0B 0xF5 0x03 0xD4 0x0B 0xFC 0x03 0x0E 0x0B 0xC9
0x0B 0xF4 0x03 0xD4 0x0B 0x00 0x00
```

or <NAK>

5.2.13 SET CONNECTION FOR ALL CON DEVICES AND CPU DEVICES

Request

Telegram

```
ESC [ S Size CpuCnt ConCnt <CpuID, ConID>[1] …<CpuID, ConID>[ CpuCnt] <ConID, CpuID>[1] …<ConID, CpuID>[
ConCnt]
```

General Description

Set a connection for all defined CON devices and CPU devices.

For each defined CPU device add the ConID, or 0 if the CPU device is disconnected.

For each defined CON device add the CpuID, or 0 if the CON device is disconnected.

SET CONNECTIONS FOR ALL CON DEVICES AND CPU DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
[	1	Server identification	`0x5B`
S	1	Command	`0x53`

Size	2	Total length of telegram (9 Bytes + data)	e.g. for
			`CpuCnt = 3`
			`ConCnt = 2`
			`0x15 0x00`

CpuCnt	2	Number of CPUs	e.g. `3 = 0x03 0x00`
ConCnt	2	Number of CONs	e.g. 2 = `0x02 0x00`
CpuID	2	ID of CPU device	e.g. `1012 = 0xF4 0x03`
ConID	2	ID of CON device	e.g. 3017 = `0xC9 0x0B`

Example

Set a connection for all defined CON devices and CPU devices

```
CpuID[1] = 1012, ConID[1] = 3017;
CpuID[2] = 1013, ConID[2] = 3028;
CpuID[3] = 1020, ConID[3] = 3040;
ConID[1] = 3017, CpuID[1] = 1012;
ConID[2] = 3028, CpuID[2] = 0;
0x1B 0x5B 0x53 0x15 0x00 0x03 0x00 0x02 0x00 0xF4 0x03
0xC9 0x0B 0xF5 0x03 0xD4 0x0B 0xFC 0x03 0x0E 0x0B 0xC9
0x0B 0xF4 0x03 0xD4 0x0B 0x00 0x00
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.14 SET EXTENDED CONNECTION

Request

Telegram

```
ESC [ b Size CpuID ConID Mode
```

General Description

Set CON device (input) connection to CPU device (output) and CPU device (input) connection to CON device (output)

Data of CON device (USB, Audio, ...) is transmitted to a CPU device

Data of CPU device (Video, USB, Audio, ...) is transmitted to a CON device

SET EXTENDED CONNECTION

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
b	1	Command	0x62
Size	2	Total length of telegram	0x0B 0x00
CpuID	2	ID of a CPU device	e.g. 1012 = 0xF4 0x03
ConID	2	Connection mode (0 = full access, 1 = video only, 2 = private mode)	0 = 0x00 0x00 1 = 0x01 0x00 2 = 0x02 0x00

Example

Set CON device connection to CPU device and CPU device connection to CON device

`CpuID = 1012`, `ConID = 3017`, Mode = private mode

```
0x1B 0x5B 0x62 0x0B 0x00 0xF4 0x03 0xC9 0x0B 0x02 0x00
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.15 SET CONNECTION OF CON DEVICE TO CPU DEVICE (BIDIRECTIONAL, PORT MODE)

Request

Telegram

```
ESC [ C Size ConPort CpuPort
```

General Description

Set connection of CON port (input) to CPU port (output) and connection of CPU port (input) to CON port (output).

Data of CON device (USB, Audio, …) will be transmitted to CPU device.

Data of CPU device (Video, USB, Audio, …) will be transmitted to CON device.

SET CONNECTIONS OF CON DEVICE TO CPU DEVICE (BIDIRECTIONAL, PORT MODE)

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
[	1	Server identification	`0x5B`
C	1	Command	`0x43`
Size	2	Total length of telegram (9 bytes)	`0x09 0x00`
ConPort	2	Port number of CON device	`e.g. 10 = 0x0A 0x00`
CpuPort	2	Port number of CPU device	`e.g. 20 = 0x14 0x00`

Example

Set CON port (ConPort = 10) connection to CPU port (CpuPort = 20)

```
0x1B 0x5B 0x43 0x09 0x00 0x0A 0x00 0x14 0x00
```

Response

<ACK> [<ECHO>] or <NAK

.[ ] = Optional elements

5.2.16 SET CONNECTION OF CON DEVICE TO CPU DEVICE (PORT MODE)

Request

Telegram

```
ESC [ F Size ConPort CpuPort
```

General Description

Set connection of CPU port (input) to CON port (output).

Data of CPU device (Video, USB, Audio, ...) will be transmitted to CON device.

SET CONNECTION OF CON DEVICE TO CPU DEVICE (PORT MODE)

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
[	1	Server identification	`0x5B`
F	1	Command	`0x46`
Size	2	Total length of telegram (9 bytes)	`0x09 0x00`
ConPort	2	Port number of CON device	e.g. `10 = 0x0A 0x00`
CpuPort	2	Port number of CPU device	e.g. `20 = 0x14 0x00`

Example

Set CON port (ConPort = 10) connection to CPU port (CpuPort = 20)

```
0x1B 0x5B 0x46 0x09 0x00 0x0A 0x00 0x14 0x00
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.17 SET CONNECTION OF CPU DEVICES TO CON DEVICES (PORT MODE)

Request

Telegram

```
ESC [ G Size CpuCnt <ConPort, CpuPort>[1] …<ConPort, CpuPort>[CpuCnt]
```

General Description

Set connection of CPU port (input) to CON port (output).

Data of CPU device (Video, USB, Audio, …) will be transmitted to CON device.

SET CONNECTION OF CPU DEVICES TO CON DEVICES (PORT MODE)

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
G	1	Command	0x47
Size	2	Total length of telegram (19 bytes)	0x13 0x00
ConCnt	2	Number of CON devices	e.g. 3 = 0x03 0x00
ConPort	2	Port number of CON device	e.g. 10 = 0x0A 0x00
CpuPort	2	Port number of CPU device	e.g. 20 = 0x14 0x00

Example

Set CON port connections to CPU ports

```
ConPort[1] = 5, CpuPort[1] = 3;
ConPort[2] = 2, CpuPort[2] = 6;
ConPort[3] = 4, CpuPort[3] = 7;
0x1B 0x5B 0x47 0x13 0x00 0x03 0x00 0x05 0x00 0x03 0x00
0x02 0x00 0x06 0x00 0x04 0x00 0x07 0x00
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

CHAPTER 5: OPERATION AND SPECIFICATIONS

5.2.18 GET CPU DEVICE CONNECTED TO CON DEVICE (PORT MODE)

Request

Telegram

```
ESC [ B Size ConID
```

General Description

Get port of CPU device (input) connected to CON device (output).

GET CPU DEVICE CONNECTED TO CON DEVICE (PORT MODE)

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
B	1	Command	0x42
Size	2	Total length of telegram (7 bytes)	0x07 0x00
ConID	2	ID of CON device	e.g. 5 = 0x05 0x00

Example

Get CPU device connected to CON device (ConID = 3017)

```
0x1B 0x5B 0x42 0x07 0x00 0x05 0x00
```

5.2.19 GET CPU DEVICES CONNECTED TO CON DEVICES (PORT MODE)

Request

Telegram

```
ESC [ D Size ConCnt ConID[1] … ConID[ConCnt]
```

General Description

Get CPU devices (input) connected to CON device (output).

For ConCnt = 0, all CON devices will be returned.

GET CPU DEVICES CONNECTED TO CON DEVICES (PORT MODE)

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
D	1	Command	0x44
Size	2	Total length of telegram (13 bytes)	e.g. for ConCnt = 3 0x0D 0x00
ConCnt	2	Number of CON devices	e.g.3 = 0x03 0x00
ConID	2	ID of CON device	e.g. 2 = 0x02 0x00

[

Example

Return ports of CPU devices connected to CON devices (ConPort = 2, 4, 5)

```
0x1B 0x5B 0x44 0x0D 0x00 0x03 0x00 0x02 0x00 0x04 0x00 0x05 0x00
```

5.2.20 SET LOCAL CPU CONNECTION

Request

Telegram

```
ESC [ f Size ConID KVM
```

General Description

Set CON device (input) connection to local CPU device (output).

0x1B 0x5B 0x66 0x09 0x00 0xC9 0x0B 0x03 0x00

Response

<ACK> [<ECHO>] or <NAK>.

[ ] = Optional elements

SET LOCAL CPU CONNECTION

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
[	1	Command	0x66
Size	2	Total length of telegram (9 bytes)	0x09 0x00
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B
KVM	2	Connection mode	Primary port: 1 = 0x01 0x00 Secondary port: 2 = 0x02 0x00 Local CPU: 3 = 0x03 0x00

[

Example

Set CON device connection to local CPU

```
0x1B 0x5B 0x66 0x09 0x00 0xC9 0x0B 0x03 0x00
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.21 SET CONNECTION OF SINGLE CPU EXTENDERS TO SINGLE CON EXTENDERS IN MULTI-HEAD DEVICES

Request

Telegram

```
ESC [ l Size CpuID ConID
```

General Description

Set CPU extender connection (input) to CON extender (output)

Input data of CPU extender (Video, USB, Audio, ...) will be transmitted to CON extender

SET CONNECTION OF SINGLE CPU EXTENDERS TO SINGLE CON EXTENDERS IN MULTI-HEAD DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
[	1	Command	0x6C
Size	2	Total length of telegram (13 bytes)	0x0D 0x00
CpuID	2	ID of CPU device	e.g. 1012 = 0xF4 0x03
CpuExt	2	Extender number of CPU device	e.g. 4 = 0x04 0x00
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B
ConExt	2	Extender number of CON device	e.g. 2 = 0x02 0x00

[

Example

Set CPU extender connection (CpuExt = 4) of CPU device (CpuID = 1012) to CON extender (ConExt = 2) of CON device (ConID = 3017).

0x1B 0x5B 0x6C 0x0D 0x00 0xF4 0x03 0x04 0x00 0xC9 0x00 0x02 0x00

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.22 EXECUTE CON/USER MACRO

Request

Telegram

```
ESC [ o Size Key KeyUser KeyCon ConID
```

General Description

Execute a CON or user macro.

EXECUTE CON/USER MACRO

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
[	1	Server identification	`0x5B`
p	1	Command	`0x6F`
Size	2	Total length of telegram (13 bytes)	`0x0D 0x00`
Key	2	Macro key (F1 – F32)	e.g. `F1 = 0x01 0x00`

KeyUser	2	Enable user macro (User ID from matrix)	Enable for UserID = 5
			`0x05 0x00,`
			disable
			`0x00 0x00`

ConID for CON macro	2	Enable CON macro for executing the CON macro	e.g. `3017 = 0xC9 0x0B` disable `0x00 0x00`
ConID for user macro	2	ID of CON device for executing the user macro	e.g. `3017 = 0xC9 0x0B` disable `0x00 0x00`

Example

Execute user macro F3 (`UserID = 5`) at CON device (`ConID = 3017`)

```
0x1B 0x5B 0x6F 0x0D 0x00 0x03 0x00 0x05 0x00 0x00 0x00 0xC9 0x0B
```

Execute CON macro F3 at CON device (ConID = 3017)

```
0x1B 0x5B 0x6F 0x0D 0x00 0x03 0x00 0x00 0x00 0xC9 0x0B 0x00 0x00
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.2.23 GET CPU LIST

Request

Telegram

```
ESC [ g Size First
```

General Description

Get list of all CPU devices (output) including ID, name and online status

First: Index of CPU device from which the list scan will start

GET CPU LIST

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
g	1	Command	0x67
Size	2	Total length of telegram (7 Bytes)	0x07 0x00
First	2	Index of first CPU	e.g. 3 = 0x03 0x00 0 (all) = 0x00 0x00

Example

Get all CPUs

```
0x1B 0x5B 0x67 0x07 0x00 0x00 0x00
```

Response

Telegram

```
ESC ] g Size Count Next List [1] … List [Count]
```

General Description

Count: Number of items in the CPU list

Next: Index of the next CPU, if the list of CPU devices exceeds the telegram size. Contains 0 if there are no more CPU devices.

GET CPU LIST

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5D
g	1	Command	0x67
Size	2	Total length of telegram	e.g. 33 = 0x21 0x00
Count	2	Number of CPUs	e.g. 1 = 0x01 0x00
Next	2	Index of first CPU in next list	e.g. 0 = 0x00 0x00 (no further CPU)
Id	4	ID of CPU device	e.g. 1000 = 0xE8 0x03 0x00 0x00
Name	17	Name of CPU	e.g. CPU _ Video1 = 0x43 0x50 0x55 0x5F 0x56 0x69 0x64 0x65 0x6F 0x31 0x00
Status	1	Status of CPU device	= 0x00 = offline ≠ 0x00 = online
Empty	2	Empty bytes	0x00 0x00

Example

Return list of CPUs

```
0x1B 0x5D 0x67 0x21 0x00 0x01 0x00 0x00 0x00 0xE8 0x03
0x00 0x00 0x43 0x50 0x55 0x5F 0x56 0x69 0x64 0x65 0x6F
0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x01 0x00 0x00
```

or <NAK>

5.2.24 GET CON LIST

Request

Telegram

```
ESC [ h Size First
```

General Description

Get list of all CON devices (input) including ID, name, online status, and user

First: Index of CON device from which the list scan will start

GET CON LIST

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
h	1	Command	0x68
Size	2	Total length of telegram (7 Bytes)	0x07 0x00
First	2	Index of first CON	e.g. 5 = 0x05 0x00 0 (all) = 0x00 0x00

Example

Get all CONs

```
0x1B 0x5B 0x68 0x07 0x00 0x00 0x00
```

Response

Telegram

```
ESC ] h Size Count Next List [1] … List [Count]
```

General Description

Count: Number of items in the CON list

Next: Index of the next CON, if the list of CON devices exceeds the telegram size. Contains 0, if there are no more CON devices.

GET CON LIST

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5D
h	1	Command	0x68
Size	2	Total length of telegram	e.g. 33 = 0x21 0x00
Count	2	Number of CONs	e.g. 1 = 0x01 0x00
Next	2	Index of first CON in next list	e.g. 0 = 0x00 0x00 (no further CON)
Id	4	ID of CON device	e.g. 3000 = 0xB8 0x0B 0x00 0x00
Name	17	Name of CON	e.g. CON _ Video1 = 0x43 0x4F 0x4E 0x5F 0x56 0x69 0x64 0x65 0x6F 0x31 0x00
Status	1	Status of CON device	= 0x00 = offline ≠ 0x00 = online
Info	2	Info about logged in user	e.g. user with ID 1 1 = 0x01 0x00

Example

Return list of CONs

```
0x1B 0x5D 0x68 0x21 0x00 0x01 0x00 0x00 0x00 0xB8 0x0B
0x00 0x00 0x43 0x4F 0x4E 0x5F 0x56 0x69 0x64 0x65 0x6F
0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x01 0x01 0x00
```

or <NAK>

5.2.25 GET USER LIST

Request

Telegram

```
ESC [ i Size First
```

General Description

Get list of all users (output) including ID and name

First: Index of the user from whom the list scan will start

GET USER LIST

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5D
i	1	Command	0x69
Size	2	Total length of telegram (7 Bytes)	0x07 0x00
First	2	Index of first user	e.g. 1 = 0x01 0x00 0 (all) = 0x00 0x00

Example

Get all users

```
0x1B 0x5B 0x69 0x07 0x00 0x00 0x00
```

Response

Telegram

```
ESC ] i Size Count Next List [1] … List [Count]
```

General Description

Count: Number of items in the user list

Next: Index of the next user, if the list of users exceeds the telegram size. Contains 0 if there are no more users.

GET USER LIST

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5D
i	1	Command	0x69
Size	2	Total length of telegram	e.g. 33 = 0x21 0x00
Count	2	Number of users	e.g. 1 = 0x01 0x00
Next	2	Index of first user in next list	e.g. 0 = 0x00 0x00 (no further user)
Id	4	ID of user	e.g. 1 = 0x01 0x00 0x00 0x00
Name	20	Name of user	e.g. admin = 0x61 0x64 0x6D 0x69 0x6E 0x00

Example

Return list of users

```
0x1B 0x5D 0x69 0x21 0x00 0x01 0x00 0x00 0x00 0x01 0x00
0x00 0x00 0x61 0x64 0x6D 0x69 0x6E 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
```

or <NAK>

5.2.26 GET CON LINK STATUS

Request

Telegram

```
ESC [ m Size ConID
```

General Description

Get link status of CON device

GET CON LINK STATUS

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
m	1	Command	0x6D
Size	2	Total length of telegram (7 Bytes)	0x07 0x00
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B

Example

Get link status of CON device (ConID = 3001)

```
0x1B 0x5B 0x6D 0x07 0x00 0xB9 0x00
```

Response

Telegram

```
ESC ] m Size ConID Status
```

General Description

Status: Link status of extender

GET CON LINK STATUS

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5D
m	1	Command	0x6D
Size	2	Total length of telegram (9 Bytes)	0x09 0x00
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B
Status	2	Status of extender link	Primary link: 0x01 0x00 Secondary link: 0x02 0x00 Local CPU: 0x03 0x00

Example

Return extender link for CON device connected via link 1

```
0x1B 0x5D 0x6D 0x07 0x00 0xB9 0x00 0x01 0x00
```

or <NAK>

NOTE: Only the first extender of a CON device can be queried.

5.2.27 GET CON LINK STATUS LIST

Request

Telegram

`ESC [ t Size First`

General Description

Get active link status list for all CON devices

GET CON LINK STATUS LIST

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
[	1	Server identification	`0x5B`
t	1	Command	`0x74`
Size	2	Total length of telegram (7 Bytes)	`0x07 0x00`
First	2	Index of first CON	e.g. `5 = 0x05 0x00` `0 (all) = 0x00 0x00`

Example

Get link status of CON list (First = 0)

`0x1B 0x5B 0x74 0x07 0x00 0x00 0x00`

Response

Telegram

`ESC ] h Size Count Next`

General Description

Return active link status for all CON devices

Count: Number of items in the CON list

Next: Index of the next CON, if the list of CON devices exceeds the telegram size. Contains 0, if there are no more CON devices.

GET CON LINK STATUS LIST

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
[	1	Server identification	`0x5D`
h	1	Command	`0x74`
Size	2	Total length of telegram	e.g. `17 = 0x11 0x00`
Count	2	Number of CONs	e.g. `2 = 0x02 0x00`
Next	2	ID of first CON in next list (max. 256 entries per static list)	e.g. `0 = 0x00 0x00`
ConID	2	ID of CON device	e.g. `3017 = 0xC9 0x0B`
Status	2	Status of extender link	Primary link: `0x01 0x00` Secondary link: `0x02 0x00` Local CPU: `0x03 0x00`

Example

Return extender link for CON devices (3017 & 3018) connected via link 1

```
0x1B 0x5D 0x74 0x11 0x00 0x02 0x00 0x00 0x00 0xC9 0x0B 0x01 0x00 0x0B 0xCA 0x01 0x00
```

or <NAK>

NOTE: Only the first extender of a CON device can be queried.

5.3 ASSIGNMENTS

5.3.1 GET VIRTUAL CON DEVICE

Request

Telegram

```
ESC [ T Size RConID
```

General Description

Get virtual CON device of a real CON device

GET VIRTUAL CON DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
T	1	Command	0x54
Size	2	Total length of telegram (7 Bytes)	0x07 0x00
RConID	2	ID of real CON device	e.g. 3017 = 0xC9 0x0B

Example

Get virtual CON device of a real CON device (RConID = 3017)

```
0x1B 0x5B 0x54 0x07 0x00 0xC9 0x0B
```

Response

Telegram

```
ESC ] T Size RConID VConID
```

General Description

Return virtual CON device of a real CON device

GET VIRTUAL CON DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5D
T	1	Command	0x54
Size	2	Total length of telegram (9 Bytes)	0x09 0x00
RConID	2	ID of real CON device	e.g. 3017 = 0xC9 0x0B
VConID	2	ID of virtual CON device	e.g. 4034 = 0xC2 0x0F

Example

Return virtual CON device (VConID = 4034) of a real CON device (RConID = 3017)

```
0x1B 0x5B 0x54 0x09 0x00 0xC9 0x0B 0xC2 0x0F
```

or <NAK>

5.3.2 SET VIRTUAL CON DEVICE TO A REAL CON DEVICE

Request

Telegram

```
ESC [ U Size RConID VConID
```

General Description

Set virtual CON device to a real CON device

SET VIRTUAL CON DEVICE TO A REAL CON DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
U	1	Command	0x55
Size	2	Total length of telegram (9 Bytes)	0x09 0x00
RConID	2	ID of real CON device	e.g. 3017 = 0xC9 0x0B
VConID	2	ID of virtual CON device	e.g. 4034 = 0xC2 0x0F

Example

Set virtual CON device (VConID = 4034) to a real CON device (RConID = 3017)

```
0x1B 0x5B 0x55 0x09 0x00 0xC9 0x0B 0xC2 0x0F
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.3.3 GET REAL CPU DEVICE

Request

Telegram

`ESC [ V Size VCpuID`

General Description

Get real CPU device of a virtual CPU device

GET REAL CPU DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
V	1	Command	0x56
Size	2	Total length of telegram (7 Bytes)	0x07 0x00
VCpuID	2	ID of virtual CPU device	e.g. 2018 = 0xE2 0x07

Example

Get real CPU device of a virtual CPU device (VCpuID = 2018)

`0x1B 0x5B 0x56 0x07 0x00 0xE2 0x07`

Response

Telegram

`ESC ] V Size VCpuID RCpuID`

General Description

Return real CPU device of a virtual CPU device

RETURN REAL CPU DEVICE OF A VIRTUAL CPU DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
V	1	Command	0x56
Size	2	Total length of telegram (7 Bytes)	0x07 0x00
VCpuID	2	ID of virtual CPU device	e.g. 2018 = 0xE2 0x07

RETURN REAL CPU DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5D
V	1	Command	0x56
Size	2	Total length of telegram (9 Bytes)	0x09 0x00
VCpuID	2	ID of virtual CPU device	e.g. 2018 = 0xE2 0x07
RCpuID	2	ID of real CPU device	e.g. 1012 = 0xF4 0x03

Example

Return real CPU device (`RCpuID = 1012`) of a virtual CPU device (`VCpuID = 2018`)

`0x1B 0x5D 0x56 0x09 0x00 0xE2 0x07 0xF4 0x03`

or <NAK>

5.3.4 SET REAL CPU DEVICE TO A VIRTUAL CPU DEVICE

Request

Telegram

`ESC [ W Size VCpuID RCpuID`

General Description

Set real CPU device to a virtual CPU device

SET REAL CPU DEVICE TO A VIRTUAL CPU DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
W	1	Command	0x57
Size	2	Total length of telegram (9 Bytes)	0x09 0x00
VCpuID	2	ID of virtual CPU device	e.g. 2018 = 0xE2 0x07
RCpuID	2	ID of real CPU device	e.g. 1012 = 0xF4 0x03

Example

Set real CPU device (RCpuID = 1012) to a virtual CPU device (VCpuID = 2018)

`0x1B 0x5B 0x57 0x09 0x00 0xE2 0x07 0xF4 0x03`

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.3.5 GET VIRTUAL CON DEVICES

Request

Telegram

`ESC [ X Size ConCnt RConID[1] … RConID[ConCnt]`

General Description

Get virtual CON devices of a real CON devices

For ConCnt = 0, all real CON devices with assignments to virtual CON devices will be returned.

GET VIRTUAL CON DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
X	1	Command	0x58
Size	2	Total length of telegram (7 Bytes + data)	e.g. for ConCnt = 3 0x0D 0x00
ConCnt	2	Number of CON device	e.g. 3 = 0x03 0x00
RConID	2	ID of real CON device	e.g. 3017 = 0xC9 0x0B

Example

Return virtual CON devices of a real CON devices as pairs (RConID = 3017, 3028, 3040)

`0x1B 0x5B 0x58 0x0D 0x00 0x03 0x00 0xC9 0x0B 0xD4 0x0B 0xE0 0x0B`

Response

Telegram

`ESC ] X Size ConCnt <RConID, VConID>[1] …<RConID, VConID>[ConCnt]`

General Description

Return virtual CON devices of real CON devices as pairs

RETURN VIRTUAL CON DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5D
X	1	Command	0x58
Size	2	Total length of telegram (7 Bytes + data)	e.g. for ConCnt = 3 0x13 0x00
ConCnt	2	Number of CON devices	e.g. 3 = 0x03 0x00
RConID	2	ID of real CON device	e.g. 3017 = 0xC9 0x0B
VConID	2	ID of virtual CON device	e.g. 4034 = 0xC2 0x0F

Example

Returns virtual CON of a real CON as pairs

```
RConID[1] = 3017, VConID[1] = 4034;
RConID[2] = 3028, VConID[2] = 4042;
RConID[3] = 3040, VConID[3] = 4045;
0x1B 0x5D 0x58 0x13 0x00 0xC9 0x0B 0xC2 0x0F 0xD4 0x0B 0xCA 0x0F 0xE0 0x0B 0xCD 0x0F
```

or <NAK>

5.3.6 SET VIRTUAL CON DEVICES TO REAL CON DEVICES

Request

Telegram

```
ESC [ Y Size ConCnt <RConID, VConID>[1] …<RConID, VConID>[ConCnt]
```

General Description

Set virtual CON devices to real CON devices

SET VIRTUAL CON DEVICES TO REAL CON DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
Y	1	Command	0x59
Size	2	Total length of telegram (7 Bytes + data)	e.g. for ConCnt = 3 0x13 0x00
ConCnt	2	Number of CON devices	e.g. 3 = 0x03 0x00
RConID	2	ID of real CON device	e.g. 3017 = 0xC9 0x0B
VConID	2	ID of virtual CON device	e.g. 4034 = 0xC2 0x0F

Example

Set virtual CON devices to real CON devices

```
RConID[1] = 3017, VConID[1] = 4034;
RConID[2] = 3028, VConID[2] = 4042;
RConID[3] = 3040, VConID[3] = 4045;
0x1B 0x5B 0x59 0x13 0x00 0xC9 0x0B 0xC2 0x0F 0xD4 0x0B 0xCA 0x0F 0xE0 0x0B 0xCD 0x0F
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.3.7 GET REAL CPU DEVICES

Request

Telegram

`ESC [ Z Size CpuCnt VCpuID[1] … VCpuID[CpuCnt]`

General Description

Get real CPU devices of virtual CPU devices

For CpuCnt = 0, all virtual CPU devices with assignments to virtual CPU devices will be returned.

GET REAL CPU DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
Z	1	Command	0x5A
Size	2	Total length of telegram (7 Bytes + data)	e.g. for ConCnt = 3 0x0D 0x00
CpuCnt	2	Number of CPU devices	e.g. 3 = 0x03 0x00
VCpuID	2	ID of virtual CPU device	e.g. 2018 = 0xE2 0x07

Example

Get real CPU devices of virtual CPU devices (VCpuID = 2018, 2030, 2035)

`0x1B 0x5B 0x5A 0x0D 0x00 0x03 0x00 0xE2 0x07 0xEE 0x07 0xF3 0x07`

Response

Telegram

`ESC ] Z Size CpuCnt <VCpuID, RCpuID>[1] …<VCpuID, RCpuID>[CpuCnt]`

General Description

Return real CPU devices of virtual CPU devices as pairs

GET REAL CPU DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5D
Z	1	Command	0x5A
Size	2	Total length of telegram (7 Bytes + data)	e.g. for CpuCnt = 3 0x13 0x00
CpuCnt	2	Number of CPU devices	e.g. 3 = 0x03 0x00
VCpuID	2	ID of virtual CPU device	e.g. 2018 = 0xE2 0x07
RCpuID	2	ID of real CPU device	e.g. 1012 = 0xF4 0x03

Example

Return real CPU devices of virtual CPU devices as pairs

```
VCpuID[1] = 2018, RCpuID[1] = 1012;
VCpuID[2] = 2030, RCpuID[2] = 1013;
VCpuID[3] = 2035, RCpuID[3] = 1020;
0x1B 0x5D 0x5A 0x13 0x00 0x03 0x00 0xE2 0x07 0xF4 0x03 0xEE 0x07 0xF5 0x03 0xF3 0x07 0xFC 0x03
```

or <NAK>

5.3.8 SET REAL CPU DEVICES

Request

Telegram

```
ESC [ a Size CpuCnt <VCpuID, RCpuID>[1] …<VCpuID, RCpuID>[CpuCnt]
```

General Description

Set real CPU devices to virtual CPU devices

SET REAL CPU DEVICES

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
a	1	Command	0x61
Size	2	Total length of telegram (7 Bytes + data)	e.g. for CpuCnt = 3 0x13 0x00
CpuCnt	2	Number of CPU devices	e.g. 3 = 0x03 0x00
VCpuID	2	ID of virtual CPU device	e.g. 2018 = 0xE2 0x07
RCpuID	2	ID of real CPU device	e.g. 1025 = 0x16 0x04

Example

Set real CPU devices to virtual CPU devices

```
VCpuID[1] = 2018, RCpuID[1] = 1012;
VCpuID[2] = 2030, RCpuID[2] = 1013;
VCpuID[3] = 2035, RCpuID[3] = 1020;
0x1B 0x5B 0x61 0x13 0x00 0x03 0x00 0xE2 0x07 0xF4 0x03
0xEE 0x07 0xF5 0x03 0xF3 0x07 0xFC 0x03
```

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.3.9 SET REAL CPU DEVICE TO A CPU GROUP

Request

Telegram

`ESC [ q Size RCpuID GCpuID`

General Description

Set real CPU device to a CPU group

SET REAL CPU DEVICE TO A CPU GROUP

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
[	1	Server identification	`0x5B`
q	1	Command	`0x71`
Size	2	Total length of telegram (9 Bytes)	`0x09 0x00`
RCpuID	2	ID of real CPU device	e.g. `1006 = 0xEE 0x03`
GCpuID	2	ID of CPU group	e.g. `2003 = 0xD3 0x07`

Example

Set real CPU device (RCpuID = 1006) to CPU group (GCpuID = 2003)

`0x1B 0x5B 0x71 0x09 0x00 0xEE 0x03 0xD3 0x07`

Delete the CPU group assignment of a real CPU device (RCpuID = 1006) with GCpuID = 0

`0x1B 0x5B 0x71 0x09 0x00 0xEE 0x03 0x00 0x00`

Remove all real CPU devices from a CPU group (GCpuID = 2003) with RCpuID = 0

`0x1B 0x5B 0x71 0x09 0x00 0x00 0x00 0xD3 0x07`

Response

<ACK> [<ECHO>] or <NAK>

[ ] = Optional elements

5.3.10 GET CPU GROUP OF A REAL CPU DEVICE

Request

Telegram

`ESC [ p Size RCpuID`

General Description

Get CPU group of real CPU device

GET CPU GROUP OF A REAL CPU DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
[	1	Server identification	`0x5D`
p	1	Command	`0x70`
Size	2	Total length of telegram (7 Bytes)	`0x07 0x00`
RCpuID	2	ID of CPU group	e.g. `1006 = 0xEE 0x03`

Example

Get CPU group of real CPU device (RCpuID = 1006)

```
0x1B 0x5B 0x70 0x07 0x00 0xEE 0x03
```

Response

Telegram

```
ESC ] p Size RCpuID GCpuID
```

General Description

Return CPU group of a real CPU device

RETURN CPU GROUP OF A REAL CPU DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	`0x1B`
[	1	Server identification	`0x5D`
p	1	Command	`0x70`
Size	2	Total length of telegram (9 Bytes)	`0x07 0x00`
RCpuID	2	ID of CPU group	e.g. `1006 = 0xEE 0x03`
GCpuID	2	ID of CPU group	e.g. 2003 `= 0xD3 0x07`

Example

Get CPU group (GCpuID = 2003) of real CPU device (RCpuID = 1006)

```
0x1B 0x5D 0x70 0x09 0x00 0xEE 0x03 0xD3 0x07
```

or <NAK>

5.3.11 LOGIN USER AT CON DEVICE

Request

Telegram

```
ESC [ e Size ConID UserID
```

General Description

Login a user at a CON device. Access to CPUs is immediately available.

LOGIN USER AT CON DEVICE

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
e	1	Command	0x65
Size	2	Total length of telegram (9 Bytes)	0x09 0x00
ConID	2	ID of CON device	e.g. 3017 = 0xC9 0x0B
UserID	2	ID of User	e.g. 1 = 0x01 0x00

Example

Login user (UserID = 1) at CON device (ConID = 3017)

```
0x1B 0x5B 0x49 0x09 0x00 0xC9 0x0B 0x01 0x00
```

Response

<ACK> [<ECHO>]

or <NAK>

[ ] = Optional elements

NOTE: To logout a user, use the UserID = 0.

5.3.12 SET FIX FRAME COLOR

Request

Telegram

```
ESC [ r Size CpuID ConID ColorID
```

General Description

Set specific fix frame color to CON or CPU

SET FIX FRAME COLOR

TYPE	BYTES	DESCRIPTION	HEX CODING
ESC	1	Control character	0x1B
[	1	Server identification	0x5B
r	1	Command	0x72
Size	2	Total length of telegram (11 Bytes)	0x0B 0x00
CpuID	2	ID of CPU device	e.g. 1001 = 0xE9 0x03
ConID	2	ID of CON device	e.g. 3007 = 0xBF 0x0B
ColorID	2	Color code	e.g. green = 0x02 0x00

Example

Set red frame for CPU device (CpuID = 1001)

```
0x1B 0x5B 0x72 0x0B 0x00 0xE9 0x03 0x00 0x00 0x04 0x00
```

Set green frame for CON device (ConID = 3007)

```
0x1B 0x5B 0x72 0x0B 0x00 0x00 0x00 0xBF 0x0B 0x02 0x00
```

SET FIX FRAME COLOR

COLOR	COLOR CODE
Off	0x00
Blue	0x01
Green	0x02
Cyan	0x03
Red	0x04
Magenta	0x05
Yellow	0x06
White	0x07

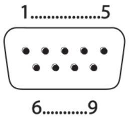
Response

<ACK> [<ECHO>] or <NAK>

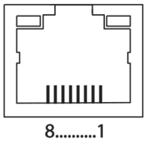
[ ] = Optional elements

5.4 CONNECTOR PINOUTS

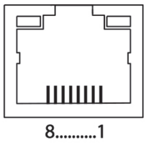
DB9 (Serial) RS-232

Picture	Pin	Signal	Pin	Signal
	1	n.c.	6	DSR
	2	RxD	7	RTS
	3	TxD	8	CTS
	4	DTR	9	n.c.
	5	GND		

RJ-45

Picture	Pin	Signal	Pin	Signal
	1	D1+	5	n.c
	2	D1–	6	D2–
	3	D2+	7	n.c
	4	n.c	8	n.c

RJ-45 (Serial)

Picture	Pin	Signal	Pin	Signal
	1	DCD	5	RxD
	2	DSR	6	TxD
	3	RTS	7	CTS
	4	GND	8	DTR

6. BEST PRACTICES

This chapter provides an overview of the most commonly used switching commands and how they can be operated by using proven code examples of the external serial control.

Full Access (establishing a KVM connection)

Set CON device (`ConID = 3017`) connection to CPU device (`CpuID = 1012`):

```
0x1B 0x5B 0x62 0x09 0x00 0xF4 0x03 0xC9 0x0B 0x00 0x00
```

Disconnect:

```
0x1B 0x5B 0x62 0x09 0x00 0x00 0x00 0xC9 0x0B 0x00 0x00
```

Video Access (establishing a video only connection)

Set CPU device (`CpuID = 1012`) connection to CON device (`ConID = 3017`):

```
0x1B 0x5B 0x62 0x09 0x00 0xF4 0x03 0xC9 0x0B 0x01 0x00
```

Disconnect:

```
0x1B 0x5B 0x62 0x09 0x00 0x00 0x00 0xC9 0x0B 0x01 0x00
```

Private Access (establishing an exclusive KVM session)

Set CON device connection to CPU device and CPU device connection to CON device, `CpuID = 1012` and `ConID = 3017:`

```
0x1B 0x5B 0x62 0x0B 0x00 0xF4 0x03 0xC9 0x0B 0x02 0x00
```

Disconnect:

```
0x1B 0x5B 0x62 0x0B 0x00 0x00 0x00 0xC9 0x0B 0x02 0x00
```

USB 2.0 Access (establishing a USB 2.0 data connection)

To set a USB 2.0 connection based on devices that only consists of USB 2.0 standalone extenders, you have to use the bidirectional switching command:

1. Set CPU device (`CpuID = 1012`) connection to CON device (`ConID = 3017`) and CON device (`ConID = 3017`) connection to CPU device (`CpuID = 1012`):

```
0x1B 0x5B 0x50 0x09 0x00 0xF4 0x03 0xC9 0x0B
```

Switching from a device within an existing connection to another device requires closing the current connection at first. The disconnect must be performed by using the bidirectional command:

1. Disconnect CPU device (CpuID = 1012) from CON device (`ConID = 3017`):

```
0x1B 0x5B 0x50 0x09 0x00 0xC9 0x0B 0x00 0x00
```

2. Connect to the new CON device

For the disconnect just use 0x00 0x00 instead of the concrete CpuID.

After disconnecting the existing connection, a switching break of 1-2 seconds is strongly recommended until the next switching operation should be executed.

CHAPTER 7: TROUBLESHOOTING

7. TROUBLESHOOTING

In the following chapters, support for problems with the DKM Switch API is provided. If you have problems regarding the involved devices, especially the DKM Switch matrix, refer to the respective device manuals.

7.1 NETWORK ERROR

NETWORK ERROR

DIAGNOSIS	POSSIBLE REASON	MEASURE
Network settings are not assumed after editing.	Restart of the matrix not yet completed.	Restart the matrix.

7.2 FAILURE AT THE MATRIX

FAILURE AT THE MATRIX

DIAGNOSIS	POSSIBLE REASON	MEASURE
Serial control impossible or only restrictedly possible.	Different Baud rate of CPU and matrix.	Adapt Baud rate in the CPU.
Serial control via RJ-45 port not possible.	Wrong network cable	Use a crossed network cable

CHAPTER 8: TECHNICAL SUPPORT

8. TECHNICAL SUPPORT

Before contacting support make sure you have read this manual, and then installed and set-up your DKM Switch as recommended.

8.1 SUPPORT CHECKLIST

To efficiently handle your request it is necessary to complete our checklist for support and problem cases. Keep the following information available before you call:

- Company, name, phone number and email
- Type and serial number of the device (see bottom of device)
- Date and number of sales receipit, name of dealer if necessary
- Nature, circumstances and duration of the problem
- Involved components (such as graphic source/CPU, OS, graphic card, monitor, USB-HID/USB 2.0 devices, interconnect cable) including manufacturer and model number
- Results from any testing you have done

8.2 CONTACTING BLACK BOX TECHNICAL SUPPORT

Contact Black Box Technical Support via phone at 877-877-2269 or via email at info@blackbox.com

CHAPTER 9: GLOSSARY

9. GLOSSARY

9.1 VIDEO AND KVM GLOSSARY

The following terms are commonly used in this guide or in video and KVM technology:

AES/EBU: Digital audio standard that is officially known as AES3 and that is used for carrying digital audio signals between devices.

CATx: Any CAT5e (CAT6, CAT7) cable

CGA: Color Graphics Adapter (CGA) is an old analog graphic standard with up to 16 displayable colors and a maximum resolution of 640 x 400 pixels.

Component Video: Component Video (YPbPr) is a high-quality video standard that consists of three independently and separately transmittable video signals, the luminance signal and two color difference signals.

Composite Video: Composite Video is also called CVBS and it is part of the PAL TV standard.

CON Unit: Component of a KVM Extender or Media Extender to connect to the console (monitor(s), keyboard and mouse; optionally also with USB 2.0 devices)

Console: Keyboard, mouse and monitor

CPU Unit: Component of a KVM Extender or Media Extender to connect to a source (computer, CPU)

CVBS: The analog color video baseband signal (CVBS) is also called Composite Video and it is part of the PAL TV standard.

DDC: Display Data Channel (DDC) is a serial communication interface between monitor and source (computer, CPU). It allows a data exchange via monitor cable and an automatic installation and configuration of a monitor driver by the operating system.

DisplayPort: A VESA standardized interface for an all-digital transmission of audio and video data. It is differentiated between the DisplayPort standards 1.1 and 1.2. The signals have LVDS level.

Dual-Access: A system to operate a source (computer, CPU) from two consoles.

Dual-Link: A DVI-D interface for resolutions up to 2560 x 2048 by signal transmission of up to 330 MPixel/s (24-bit).

Dual-Head: A system with two video connections

DVI: Digital video standard, introduced by the Digital Display Working Group (http://www.ddwg.org). Single-Link and Dual-Link standards are distinguished. The signals have TMDS level.

DVI-I: A combined signal (digital and analog) that allows running a VGA monitor at a DVI-I port – in contrast to DVI-D (see DVI).

EGA: The Enhanced Graphics Adapter (EGA) is an old analog graphic standard, introduced by IBM in 1984. A DB9 connector is used for connection.

Fiber: Singlemode or multimode fiber cables

HDMI: An interface for an all-digital transmission of audio and video data. It is differentiated between the HDMI standards 1.0 to 1.4a. The signals have TMDS level.

KVM: Keyboard, video and mouse

Mini-XLR: Industrial standard for electrical plug connections (3-pole) for the transmission of digital audio and control signals

Multimode: 62.5µ multimode fiber cable or 50µ multimode fiber cable

OSD: The On-Screen Display is used to display information or to operate a device.

Quad-Head: A system with four video connections

RCA (Cinch): A non-standard plug connection for transmission of electrical audio and video signals, especially with coaxial cables

S/PDIF: A digital audio interconnect that is used in consumer audio equipment over relatively short distances.

SFP: SFPs (Small Form Factor Pluggable) are pluggable interface modules for Gigabit connections. SFP modules are available for CATX and fiber interconnect cables.

Single-Link: A DVI-D interface for resolutions up to 1920 x 1200 by signal transmission of up to 165 MPixel/s (24-bit). Alternative frequencies are Full HD (1080p), 2K HD (2048 x 1080) and 2048 x 1152.

Single-Head: A system with one video connection

Singlemode: 9µ single-mode fiber cable

S-Video (Y/C): S-Video (Y/C) is a video format transmitting luminance and chrominance signals separately. Thereby it has a higher quality standard than CVBS.

TOSLINK: Standardized fiber connection system for digital transmission of audio signals (F05 plug connection)

Triple-Head: A system with three video connections

USB-HID: USB-HID devices (Human Interface Device) allow for data input. There is no need for a special driver during installation; "New USB-HID device found" is reported. Typical HID devices include keyboards, mice, graphics tablets, and touchscreens. Storage, video, and audio devices are not HID.

VGA: Video Graphics Array (VGA) is a computer graphics standard with a typical resolution of 640 x 480 pixels and up to 262,144 colors. It followed the graphics standards MDA, CGA, and EGA.

9.2 API-SPECIFIC GLOSSARY

ACK: Since packet transfer is not reliable, a technique known as positive acknowledgment with retransmission is used to guarantee reliability of packet transfers.

API: An application programming interface (API) is a specification intended to be used as an interface by software components to communicate with each other. An API may include specifications for routines, data structures, object classes, and variables.

Echo: The response of the DKM Switch matrix to an external command (optional).

NACK: A transmission control character sent by a station as a negative response to the station with which the connection has been set up.

Serial: In telecommunication and computer science, serial communication is the process of sending data one bit at a time, sequentially, over a communication channel or computer bus.

TCP/IP: The Internet protocol suite is the set of communication protocols used for the Internet and similar networks and generally the most popular protocol stack for wide area networks.

DISCLAIMER/TRADEMARKS

DISCLAIMER

Black Box Corporation shall not be liable for damages of any kind, including, but not limited to, punitive, consequential or cost of cover damages, resulting from any errors in the product information or specifications set forth in this document and Black Box Corporation may revise this document at any time without notice.

TRADEMARKS USED IN THIS MANUAL

Black Box and the Black Box logo type and mark are registered trademarks of Black Box Corporation.

Any other trademarks mentioned in this manual are acknowledged to be the property of the trademark owners.

NOTES

NEED HELP?
LEAVE THE TECH TO US

LIVE 24/7 TECHNICAL SUPPORT

1.877.877.2269

BLACK BOX