thinklogical®

A BELDEN BRAND

# ICT18 INTEGRATED CLIENT TRANSMITTER

# API

## USER'S MANUAL

COMPLETE DESCRIPTION AND USE OF THE ICT18 APPLICATION PROGRAMMING INTERFACE

Revision A, November 2025

# Table *of* Contents

## Copyright Notice

**Thinklogical,** A **BELDEN** BRAND®
**100 Washington Street**
**Milford, Connecticut 06460 U.S.A.**
**Telephone: 1-203-647-8700**

**Subject:** ICT18 API User's Manual
**Revision:** A, November 2025

ANAB ACCREDITED ISO/IEC 17021 MANAGEMENT SYSTEMS CERTIFICATION BODY
Certified to ISO 9001:2015
TÜVRheinland® Precisely Right.
MADE IN USA

| | |
|---|---|
| Website: | https://www.thinklogical.com |
| Meta (Facebook): | www.facebook.com/ThinklogicalUSA |
| LinkedIn: | www.linkedin.com/company/thinklogical |
| YouTube: | www.youtube.com/user/thinklogicalNA |
| X (Twitter): | @thinklogical |

## To the Reader:

**! READ THE INSTRUCTIONS THOROUGHLY BEFORE STARTING ANY PROCEDURE! !**

STOP WARNING! Do not attempt to open or disassemble this product. Please contact your dealer or ***thinklogical*** for qualified servicing. NO USER SERVICEABLE PARTS INSIDE !

For any product support, technical issues or other questions not covered in this document, please feel free to contact us:

- **Email:** support@thinklogical.com
- **Telephone:** **1-203-647-8700** *or* **1-800-291-3211**
- **Website:** https://www.thinklogical.com **Chat live** with a Customer Service Representative.
- **ICT18 Product Manual:** https://www.thinklogical.com/wp-content/uploads/2025/03/Manual_ICT18_Rev_B.pdf

# The ICT18 Application Programming Interface

This document describes the Application Programming Interface (API) command set used to control Thinklogical's **ICT18 Integrated Client Transmitter**. The ICT18 hosts any standard VDI Client software and is compatible with most third-party accredited software images. Users can also boot and load the OS from servers on their own network.

## Enumeration

The ICT18 communicates with an external PC by means of an FTDI USB (mini-B) serial port, labeled MGMT, located on each of up to 18 Processor Platform Cards (PPC) and on the Control Module. The MGMT Ports are used for configuration, status reporting, troubleshooting and updating firmware & software.

**Baud Rate:** 115200, **Data Bits:** 8, **Parity:** None, **Stop Bits:** 1, **Flow Control:** None.

*Although sent characters are echoed back by default, this can be modified using this API. See pg.* [6].

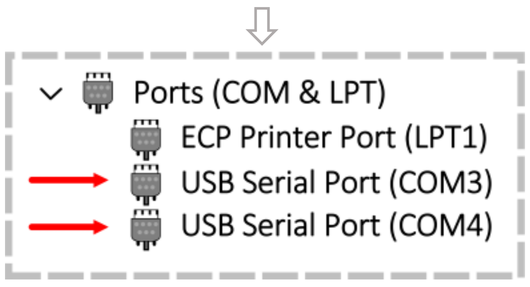
**Control Module MGMT port** – Used for Chassis, Tx Card and PPC functions. It includes settings that can be applied to all cards in the chassis simultaneously or one card individually. ***This is the connection most often used for such functions.*** This port is also used to update the Chassis FPGA Program Code.

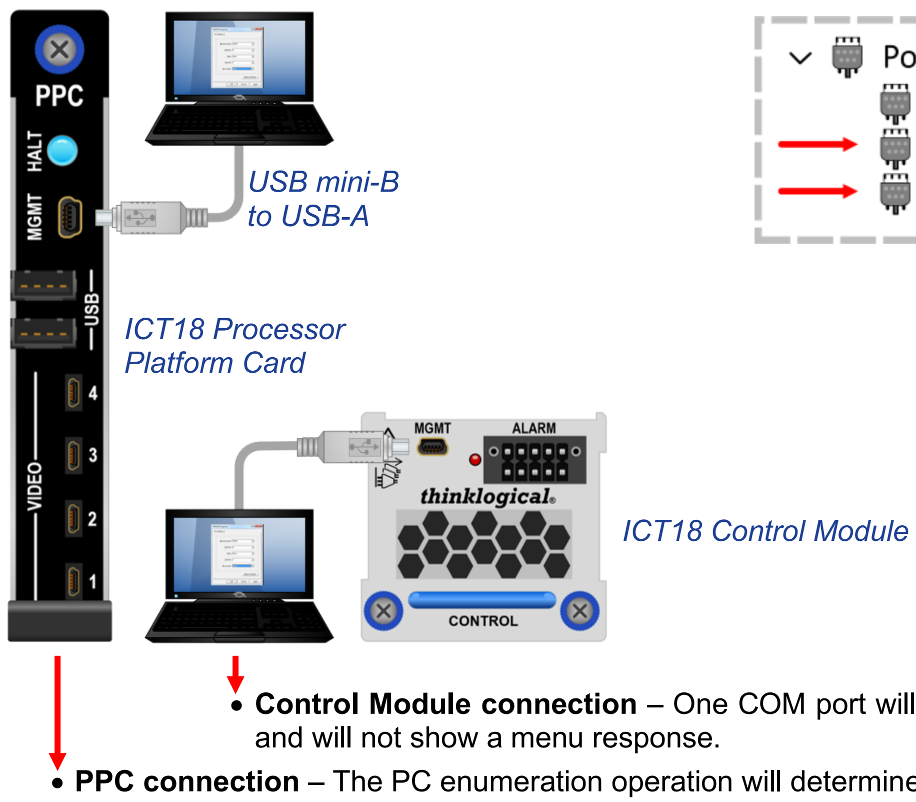
For example, to set the EDID table of a card or video port:

1. Connect to the Control Module MGMT Port.
2. Select 3: Processor Platform Cards.
3. Select 3: PPC EDID Parameters.

**PPC MGMT port** – Accesses both the PPC and the TX Card. This port is used to update both cards' FPGA Program Codes and for more detailed card-specific functions.
Open a terminal emulator and connect to either the PPC or Control Module's MGMT port with the USB mini-B cable.

Once the USB cable is connected, it will enumerate as a serial port. **Two new Serial Ports** will be assigned when connecting to either module:





- **Control Module connection** – One COM port will be used while the other is reserved and will not show a menu response.
- **PPC connection** – The PC enumeration operation will determine which COM port will be for the PPC and which will be for the TX Card.

Select **<enter>** to display a menu. Select "**m**" for the top (main) menu level. This also confirms your connection.

## User Inputs

A user can issue an operation, a command and parameters. An **operation** consists of GET or SET to retrieve or change a setting. A **command** is related to what can be read or changed, such as a fan speed setting or temperature range. **Parameters** are inputs that provide specific instructions or data, influencing the behavior and output of the command.

Certain functions require parameters for GET and SET operations. Some require the user to indicate the module or specific blade (the Transmitter Cards and PPCs are commonly referred to as a *blades*) from which to retrieve the information. Other functions do not require a parameter specifying a module but may require other parameters, such as a value or enable/disable.

Parameters for each function are specific to that command, but the command structure is always the same. Example: **GET(CFAN)** retrieves the fan setting from the Control Module. A typical reply may be **CFAN=OK:16** This indicates that the command was executed correctly (OK), along with its current fan speed setting (16). To change a setting, use the **SET** operation for the command to be changed and for the new parameter.

In the case of the fan speed setting, for instance, a user may issue **SET(CFAN:50)**. *Note that the command and parameter are in* (parenthesis).
The expected reply will be **CFAN=OK:50**
This indicates the reply was properly executed and the new fan speed setting is 50.

*If a user attempts to issue a non-executable parameter, the reply will indicate this*. In the case of a fan speed setting, for example, parameters range from 0 to 255. Any other number will not be accepted. Thus, issuing **SET(CFAN:300)** will result in **CFAN=BAD:50**, indicating that the out-of-range SET(FAN:300) was not executed and the current fan speed remains at 50.

## Commands and Operations

The following format is typical for all commands.

### *Command Structure*

**GET** – Reads (or ‘gets’) data by calling a function related to that data.
**SET** – Writes (or ‘sets’) data by calling a function related to that data.
**(FUNCTION:X)** The function related to data to be read. X may indicate from where the data is read, such as C for a control unit or a number to indicate a slot. X tends to conform to that standard, but there may be exceptions, depending on the function being called. Some functions do not have parameters. If the parameter does not relate to a particular subassembly, it is omitted because it is a specific function.

### *Reply Structure*

A command, whether SET or GET, receives a reply such as the following:
**FUNCTION=<status>:<parameters returned, if required>**

Status may consist of OK, NO, BAD or PARAM
**OK** – Command was valid and all is well.
**NO** – The command was valid but cannot executed. This usually means that the function does not have a SET operation.
**BAD** – The command was a valid command but count not be executed. This usually means one or more of the parameters were invalid. For instance, one asks for the status of slot 20, a slot that does not exist.
**PARAM** – The command is valid but the parameter is not.

### *Command Line and Reply Format*

This paragraph describes commands related to a particular Control Module, Power Supply, TX Card or PPC. (Other command and reply lines are specifically stated in a particular function.)
In the command line, Control Modules are designated by issuing either a C or a number that ranges from 1 to 18, for the corresponding  module or slot number.
**GET(<FUNCTION>:C)** Indicates that the function gets telemetry related to the Control Module C.
**GET(<FUNCTION>:2)** indicates the API user requests telemetry related to slot 2.

Reply lines are formatted in the following manner: **<FUNCTION>=<STATUS>:<TELEMETRY>**
Status can be OK, NO or BAD, as described above.

Telemetry is function specific. If more than one datapoint is part of the reply, each data point is delimited by a forward slash ( / ): **<telemetry1>/<telemetry2>/<telemetry3>/ etc.**

If the telemetry is related to multiple slots, each slot's telemetry is delimited by a comma (,):

**<telemetry1 slot1>/<telemetry2 slot1>,<telemetry1 slot2>/<telemetry2 slot2>,**
**<telemetry1 slotN>/<telemetry2 slotN>**

Certain functions and replies may contain several points of telemetry, while others may have only one, and some may have none.

Some functions are only for a specific module (CFAN, for instance, below). Others are for multiple slots with multiple telemetry points (see WHAT, below). Some functions are for a specific module, but will retrieve multiple points of telemetry (TXBLADE, pg. [11], for example). Each is command specific.

## Serial Port Echo (ECHO)

The **GET** operation reads or changes whether characters sent to the Control Module by means of the serial port are sent back to the sender. When Echo is off, the command prompt and certain formatting characters will not be sent. These are only necessary for a human operator using a simple terminal since a machine sender does not need formatting characters.
Command: **GET(ECHO)**
Reply: **ECHO=OK:<X>**, where X can be 1 for Echo on or 0 for Echo off.
Example: **GET(ECHO ECHO=OK:1)** meaning the command was correct and the  Echo state is on (1).

**SET** changes the Echo behavior to either Echo back or not.
Command: **SET(ECHO:0)**
Reply: **ECHO=OK:0** meaning the command was executed correctly and the current Echo state is off (0).
Command: **SET(ECHO:1)**
Reply: **ECHO=OK:1** meaning the command was executed correctly and the current Echo state is on (1).
Any parameter other than 0 or 1 will see **ECHO=BAD**

## What is this/What's installed (WHAT)

The **GET** operation reads the number of slots and what may be installed in each of them.
Command: **GET(WHAT)**
Reply: **WHAT=<status>:<number of slots>,<slot 1 TX/slot1 PPC>,<slot N TX/slot N PPC>,**
Example: **WHAT=OK:18,Emp/Emp,TX/PPC,Emp/Emp,TX/PPC…**
Translation: This unit has a total of 18 slots. Slot 1 is empty, slot 2 is populated with a transmitter and PPC, slot 3 is empty, slot 4 is populated with a transmitter and PPC, and so on for the remaining slots.
The **SET** function is not implemented:
Command: **SET(WHAT)**
Reply: **WHAT=NO**, meaning this command is not executed.

## Fan Speed (CFAN)

The **GET** operation reads or controls the fan speed, with a duty cycle ranging from 0 to 255.
Command: **GET(CFAN)**
Reply: **CFAN=OK:50**, meaning the command was executed correctly (OK) and the fan speed is 50.
Command: **SET(CFAN:<NNN>** where NNN is a number related to the speed, ranging from 0 to 255.
Example: **SET(CFAN:50)**
Reply: **CFAN=OK:50**, meaning the command was executed correctly (OK) and the current fan speed is 50, as requested.
Command: **SET(CFAN:300)** *This command is out of range.*
Reply: **CFAN:BAD:50**, meaning a bad parameter (BAD) was attempted and the fan speed is currently 50.

## API Control (API)

The **GET** operation determines if the API is active, version number, etc.
Command: **GET(API)**
Reply: **API=OK:1,1.1.6** This means the API is active (1) and the current API version is 1.1.6

By default, anything sent to the serial port is parsed by the API. There is a verbose menu, predating this API. To turn that API off and this menu on, issue the command **SET(API:0)**. Any character sent after this command will be interpreted as a menu command.

To turn the API back on, power cycle the Control Module.

## Product Name (or Part Number (PRTNUM)

This command reads the part number of a particular module (Control Module or blade)
**GET(PRTNUM:X)**, where X is either *C*, to retrieve this data from the Control Module, or a *blade number*, ranging from 1 to 18, to retrieve this data from a specified blade.
Command: **GET(PRTNUM:C)**
Reply **PRTNUM=OK:C:CHS-HD0004**, meaning the command was executed correctly (OK) and the part number for the Control Module ( C) is C:CHS-HD0004.
Command: **GET(PRTNUM:2)**
Reply: **PRTNUM=OK:2:ICT-XMF-C00064/ICT-PPC-100043**, meaning the command was executed correctly (OK) and blade 2 is a transmitter with the part number ICT-XMF-C00064 and a PPC with the part number ICT-PPC-100043.
Command: **GET(PRTNUM:1)**
Reply: **PRTNUM=OK:1:NI/NI,** , meaning the command executed correctly (OK) but a transmitter and a PPC are **N**ot **I**nstalled (**NI**/**NI**) in slot 1.
Command: **GET(PRTNUM:20)**
Reply: **PRTNUM=BAD:20.** The command was invalid because slot 20 does not exist.
The **SET** command is not implemented. Attempts to set a part number using SET(PRTNUM:X) will not be executed.
Reply: **PRTNUM=NO**

## Serial Number (SERNUM)

The **GET** operation reads the serial numbers associated with the Control Board or a blade installed in a particular slot, then issues a **GET(SERNUM:X)**, where X is either C or a number between 1 and 18, corresponding to a slot.
Command: **GET(SERNUM:C)**
Reply: **SERNUM=OK:C:123456789.**
Command: **GET(SERNUM:2)**
Reply: **SERNUM=OK:2:11-250147/11-250098**, meaning the TX serial number is 11-250147. The PPC has serial number 11-250098.
Command: **GET(SERNUM:1)**
Reply: **SERNUM=OK:1:NI/NI**
Slot 1 does not have a TX Card or PPC installed, but the command was valid, indicated by OK, but the modules are **N**ot **I**nstalled (**NI**/**NI**).
Command: **GET(SERNUM:19)**
Reply: **SERNUM=19:BAD** This is an invalid serial number request as there is no slot 19.
The **SET** command is not implemented. Attempts to set a serial number using SET(SERNUM:X) will not be executed.
Command: **SET(SERNUM:X)**
Reply: **SERNUM=NO**

## FPGA Version/Software Version (FWVER)

The **GET** operation reads the firmware and EEPROM versions.
Command **GET(FWVER:C)** reads the firmware and EEPROM versions of the Control Module.
Reply: **FWVER=OK:C:1.38.0/1**, where 1.38.0 is the firmware version and 1 is the EEPROM version/
Command: **GET(FWVER:2)** reads both the FPGA and NIOS versions of blade 2.
Reply: **FWVER=OK:2:2.1.0/27.10/1.1.0/62.10**, indicates the TX has FPGA version 27 and NIOS version 10. The PPC has FPGA version 1.1.0 and NIOS version 62.10.
Command: **GEET(FWVER:1)** reads the firmware version of blade 1, which is not installed.
Reply: **FWVER=OK:1:NI/NI/NI/NI,** meaning the command was valid (OK) but the TX Cards and PPCs were **Not** **I**nstalled (**NI**/**NI**/**NI**/**NI**).
Command: **GET(FWVER:20)** reads the firmware version for blade 20, which does not exist.
Reply: **FWVER=20:BAD**, meaning the request for firmware version of blade 20 was invalid.

The **SET** command is not implemented. Attempts to set a version number using SET(FWVER:X) will not be executed.
**SET(FWVER:C)** or **SET(FWVER2)**
Reply: **FWVER:NO**

## Time

The **GET** operation retrieves the time the system has been up or the time since the last reset.
**GET(TIME)**
Reply: **TIME=OK: TIME=OK:0/0/0/17/53**
The system has been up for  0 years, 0 days, 0 hours, 17 minutes, 53 seconds.
SET is not a valid command.
**SET(TIME)**
Reply: **TIME:NO**

## Temp

The GET operation retrieves the temperature of the Control Module's PCB.
**GET(TEMP)**
Reply: **TEMP=OK:29**
**SET** is not a valid command.
**SET(TEMP)**
Reply: **TEMP:NO**

## Power

The **GET** operation retrieves status of both power supplies.
**GET(POWER)**
Reply: **POWER=OK:POK,PFL**, meaning power supply 1 is OK (POK), power supply 2 has failed (PFL).
The power supply conditions can be **POK/PNI/PFL** (**P**S **Ok**ay/**P**S **N**ot **I**nstalled/**P**S **F**ai**l**ed).
**SET** is not a valid command.
**SET(POWER)**
Reply: **POWER:NO**

## Fan Tray

The **GET** operation retrieves status of the Fan Tray.
**GET(FANTRAY)**
Reply: **POWER=OK:IN**, meaning the Fan Tray is installed. The Fan Tray status can be **IN** for **In**stalled or **NI** for **N**ot **I**nstalled.
**SET** is not a valid command.
**SET(FANTRAY)**
Reply: **FANTRAY:NO**

## Alarm

The **GET** operation retrieves the current status of the system's alarms. (Enabling and disabling an alarm is described in the following paragraphs.)
**GET(ALARM)**
Reply: **ALARM=OK:Off,Off,Off,Off,Off,On**, meaning: Command executed correctly **OK:**
Fan alarm is **Off**
Tx Card alarm is **Off**
PPC alarm is **Off**
Temp warning alarm is **Off**
Power supply1 alarm is **Off**
Power supply2 alarm is **On**
**SET** is not a valid command.
**SET(ALARM)**
Reply: **ALARM:NO** *(See the following paragraphs to set an alarm.)*

## Alarm Fan Enable (ALFANMASK)

The **GET** operation allows a user to read, enable or disable the Fan alarm.
**GET(ALFANMASK)**
Reply: **ALFANMASK=OK:E**
Meaning OK, the message was valid and the Fan alarm mask is enabled (E). *Or (D) for disabled.*
The **SET** operation will enable or disable the alarm.
**SET(ALFANMASK:D)**
Reply: **ALFANMASK=OK:D**, meaning the Fan alarm is disabled (D). This alarm can be enabled by issuing **SET(ALFANMASK:E)**
Expected reply: **ALFANMASK=OK:E**
Any parameter other than E or D will receive the reply **ALFANMASK=OK:BAD**, meaning the command is valid but its parameter is not.

## Alarm TX Blade Enable (ALTXBLADEMASK)

The **GET** operation allows a user to read, enable or disable the TX Card alarm.
**GET(ALTXBLADEMASK)**
Reply: **ALTXBLADEMASK=OK:E**, meaning OK, the message is valid and the alarm mask is enabled (E). *Or (D) for disabled.*
The **SET** operation can enable or disable the alarm.
**SET(ALTXBLADEMASK:D)**
Reply: **ALTXBLADEMASK=OK:D**, meaning the alarm is disabled (D). This alarm can be enabled with: **SET(ALTXBLADEMASK:E)**
Expected reply: **ALTXBLADEMASK=OK:E**
Any parameter other than E or D will receive the reply **ALTXBLADEMASK=OK:BAD**, meaning the command is valid but its parameter is not.

## Alarm PPC Blade Enable (ALPPCBLADEMASK)

The **GET** operation allows a user to read, enable or disable the PPC alarm.
**GET(ALPPCBLADEMASK)**
Reply: ALPPCBLADEMASK=OK:E, meaning OK, the message is valid and the alarm mask is enabled (E). *Or (D) for disabled.*
The **SET** operation will enable or disable the alarm.
**SET(ALPPCBLADEMASK:D)**
Reply: **ALTXBLADEMASK=OK:D**, meaning the fan alarm is disabled (D). This alarm can be enabled with: **SET(ALPPCBLADEMASK:E)**
Expected reply: **ALPPCBLADEMASK=OK:E**
Any parameter other than E or D will receive the reply **ALPPCBLADEMASK=OK:BAD**, meaning the command is valid but its parameter is not.

## Alarm Temperature Enable (ALTEMPMASK)

The **GET** operation allows a user to read, enable or disable the Temperature alarm.
**GET(ALTEMPMASK)**
Reply: **ALTEMPMASK=OK:E**, meaning OK, the message is valid and the alarm mask is enabled (E). *Or (D) for disabled.*
The **SET** operation will enable or disable the alarm.
**SET(ALTEMPMASK:D)**
Reply: **ALTEMPMASK=OK:D**, meaning the alarm is disabled (D). This alarm can be enabled with: **SET(ALTEMPMASK:E)**
Expected reply: **ALTEMPMASK=OK:E**
Any parameter other than E or D will receive the reply **ALTEMPMASK=OK:BAD**, meaning the command is valid but its parameter is not.

## Alarm Power Supply 1 Enable (ALPS1MASK)

The **GET** operation allows a user to read, enable or disable the Power Supply 1 alarm.
**GET(ALPS1MASK)**
Reply: **ALPS1MASK=OK:E**, meaning OK, the message is valid and the PS1 alarm mask is enabled (E). *Or (D) for disabled.*
The **SET** operation will enable or disable the alarm.
**SET(ALPS1MASK:D)**
Reply: **ALPS1MASK=OK:D**, meaning, the alarm is disabled (D). This alarm can be enabled with: **SET(ALPS1MASK:E)**
Expected reply: **ALPS1MASK=OK:E**
Any parameter other than E or D will receive the reply **ALPS1MASK=OK:BAD**, meaning the command is valid but its parameter is not.

## Alarm Power Supply 2 Enable (ALPS2MASK)

The **GET** operation allows a user to read, enable or disable the Power Supply 2 alarm.
**GET(ALPS2MASK)**
Reply: **ALPS2MASK=OK:E**, meaning OK, the message is valid and the PS2 alarm mask is enabled (E). *Or (D) for disabled.*
The **SET** operation will enable or disable the alarm.
**SET(ALPS2MASK:D)**
Reply: **ALPS2MASK=OK:D**, meaning, the alarm is disabled (D). This alarm can be enabled with:
**SET(ALPS2MASK:E)**
Expected reply: **ALPS2MASK=OK:E**
Any parameter other than E or D will receive the reply **ALPS2MASK=OK:BAD**, meaning the command is valid but its parameter is not.

## Thermal States (THERM)

The **GET** operation reads the thermal states of both Power Supplies, the Control Card and any installed blades.
**GET(THERM:P)**
Reply: **THERM=OK:P:32,29,NoAC**, meaning: Command executed correctly **OK:P**
The Control Board's temperature is **32**
Power Supply1's temperature is **29**
Power Supply 2 is off **NoAC**
Command: **GET(THERM:2)**
Reply: **THERM=OK:2:67,96,11.9,2.1,3855,58,11.9,01.6,**
Command executed correctly **OK:2** *(slot 2 blades)*
TX board temperature **67** C
TX FPGA temperature **96** C
TX board Voltage in ,**11.9** V
TX board Current in **2.1** A
PPC Fan RPM **3855**
PPC Heatsink temperature **58**
PPC Voltage in **11.9** V
PPC Current in **1.6** A
The **SET** function is disabled.

## Low-Speed Connect State (LSCON)

The **GET** operation reads the state of a particular blade's low-speed connection.
Command: **GET(LSCON:2)**
Reply: **LSCON:OK:2:NO**, meaning blade 2 does not have its low speed connected.
Command: **GET(LSCON:1)**
Reply: **LSCON=OK:1:NI**, meaning Blade slot 1 is Not Installed.
Command: **GET(LSCON:20)**
Reply: **LSCON:BAD**, meaning a bad parameter. In this case, slot 20 does not exist.
The **SET** command is not implemented.
Command: **SET(LSCON)**
Reply: **LSCON:NO**

## TX Blade Status (TXBLADE)

The **GET** operation reads the state of a particular TX blade.
Command: **GET(TXBLADE:2)**
Reply: **TXBLADE=OK:2:Yes/296.98/Yes/296.98/Yes/296.98/Yes/296.98/**
This reports each head as being valid or invalid, along with its associated PCLOCK.
The **SET** function is disabled.

## TX Intuitive Mouse (IMOUSE)

The **GET** operation controls the state of the intuitive mouse feature on a particular TX blade.
Command: **GET(IMOUSE:X)**, where X is the slot number for the blade.
Reply **IMOUSE:X.Y** where X is the slot number and Y is either 0 for disabled or 1 for enabled.
The **SET** command:
**SET(IMOUSE:X,Y)** where X is the blade's slot number and Y is either 0 for disabled or 1 for enabled.

## Regulatory Compliance Requirements

### Symbols Found on Our Products

Markings and labels on our products follow industry-standard conventions. Regulatory markings found on our products comply with all required domestic and most international requirements.

Intertek
3072510

### Regulatory Compliance

Thinklogical's® products are designed and made in the U.S.A. These products have been evaluated by a certified testing laboratory and found compliant with the following standards for both domestic USA and most international locations:

#### North America

**Safety**
UL 62368-1:2019 Ed.3+R:22 Oct 2021
CSA C22.2#62368-1:2019 Ed.3+U1

**LASER Safety**
CDRH 21 CFR 1040.10
Class 1 LASER Product
Canadian Radiation Emitting Devices Act, REDR C1370
IEC 60825-1:2014
Class 1 LASER Product

**Electromagnetic Interference**
FCC 47CFR Part 15 Subpart B: 2013 Class A
IC ICES-003:2020 Ed.7

#### Australia & New Zealand

This is a Class A product. In a domestic environment this product may cause radio interference, in which case the user may be required to take corrective action.

#### European Union Declaration of Conformity

Manufacturer's Name & Address:

**Thinklogical,** A **BELDEN** BRAND®
**100 Washington Street**
**Milford, Connecticut 06460 USA**

Thinklogical's products comply with the requirements of the Low Voltage Directive 2014/35/EU, the EMC Directive 2014/30/EU, the RoHS Directive 2011/65/EU, the WEEE Directive 2012/19/EU and carry the CE marking accordingly.

## Standards with Which This Product Complies

**Safety**

EN IEC 62368-1:2020+A11
BS EN IEC 62368-1:2020+A11
CB Scheme Certificate

**Electromagnetic Emissions**

CENELEC EN 55032:2015+A11

**Electromagnetic Immunity**

CENELEC EN 55035:2017+A11
BS EN 55035
EN 61000-3-2:2019+A1 Harmonics
EN 61000-3-3:2013+A1; A2 Flicker
EN 61000-4-2:2009 Electro-Static Discharge Test
EN 61000-4-3:2006 A1:2008, A2:2010 Radiated Immunity Field Test
EN 61000-4-4:2004 Electrical Fast Transient Test
EN 61000-4-5:2006 Power Supply Surge Test
EN 61000-4-6:2009 Conducted Immunity Test
EN 61000-4-11:2004 Voltage Dips & Interrupts Test

## Additional Electromagnetic Emissions Information

The following statements may be appropriate for certain geographical regions and might not apply to your location:

- This Class A digital apparatus meets all requirements of the Canadian Interference-Causing Equipment Regulations. *Cet appareil numérique de la classe A respecte toutes les exigencies du Règlement sur le mat érial brouilleur du Canada*.
- This is a Class A product. In a domestic environment, this product may cause radio interference, in which case the user may be required to take corrective action.
- This equipment has been verification-tested and found to be compliant with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate, radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications in which case the user may be required to make adequate corrective measures at their own expense.
- This Class A digital apparatus complies with Canadian ICES-003 and has been verified as compliant within the Class A limits of the FCC Radio Frequency Device Rules (FCC Title 47, Part 15, Subpart B CLASS A), measured to CISPR 22:1993 limits and methods of measurement of Radio Disturbance Characteristics of Information Technology Equipment.
- The user may notice degraded audio performance in the presence of electro-magnetic fields.
- The customer shall verify that this product meets the appropriate national/regional requirements if those requirements for conducted/radiated electromagnetic emissions fall outside the scope of testing currently performed on this product.

# Thinklogical Support

## Customer Support

- **Website:** https://www.thinklogical.com/downloads/

Visit our website for products, updates, support documents and useful information about all the products and services we offer, including:

- **FPGA Update Guides**
- **Quick-Start Guides**
- **User Manuals** (for viewing online or for download)
- **Visio® Stencils**
- **Chat Live** with a Technical Support Representative.

## Technical Support

For technical issues/questions, product repairs, updates or request for Return Merchandise Authorization, use either of the following methods:

- **Email:** support@thinklogical.com – (preferred)
- **Telephone:** **1-203-647-8700** or **1-800-291-3211 -** Monday-Friday, between 8:30am and 5:00pm, Eastern Time Zone.

## Product Support

### Warranty

Thinklogical warrants this product against defects in materials and workmanship for a period of one year from the date of delivery, with longer terms available at the time of purchase on most products. Thinklogical and its suppliers disclaim all other warranties. Please refer to your product invoice for the Warranty Terms & Conditions.

Defect remedy shall be the repair or replacement of the product, provided that the defective product is returned to the authorized dealer within a year from the date of delivery.

### Return Merchandise Authorization

If you wish to return your device, contact the Thinklogical-authorized dealer where you purchased the device.

Or -

If you need to return a product to Thinklogical directly, please use the support email address above. Support will need the device's serial number and a description of the issue. You will then be assigned a **R**eturn **M**erchandise **A**uthorization (RMA) number. Pack the device in its original box, if possible, and return it with the RMA number printed on the outside of the box. **Please DO NOT return a product to Thinklogical without a *Return Merchandise Authorization*.**

### Our Address

If you need to write to us or return a product, please use the following address:

**Thinklogical,** A **BELDEN** BRAND®
**100 Washington Street**
**Milford, CT 06460 USA**
**Attn:** ***RMA#***

*Please include the Return Merchandise Authorization number.*

## About Thinklogical, A BELDEN BRAND®

**thinklogical, A BELDEN BRAND**
**100 Washington St.**
**Milford, CT 06460**

Thinklogical, a Belden Brand®, is the leading manufacturer and provider of fiber-optic and CATx video, KVM, audio, and peripheral extension and switching solutions used in video-rich, big-data computing environments.

**Thinklogical offers the only fiber-optic KVM Matrix Switches in the world that are accredited to the Common Criteria EAL4, TEMPEST SDIP 24 Level B, and NATO NIAPC Evaluation Scheme: GREEN and the U.S. DoD DISA JITC UCR 2013 APL information assurance standards. And Thinklogical Velocity products are the first system with both KVM and video matrix switching capabilities to be placed on the Unified Capabilities Approved Product List (UC APL) under the Video Distribution System (VDS) category.**

**Thinklogical products are designed and manufactured in the USA and are certified to the ISO 9001:2015 standard.**

Certified to ISO 9001:2015

Thinklogical is headquartered in Milford, Connecticut, U.S.A., and is owned by Belden, Inc., St. Louis, Missouri (http://www.belden.com). For more information about Thinklogical products and services, please visit https://www.thinklogical.com.

NOTES: