

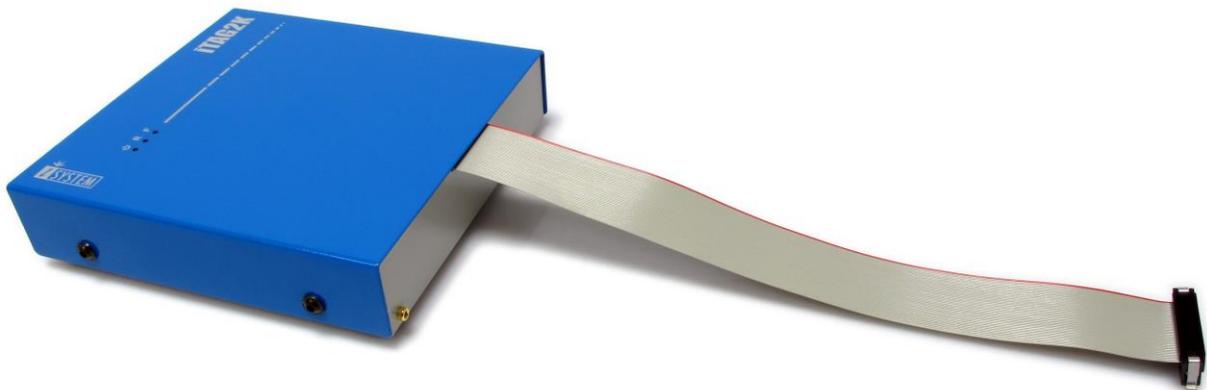
---

---

## Hardware Reference

# iTAG.2K JTAG/ETM Debugger

|               |                |
|---------------|----------------|
| Ordering code | ITAG-CORTEX-2K |
|---------------|----------------|



Thank you for purchasing this product from iSYSTEM. This product has been carefully crafted to satisfy your needs. Should any questions arise, do not hesitate to contact your local distributor or iSYSTEM directly. Our technical support personnel will be happy to answer all your technical support questions.

All information, including contact information, is available on our web site [www.isystem.com](http://www.isystem.com). Feel free also to explore our alternative products.

iSystem constantly yields for development and therefore certain pictures in this documentation may vary slightly from the actual product you received. The differences should be minor, but should you find more serious inconsistencies of the product with the documentation, please contact your local distributor for more information.

This document and all documents accompanying it are copyrighted by iSYSTEM and all rights are reserved. Duplication of these documents is allowed for personal use. For every other case a written consent from iSYSTEM is required.

Copyright © 2015 iSYSTEM, AG.

All rights reserved.

All trademarks are property of their respective owners.



## Supported cores and microcontrollers

| Supported ARM cores |
|---------------------|
| Cortex-M0           |
| Cortex-M1           |
| Cortex-M3           |
| Cortex-M4           |
| ARM720T             |
| ARM7DI              |
| ARM7TDMI            |
| ARM7TDMI-S          |
| ARM9TDMI            |
| ARM920T             |
| ARM922T             |
| ARM925              |
| ARM926EJ-S          |
| ARM940T             |
| ARM946E-S           |
| ARM966E-S           |
| XScale              |

Check the Supported MCUs section at [www.isystem.com](http://www.isystem.com) for up-to-date list of supported microcontrollers.

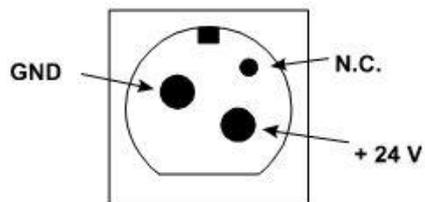
## Status LEDs

There are three status LEDs on the iTAG.2K. Status LEDs inform the user of the current status of the emulation system. Their meaning is:

-  – When lit, the unit is turned on
- R – When lit, the target application being controlled is running
- F – When lit, the unit is free for communication, i.e. winIDEA can connect to it.

## Powering the emulator

A round 3-pin power connector is located on the rear of the iTAG.2K JTAG/ETM Debugger.



*Power connector pinout, view from the rear of the Emulator*

The iTAG.2K unit accepts a wide input voltage range from 12V to 24V DC, thus enabling the Emulator to work also with a 12V or 24V car battery. The necessary power supply is delivered beside the iTAG.2K unit.

**Note:** Use only original iSYSTEM accessories for powering the iTAG.2K. If you wish to use a power supply different from the delivered one, please consult with iSYSTEM first.

iTAG.2K is a universal on-chip emulation platform. Additionally, a target specific cable adapter is required to debug the target based on a specific architecture.

Valid input voltage range for all debug signals is between 1.8 - 5.5V

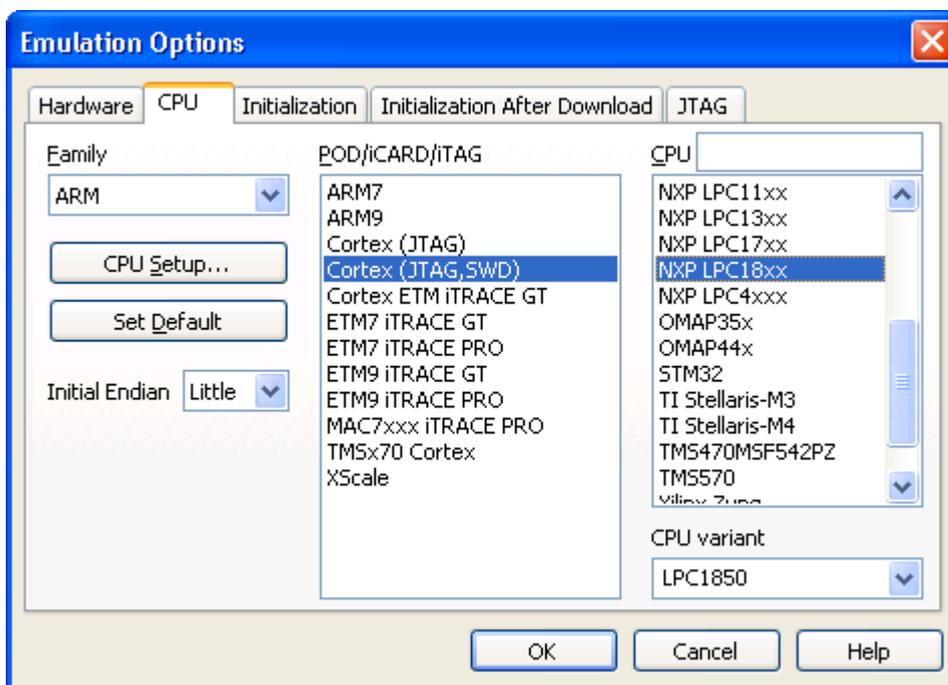
## Operating Temperature Range

All iSYSTEM products, unless explicitly otherwise noted, are specified to operate at room temperatures (specifically, between 10°C/50°F and 40°C/105°F).

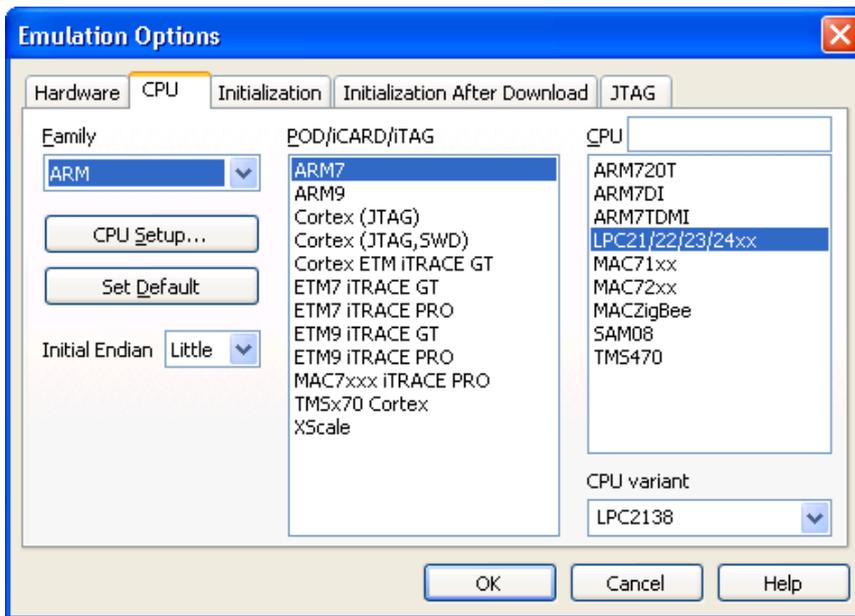
## winIDEA Configuration

Based on the ARM architecture being debugged, choose proper setting in the 'POD/iCARD/iTAG' selection in the 'Emulation Options' dialog. Next, in the CPU area to the right, select your microcontroller family and then a specific target device in the 'CPU variant' combo box in the right bottom corner. Microcontroller specific settings are configured in the dialog, which is opened by pressing the 'CPU Setup...' button.

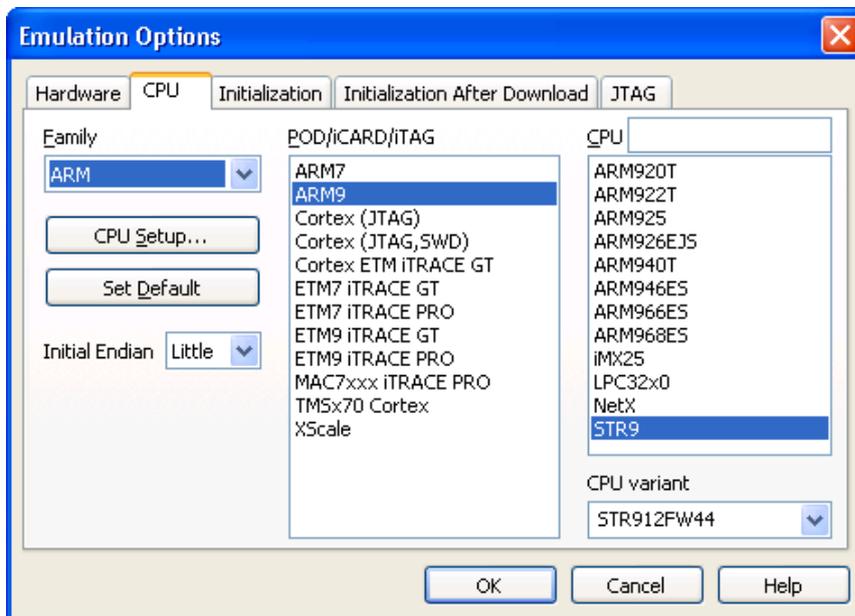
- When debugging Cortex-M based devices select 'Cortex (JTAG/SWD)' selection under the 'POD/iCARD/iTAG' selection.



- Select 'ARM7' under the 'POD/iCARD/iTAG' selection when debugging ARM7 based devices.



- Select 'ARM9' under the 'POD/iCARD/iTAG' selection when debugging ARM9 based devices.



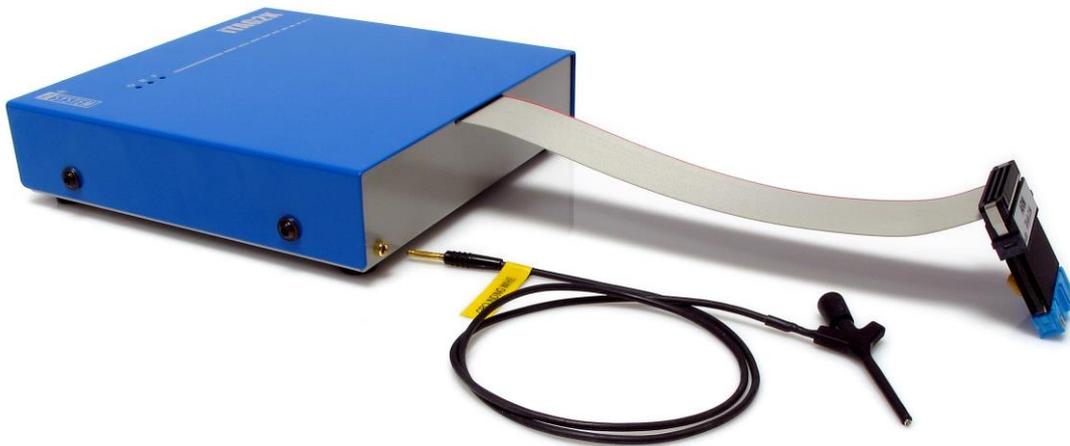
## Using grounding wire

In case of the on-chip emulation, it has been proven that a development tool can be damaged at the moment when the emulator's debug connector is plugged into the target system when neither the target nor the emulator are powered up yet. At this point in time, there could be ground potential difference between the emulator and the target way over 1000V. Such voltage difference is then discharged over the emulator and the target, which can destroy electronic components of the emulator and/or the target.

The voltage difference can be introduced by:

- power supply (target, emulator), which does not have the power outlet ground connected with the power supply ground.
- power outlets which have different ground potentials
- PC, when iTAG.2K connects to the PC through the USB port

Connecting a dedicated grounding wire, which is shipped with the iTAG.2K unit, between the iTAG.2K system and the target before the target debug cable adapter is connected to the target, makes the complete development system even more robust and resistant to the mentioned electrical discharge problem - despite the fact the iTAG.2K development system features already a high quality protection on all connecting signals by default.



*iTAG.2K with the grounding wire and the ground pin in the left bottom corner*

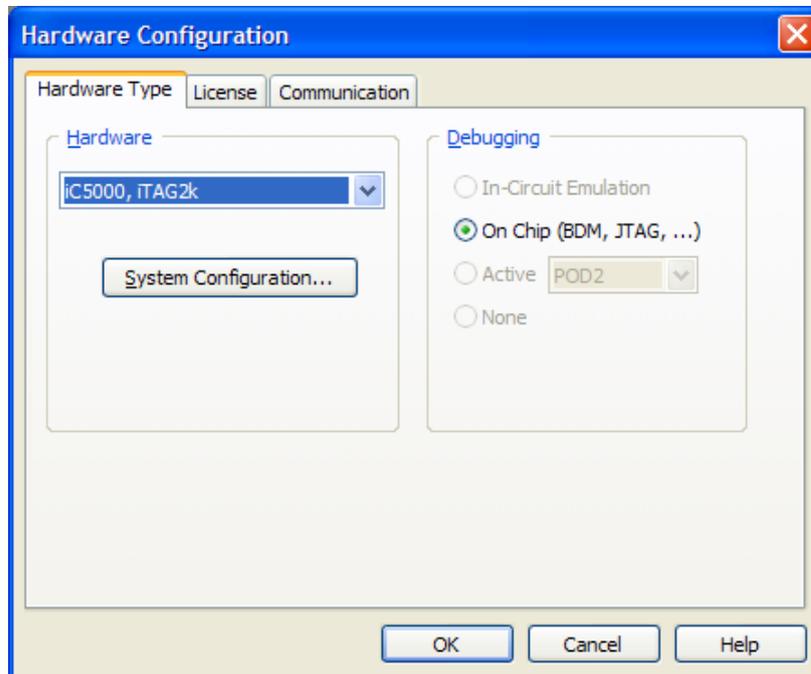


*The grounding wire connecting the target and iTAG.2K*

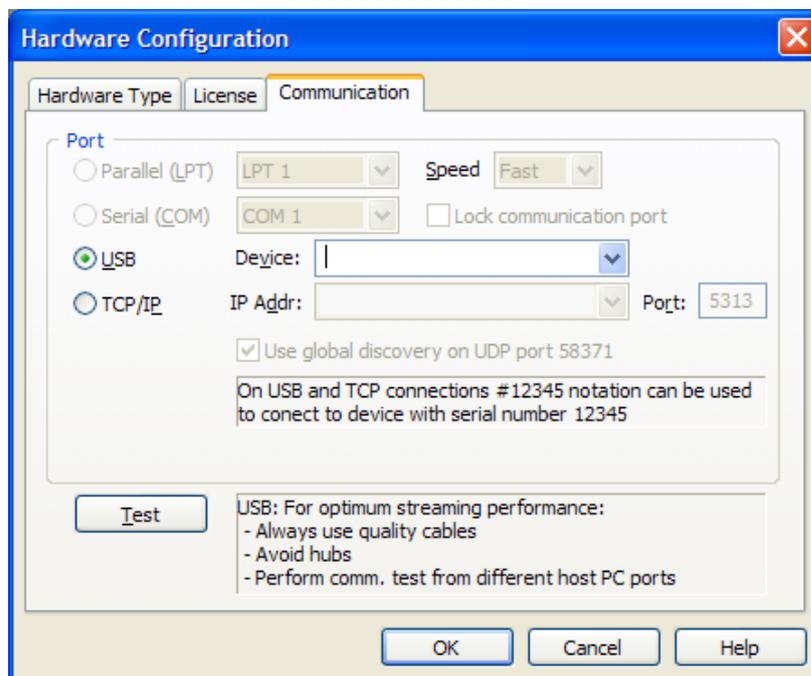
## Communication

iTAG.2K supports two types of communication: 10/100M Ethernet and USB 1.1/2.0. It is recommended to use USB2.0 interface since it provides the fastest transfer from the iTAG.2K development system to the PC where winIDEA IDE runs. This will guarantee a maximum performance of the iTAG.2K development system.

Select the 'iC5000, iTAG2k' selection in the Hardware Configuration dialog under the Hardware/Hardware... menu.



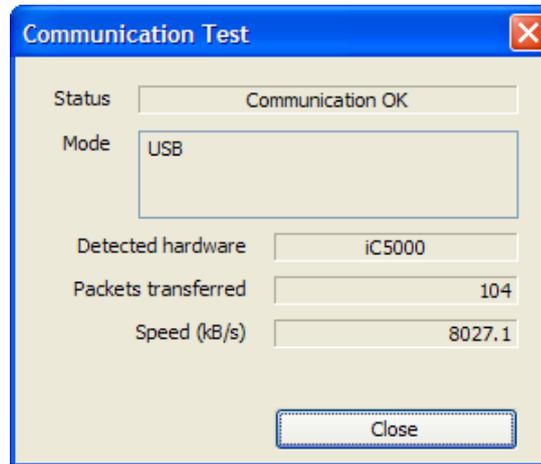
Specify the communication port through which the iTAG.2K unit connects to the PC in the Hardware/Hardware/Communication tab.



*Hardware Configuration dialog, Communication page*

- Universal Serial Bus (USB) - select when the Emulator is attached to the PC's USB port. The Emulator is selected in the Device pull-down menu. Note that iTAG.2K will enumerate as iC5000 on the USB port. When the Emulator is connected to the USB port of a computer for the first time, Windows will detect a new device and prompt you for the driver for it. Specify the path to the USB directory in the winIDEA installation directory or the distribution CD. **If only one emulator is connected to a PC via USB then Device combo box can be left empty (recommended). In this case if you exchange the emulator with another one, you don't have to change communication settings.**
- TCP/IP – This option sets the TCP/IP properties of the iTAG.2K. See 'Setting up TCP/IP communication' section for more details on TCP/IP setup.

Use the 'Test' button to test the communication settings.



*Communication test window*

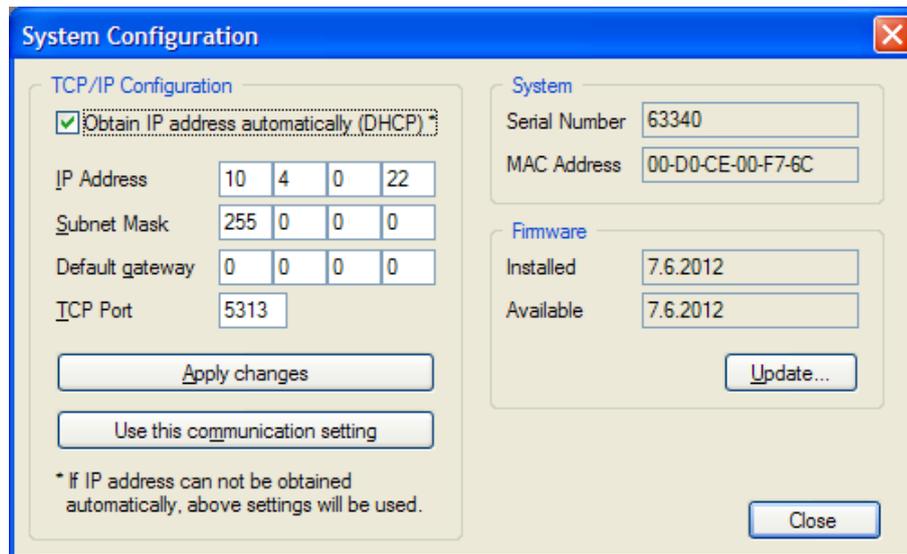
## Setting up TCP/IP communication

If the Emulator is connected using the Ethernet, its TCP/IP settings must be configured on both sides: the Emulator and in winIDEA.

More information on configuring the Emulator and winIDEA can be found in the Hardware User's Guide.

### First step: Configuring the Emulator

The Emulator must be connected using an alternate option – either through the serial port, through the parallel port or through USB. The connection must be set up in the 'Hardware/Communication' tab. Then, select the 'Hardware/Hardware Type' tab and click on the 'System Configuration...' button.



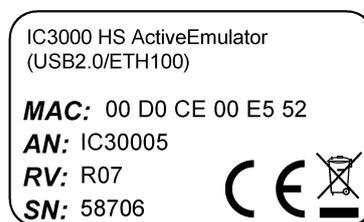
*System configuration options*

The TCP/IP settings can be obtained from the DHCP server on the network. If such a server is not available, the settings can be set manually. In this case, in the TCP/IP Configuration window, the IP Address, the Subnet Mask and the TCP Port must be specified. The default gateway address must be specified, if the Emulator is used via a gateway. The IP Address, available for the Emulator to use, the Subnet Mask and the default gateway, if needed, are usually defined by your network administrator. The TCP Port can be any port between 1024 and 65535, which is not already used. By default, the TCP port 5313 is used. For the information, if this port address could cause any conflicts and for an alternative port address, also contact your network administrator. When the correct settings are entered, click on the 'Apply changes' button. This writes the changes to the Emulator.

If you want winIDEA to use these settings, press the "Use this communication setting" button. Then, close the System configuration window.

The Emulator must be switched off and then on again in order for changes to take effect.

Emulator's MAC address is written on the same sticker where you will also find device serial number as it is shown on the next picture.

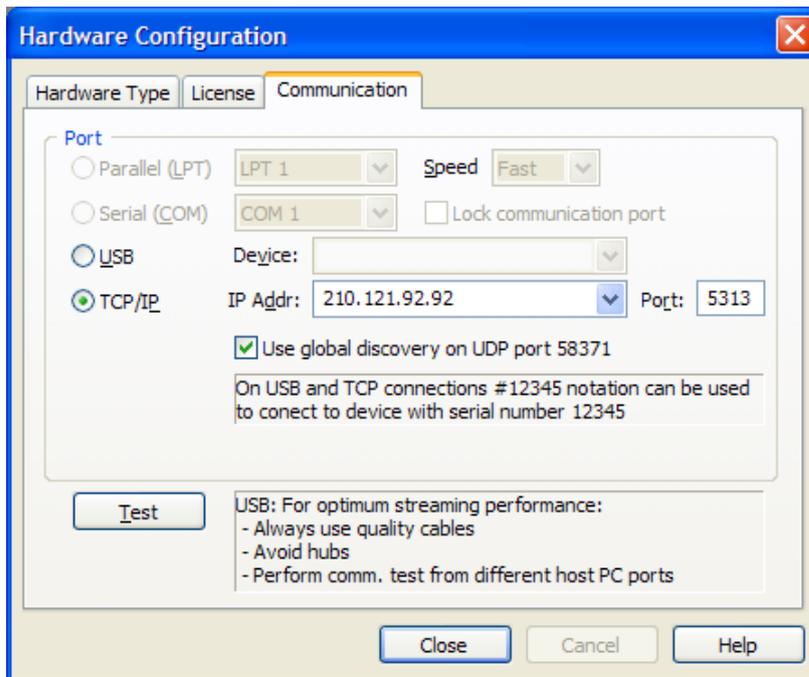


## Second step: Configuring

There are two ways of configuring TCP/IP in winIDEA: manually or by automatic discovery.

- *Manual Configuration*

Select the Hardware/Communication tab.

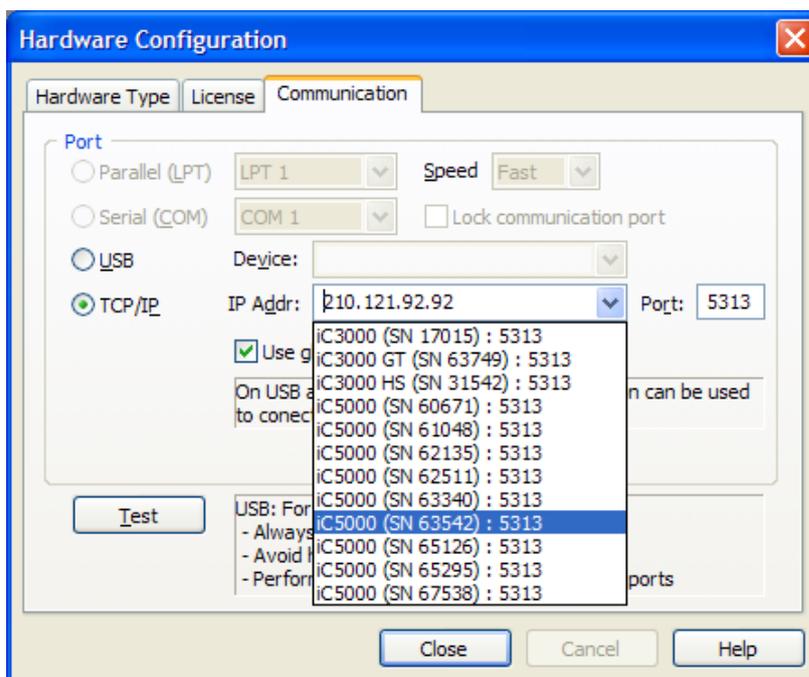


*Hardware/Communication tab*

If the “Use this communication setting” button was used when configuring the Emulator, these settings should already be set as required. If this option was not used, select the TCP/IP button and enter the IP Address and the TCP Port, as entered above into the Emulator. Connect the Emulator to the Ethernet, if not already connected, and click on the ‘Test’ button. The communication should be up and running.

- *Configuration with Automatic Discovery*

Select the Hardware/Communication tab.



*Display of discovered emulators*

First, select 'TCP/IP' type of communication. Then select the 'Use global discovery on UDP port 58371' and press the Refresh button.

In the pull-down window all emulators found on the network will be shown. The correct emulator can be identified by its serial number. Select the emulator and press the 'Test' button to ensure the communication is possible.

To be able to easier identify your own emulator, you can specify an unique port number in the first step (for example, as shown above, 23456; the number can be any number between 1024 and 65535, that is not already used on your network for other purposes – note that on the other hand more emulators can have the same port number), uncheck the 'Use global discovery' option, enter the port number, if the correct one is not entered already, and press the refresh button. Now, only the emulators on the network that use this port will be shown.

## Troubleshooting TCP/IP

If the communication test fails, there could be a problem with the IP Address, the Subnet Mask, the Default gateway address or the TCP Port.

First, make sure the Subnet Mask is correct. The subnet mask should be the same in the TCP/IP configuration of your computer and in the Emulator.

To find out the TCP/IP settings of your computer, open the command prompt and type 'ipconfig'. The computer will return something like this:

```
Ethernet adapter Local Area Connection:
    Connection-specific DNS Suffix . . :
    IP Address. . . . . : 210.121.92.121
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 210.121.92.65
```

Enter the same Subnet Mask and the Default Gateway data into Emulator.

Next, make sure, the IP Address is not already used by any other device. The easiest way to do that is to disconnect the Emulator from the Ethernet, open a command prompt and type in 'ping <ip\_address>' where <ip\_address> is the IP Address selected when configuring the Emulator, in the above example you would type in 'ping 210.121.92.92' (without quotes). The result should be 'Request timed out...'. If the result of the command is anything else (like 'Reply from...'), the IP is already taken and you should choose another one. If the result is correct, type in 'ping <ip\_address> -w 500 -t', in the above example this would mean 'ping 210.121.92.92 -w 500 -t'. This command pings the IP address every 500 milliseconds until you stop it with Control+C. You should constantly receive the information 'Request timed out'. Then, while the ping command is running, connect the Emulator and turn it on. Now, in a few moments, a ping reply should occur, in the form of 'Reply from <ip\_address>'... If this is not the case, the IP was set wrong. Try setting the IP again or select another one. If this is the case and the Emulator still cannot communicate with winIDEA, the TCP Port setting is wrong. Please select another port, set it up in the Emulator and in winIDEA and try again. When the ping is not more required, stop it using the keyboard shortcut Control+C.

If more Emulators are connected to the Ethernet and have the same IP set, only one will be active. Every Emulator must have a unique IP.

## Troubleshooting USB

During winIDEA installation USB driver is also installed. Very rarely after you power on the emulator which you connected to PC Windows show errors:

1. USB device not recognized
2. Cannot Install this Hardware.

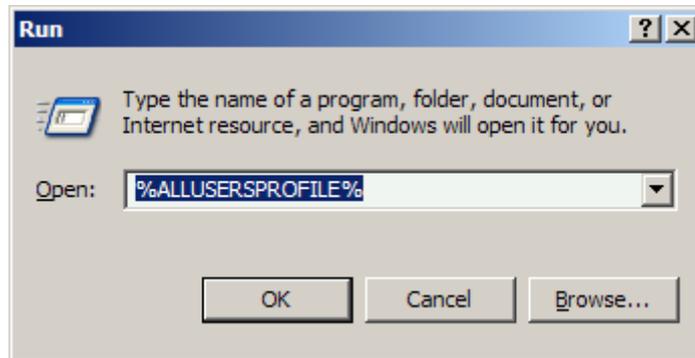
If first error is displayed you should:

- Check cable or use another USB cable.
- Connect emulator to another USB port

- Connect emulator to a different USB port. The one that resides on a PCI or PCIe card.
- Connect emulator to a PC via powered USB switch. In case a PC (usually a laptop) cannot provide enough power over USB port.

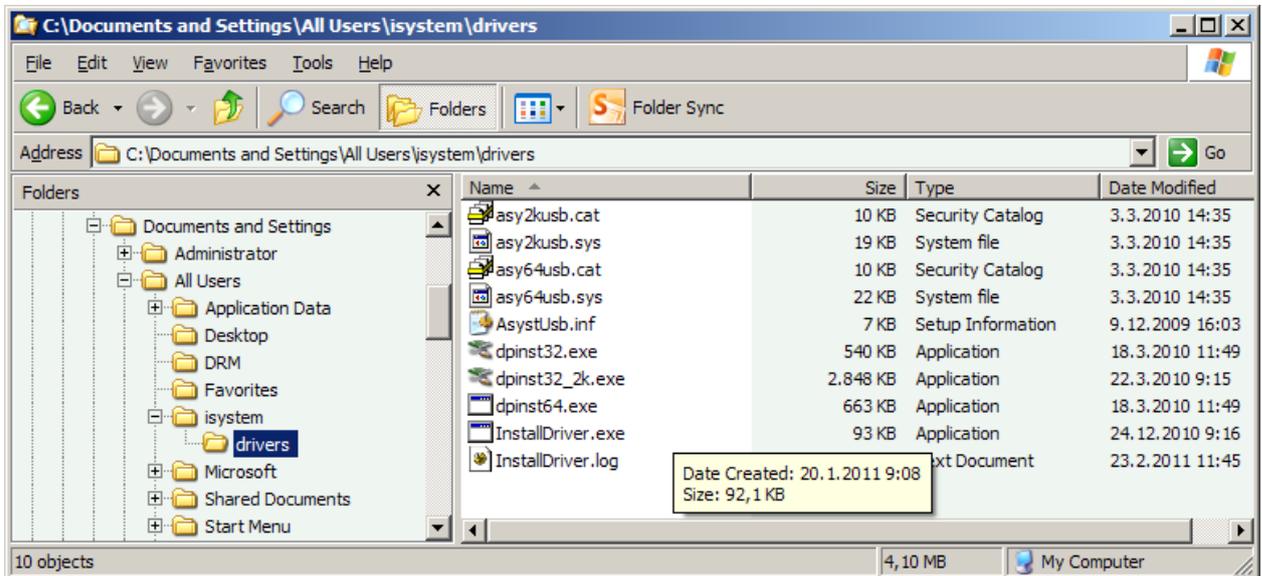
If second error is displayed or the above suggestions for the first error don't solve the problem you should reinstall the driver. Use the following procedure.

In Windows click on the start menu and select run, then type %ALLUSERSPROFILE% and press OK.



*Run Dialog which shows how to find folder with iSystem USB drivers*

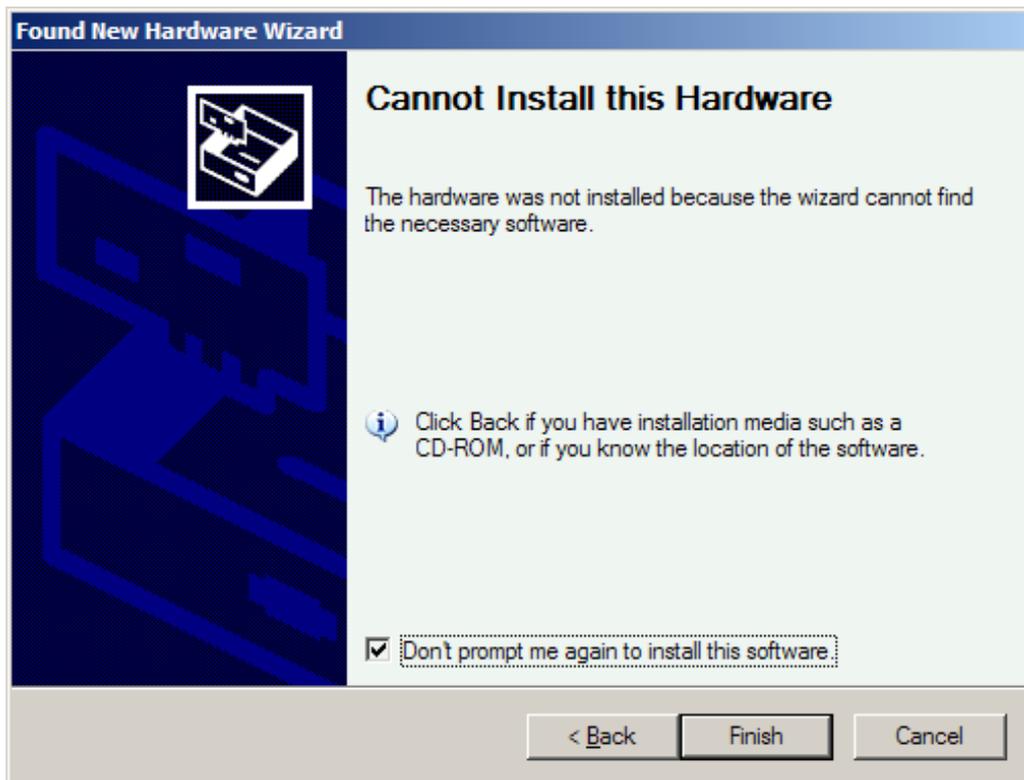
Windows explorer window will be shown. Open isystem\drivers folder and you should see the same files as on the below screenshot.



*iSystem USB drivers shown in Windows explorer*

Power down the emulator and double click on the InstallDriver.exe file which first installs new driver and then uninstalls older driver and removes registry entries which point to older driver.

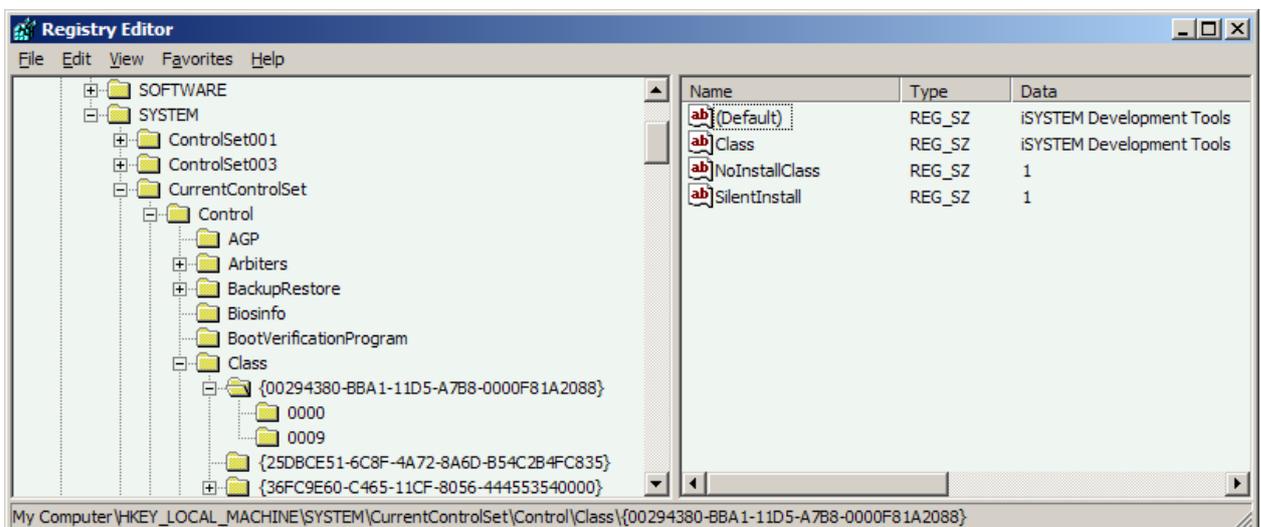
If Windows still shows “Cannot install this Hardware” dialog then manual driver uninstall procedure should be performed.



*Cannot Install this Hardware error message*

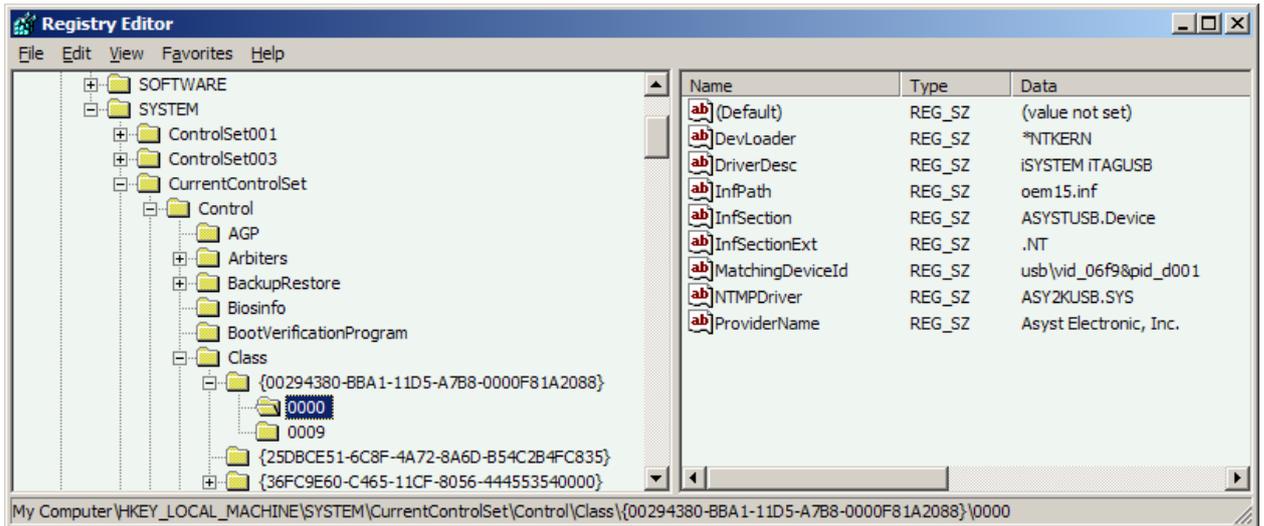
## Manual uninstallation of drivers

To manually uninstall driver you should first identify the files you must remove from the computer. From the Windows start menu select Run and type regedit and press the OK button. In regedit locate the key `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class` Press CTRL + F and enter iSystem in the search dialog and press OK.



*Regedit window*

- Regedit should display a key with several strings as shown above. Under this key there are also subkeys represented by numbers. Click on such a key and inside locate a string named InfPath which points to an INF file. Locate this INF file in `c:\Windows\Inf` folder and delete it. Repeat the procedure for all different INF files found in registry.



*Regedit window showing a key which points to an old USB driver*

- Then delete the main key (with all subkeys) which holds the data of all enumerated iSystem devices. In example shown above the key is {00294380-BBA1-11D5-A7B8-0000F81A2088}.
- At the end remove the folder where winIDEA installs the driver, for example:  
C:\Documents and Settings\All Users\isystem\drivers  
The folder path is different on windows 7 and Vista.
- Now, restart the PC and again install the iSystemDriverPack which you can find here:  
<http://www.isystem.si/SWUpdates/iSystemDriverPack.exe>

The above procedure can be executed only if you have administrator rights.

## Trace Line Calibration

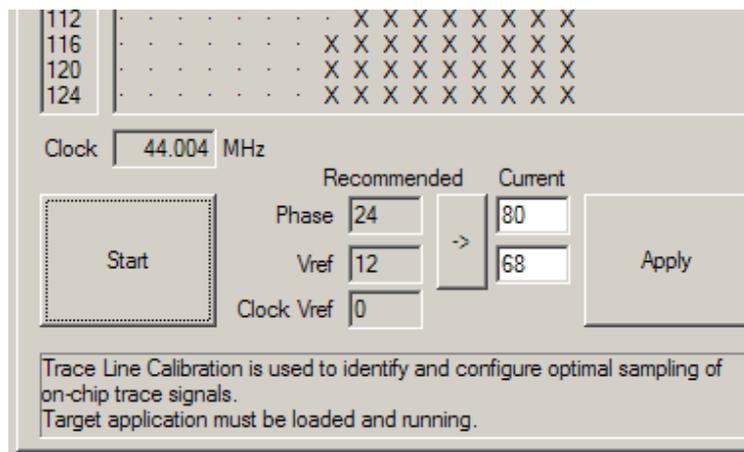
Majority of the modern embedded microcontrollers providing trace functionality, implements a so called message based trace port, where an individual trace message is broadcasted off the microcontroller through a relatively narrow physical trace port in multiple CPU cycles, at frequencies, which can be well over 100 MHz. Typically, the trace port is combined from trace data lines and a trace clock line, which is used to sample trace data lines on rising, falling or both edges (depending on the individual implementation).

At lower frequencies and good signal integrity we can consider the clock and data lines as pure digital signals, which are correctly phase aligned. As such, the external trace tool can capture them accurately without any problems.

Nowadays, capturing of the valid trace data becomes more and more challenging due to the various signal integrity issues (noise, skew, crosstalk, reflections, ground bounce...), which are introduced either due to the high frequency trace clock & data, due to the bad target PCB design or a combination of both. iTAG.2K has the ability to compensate for these issues via Trace Line Calibration functionality, which allows shifting threshold voltage and clock phase at the capture time of the trace data. When Trace Line Calibration is performed, it auto scans over these two dimensions and searches for valid and invalid settings and finds an optimum data eye.

### Example

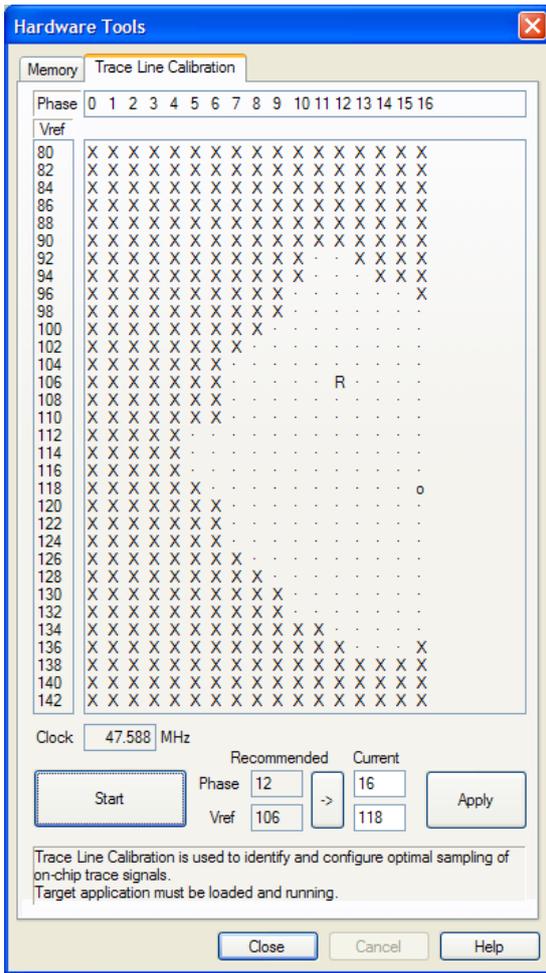
Let's assume we have a Cortex-M3 based NXP LPC1768 microcontroller running at 95 MHz. At this frequency, some of the signal integrity issues will show up for sure. After the debug download, the application should be run. Next, the "Start" button in the "Hardware/Tools/ Trace Line Calibration" should be pressed, which starts the auto-scan. After a couple of seconds, the result of the scan is collected and recommended "Vref" and "Phase" values are provided. Typically, the user just needs to press the "->" button to use the recommended values (or, if desired, enter them manually) and finally use the Apply button.



Configuration part of the Trace Line Calibration dialog

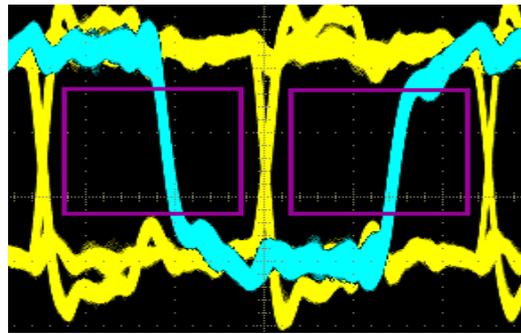
Newly applied values are stored upon Save Workspace and also used on the next debug download.

The following picture shows the result of the Trace Line Calibration and the corresponding timing view of signals on the trace port.



Trace Line Calibration window – scan has been performed and applied.

|   |                |
|---|----------------|
| X | invalid area   |
| . | valid area     |
| R | recommended    |
| o | currently used |



Good signal integrity at lower frequency with large “Data eyes”



Higher frequency: Valid “Data eyes” shown on upper data signal and how the clock (lower) must be delayed.

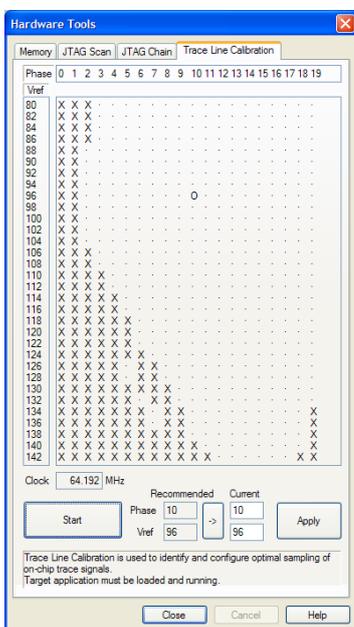
## Trace Port PCB Design Guidelines

This section contains some guidelines, which should be considered during the target PCB design to ensure the correct operation of the trace port (ETM, Nexus,...) and the external trace tool (iC5000, iTRACE GT). Note that the quality and timing of the trace port signals to the external trace tool are critical for correct and reliable trace operation.

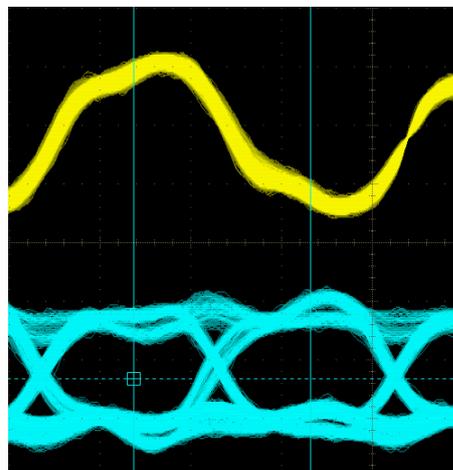
- All trace port lines on the PCB should be as short as possible (max ~2,5 cm),
- Traces should run on the same layer, or layers with the same impedance.
- Preferred layer impedance is 50 Ohm.
- Mictor ground pins should be connected directly to PCB's GND plane.
- Trace clock should be serially terminated by 47 Ohm resistor as close as possible to the driver. The value of the resistor may be changed depending on driver characteristics.
- Trace clock should be clean of crosstalk – if possible with double distance to closest nets.
- Trace clock should have only point-to-point connection – any stubs should be avoided.
- It is strongly recommended also for other (data) lines to be point-to-point only. If any stubs are needed, they should be as short as possible, when longer are required, there should be a possibility to optionally disconnect them (e.g. by jumpers).
- Trace port data bus inner crosstalk is not so important, but it is critical to isolate the whole bus from other signals (including from the trace port clock).

The following examples show, how the length of the trace lines is reflected in signal integrity and consequently in functionality. One of typical evaluation boards was used, where the CPU is located on the upper piggyback board, which fits to the lower, larger measurement board.

### Trace lines with short stubs

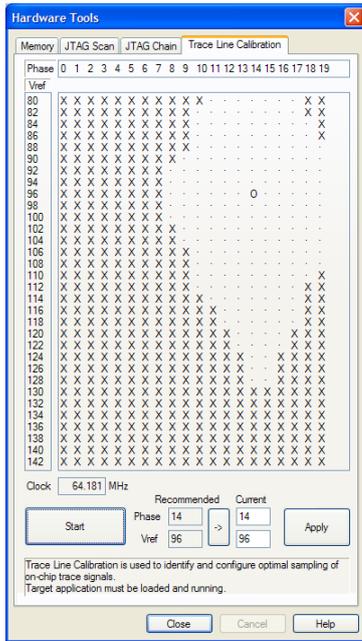


Trace Line Calibration result

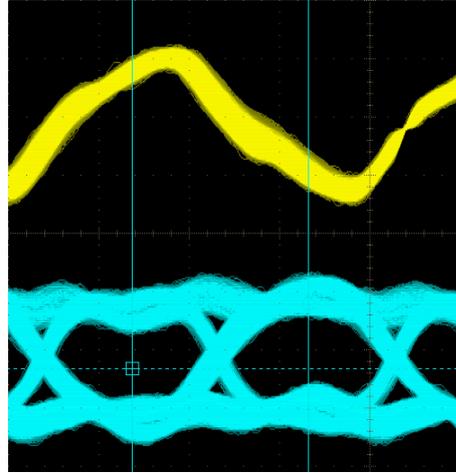


Measured by oscilloscope

Trace lines with longer stubs (over connector to other board)

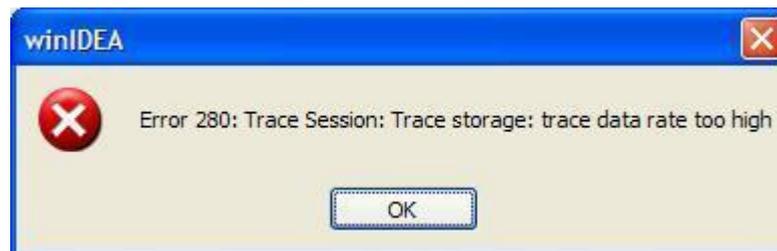


Trace Line Calibration result



Measured by oscilloscope

## Emulation Notes



Above message can occur when using trace. It indicates that the DDR (trace storage RAM) input FIFO, which accepts trace data from the system domain, has overflowed, and some portion of the trace data will be missing. It doesn't mean any hardware failure. Possible solutions:

- lower the target CPU clock
- increase Nexus clock divider, which yields lower Nexus clock, but at the same time Nexus is more prone to overflows then
- changing the trace port width e.g. from 16 bit to 12 bit or from 12 bit to 4 bit reduces the Nexus information bandwidth. Note that possible port size varies depending on the target CPU.
- IC50000-1 (ordering code) has higher trace storage bandwidth than IC50000.

## Debug Cable Adapters

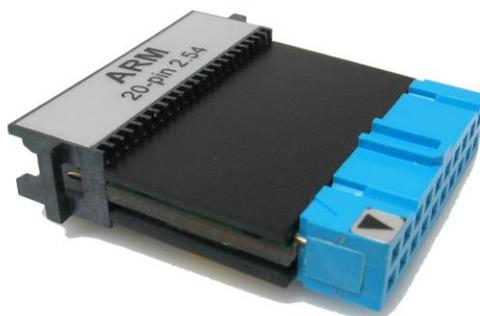
Various cable adapters are available depending on the specific target architecture and the target debug connector. They are used to connect iTAG.2K development system to the target.

Signal direction definition used throughout this document:

- O - output from the debugger to the target microcontroller
- I - input to the debugger from the target microcontroller

- **20-pin 2.54mm ARM Cable Adapter**

This adapter comes along the iTAG.2K unit.



This adapter is used to connect the iTAG.2K development system to the Cortex-A, Cortex-R, or ARM7, ARM9 based target. It can be used for targets featuring a 20-pin 2.54 pitch target debug connector with the ARM pinout.

The following pinout is valid on the target side:

| Signal direction | Signal description | Signal | Pin | Pin | Signal | Signal description | Signal direction |
|------------------|--------------------|--------|-----|-----|--------|--------------------|------------------|
| I                | Reference voltage  | VTref  | 1   | 2   | NC     | Not Connected      |                  |
| O                | Debug JTAG         | nTRST  | 3   | 4   | GND    | Ground             |                  |
| O                | Debug JTAG         | TDI    | 5   | 6   | GND    | Ground             |                  |
| O                | Debug JTAG         | TMS    | 7   | 8   | GND    | Ground             |                  |
| O                | Debug JTAG         | TCK    | 9   | 10  | GND    | Ground             |                  |
| I                | Return TCK         | RTCK   | 11  | 12  | GND    | Ground             |                  |
| I                | Debug JTAG         | TDO    | 13  | 14  | GND    | Ground             |                  |
| I/O              | System Reset       | nSRST  | 15  | 16  | GND    | Ground             |                  |
| O                | Debug request      | DBGRQ  | 17  | 18  | GND    | Ground             |                  |
| I                | Debug Acknowledge  | DBACK  | 19  | 20  | GND    | Ground             |                  |

*20-pin ARM pinout*

---

Note: 20-pin 2.54mm ARM Cable Adapter features resettable fuses on all pins except for pin 11 and 19. These fuses protect debug signals against overcurrent and cycle back to a conductive state after the excessive current fades away.

---

The adapter connects to the target via a 20-pin 2.54 mm connector (for example Yamaichi: FAS-2001-2101-2-0BF). A target should feature a matching part (for example WÜRTH ELEKTRONIK: 612 020 216 21).

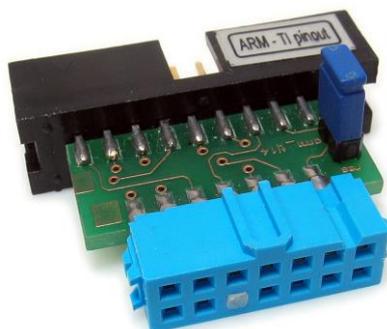
## Texas Instruments Microcontrollers

Note that targets based on Texas Instruments (TI) ARM microcontroller may feature Texas Instruments ARM 14-pin target debug connector with TI proprietary pinout.

A dedicated adapter is available for Texas Instruments ARM 14-pin pinout and can be ordered separately under the IAPIN20ARM14TI ordering code. Make sure you don't mix up Texas Instruments pinout with standard 14-pin 2.54mm ARM pinout (cable adapter IC50112).

Double check pinout of the target debug connector with the iTAG.2K cable adapter pinout before connecting iTAG.2K to the target for the first time.

|                      |                       |
|----------------------|-----------------------|
| <b>Ordering code</b> | <b>IAPIN20ARM14TI</b> |
|----------------------|-----------------------|



With this adapter, the following pinout is valid on the target side:

| Signal direction | Signal description | Signal | Pin | Pin | Signal | Signal description | Signal direction |
|------------------|--------------------|--------|-----|-----|--------|--------------------|------------------|
| Output           | Standard JTAG      | TMS    | 1   | 2   | nTRST  | Standard JTAG      | Output           |
| Output           | Standard JTAG      | TDI    | 3   | 4   | GND    | Ground             |                  |
| Input            | Reference voltage  | VTref  | 5   | 6   | n.c.   | Not connected      |                  |
| Input            | Standard JTAG      | TDO    | 7   | 8   | GND    | Ground             |                  |
| Input            | Return TCK         | RTCK   | 9   | 10  | GND    | Ground             |                  |
| Output           | Standard JTAG      | TCK    | 11  | 12  | GND    | Ground             |                  |
| Input            | Debug Acknowledge  | BERR   | 13  | 14  | nSRST  | System Reset       | In/Out           |

*ARM7 14-pin TI target connector*

A jumper is present on the adapter. If this jumper is set, the SYSTEM RESET line is connected to pin 14 on the target side. If SYSTEM RESET is not needed, then this jumper should be removed.

---

Note: this adapter can only be used with 20-pin 2.54mm ARM Cable Adapter

---

The adapter connects to the target via a 14-pin 2.54 mm connector (for example Yamaichi: FAS-1401-2101-2-0BF). A target should feature a matching part (for example WÜRTH ELEKTRONIK: 612 014 216 21).

- 20-pin 1.27mm Cortex Debug Cable Adapter

This adapter comes along the iTAG.2K unit.



Cortex Debug Adapter Board is used to connect the iTAG.2K development system to the Cortex-M (M1, M3, M4) based target. It can be used for targets featuring a 20-pin 1.27mm pitch target debug connector with Cortex-M pinout.

The following pinout is valid on the target side:

| Signal direction | Signal description | Signal    | Pin | Pin | Signal    | Signal description | Signal direction |
|------------------|--------------------|-----------|-----|-----|-----------|--------------------|------------------|
| I                | Reference voltage  | VTref     | 1   | 2   | SWDIO/TMS | SWD/JTAG           | I/O              |
|                  | Ground             | GND       | 3   | 4   | SWCLK/TCK | SWD/JTAG           | O                |
|                  | Ground             | GND       | 5   | 6   | SWO/TDO   | SWD/JTAG           | I                |
|                  | Ground             | GND       | 7   | 8   | NC/TDI    | SWD/JTAG           | O                |
|                  | Ground             | GND       | 9   | 10  | nSRST     | System Reset       | I/O              |
|                  |                    | NC_CAPGND | 11  | 12  | TRCLK     | Trace Clock        | I                |
|                  |                    | NC_CAPGND | 13  | 14  | TRD0      | Trace Data 0       | I                |
|                  | Ground             | GND       | 15  | 16  | TRD1      | Trace Data 1       | I                |
|                  | Ground             | GND       | 17  | 18  | TRD2      | Trace Data 2       | I                |
|                  | Ground             | GND       | 19  | 20  | TRD3      | Trace Data 3       | I                |

*20-pin Cortex-M pinout*

---

Note: 20-pin 1.27mm Cortex-M Cable Adapter features resettable fuses on pins 1, 2, 4, 6, 8 and 10. These protect debug signals against overcurrent and cycle back to a conductive state after the excessive current fades away. Signals on pins 12, 14, 16, 18 and 20 are protected via 47 ohm serial resistors.

---

The adapter connects to the target via a 20-pin 1.27mm connector (for example SAMTEC: FFSD-10-01-N). A target should feature a matching part (for example SAMTEC: FTSH-110-01-F-DV-K).

- 10-pin 1.27mm Cortex Debug Cable Adapter

This adapter comes along the iTAG.2K unit.



This adapter is used to connect the iTAG.2K development system system to the Cortex-M (M0, M0+, M1, M3, M4) based target. It can be used for targets featuring a 10-pin 1.27mm pitch target debug connector with the Cortex-M pinout.

The following pinout is valid on the target side:

| Signal direction | Signal description | Signal | Pin | Pin | Signal    | Signal description | Signal direction |
|------------------|--------------------|--------|-----|-----|-----------|--------------------|------------------|
| I                | Reference voltage  | VTref  | 1   | 2   | SWDIO/TMS | SWD/JTAG           | I/O              |
|                  | Ground             | GND    | 3   | 4   | SWCLK/TCK | SWD/JTAG           | O                |
|                  | Ground             | GND    | 5   | 6   | SWO/TDO   | SWD/JTAG           | I                |
|                  | Ground             | GND    | 7   | 8   | NC/TDI    | SWD/JTAG           | O                |
|                  | Ground             | GND    | 9   | 10  | nSRST     | System Reset       | I/O              |

*10-pin Cortex-M pinout*

---

Note: 10-pin 1.27mm Cortex-M Cable Adapter features resettable fuses on pins 1, 2, 4, 6, 8 and 10. These fuses protect debug signals against overcurrent and cycle back to a conductive state after the excessive current fades away.

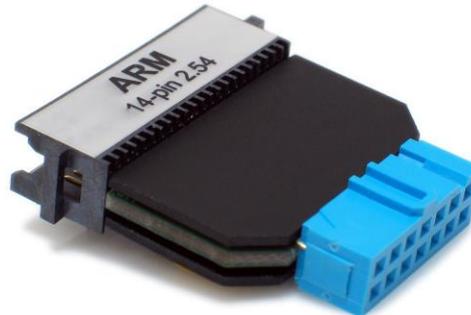
---

The adapter connects to the target via a 10-pin 1.27mm connector (for example SAMTEC: FFSD-05-01-N). A target should feature a matching part (for example SAMTEC: SHF-105-01-L-D-TH).

- 14-pin 2.54mm ARM Cable Adapter

This adapter can be ordered separately.

|                      |                |
|----------------------|----------------|
| <b>Ordering code</b> | <b>IC50112</b> |
|----------------------|----------------|



IC50112 ARM Cable Adapter is used to connect the iTAG.2K development system to the ARM7/ARM9 based target. It can be used for targets featuring a 14-pin 2.54 pitch target debug connector with ARM pinout.

The following pinout is valid on the target side:

| Signal direction | Signal description | Signal | Pin | Pin | Signal | Signal description | Signal direction |
|------------------|--------------------|--------|-----|-----|--------|--------------------|------------------|
|                  | Not Connected      | NC     | 1   | 2   | GND    | Ground             |                  |
| O                | Debug JTAG         | nTRST  | 3   | 4   | GND    | Ground             |                  |
| O                | Debug JTAG         | TDI    | 5   | 6   | GND    | Ground             |                  |
| O                | Debug JTAG         | TMS    | 7   | 8   | GND    | Ground             |                  |
| O                | Debug JTAG         | TCK    | 9   | 10  | GND    | Ground             |                  |
| I                | Debug JTAG         | TDO    | 11  | 12  | nSRST  | System Reset       | I/O              |
| I                | Reference voltage  | VTref  | 13  | 14  | GND    | Ground             |                  |

*14-pin ARM pinout*

---

Note: 14-pin 2.54mm ARM Cable Adapter features resettable fuses on all pins. These fuses protect debug signals against overcurrent and cycle back to a conductive state after the excessive current fades away.

---

The adapter connects to the target via a 14-pin 2.54 mm connector (for example Yamaichi: FAS-1401-2101-2-0BF). A target should feature a matching part (for example WÜRTH ELEKTRONIK: 612 014 216 21).

- Mictor 38-pin ARM ETM 8-bit Cable Adapter

This adapter can be ordered separately.

|                      |                |
|----------------------|----------------|
| <b>Ordering code</b> | <b>IC50114</b> |
|----------------------|----------------|



This adapter is used to connect the iTAG.2K development system to the ARM7/ARM9 based target. It can be used for targets featuring a Mictor 38-pin target debug connector with the ARM7/ARM9 ETM pinout.

Only 4 or 8-bit ETM port width (physical port size) is supported with this adapter. However, it can be also used for the ETM trace on targets where physically more than 8 ETM data lines are connected to the target debug connector by configuring the on-chip ETM module for operation with 8 or less data lines ('Hardware/Analyzer Setup/ETM' tab). Note that ETM port bandwidth proportionally drops off when less ETM data lines are used and trace overflows are more likely to occur.

If maximum bandwidth is needed and CPU provides 16-bit ETM port, use the ARM ETM 16-bit cable adapter (IC50115) in conjunction with the iC5000 development system. iTAG.2K supports only 8-bit ETM trace port.

The following pinout is valid on the target side:

| Signal direction | Signal | Pin | Pin | Signal      | Signal direction |
|------------------|--------|-----|-----|-------------|------------------|
|                  | NC     | 1   | 2   | NC          |                  |
|                  | NC     | 3   | 4   | NC          |                  |
|                  | NC     | 5   | 6   | TRACECLK    | I                |
|                  | NC     | 7   | 8   | NC          |                  |
| O                | nSRST  | 9   | 10  | NC          |                  |
| I                | TDO    | 11  | 12  | VTref       | I                |
|                  | NC     | 13  | 14  | NC          |                  |
| O                | TCK    | 15  | 16  | TRACEPKT[7] | I                |
| O                | TMS    | 17  | 18  | TRACEPKT[6] | I                |
| O                | TDI    | 19  | 20  | TRACEPKT[5] | I                |
| O                | nTRST  | 21  | 22  | TRACEPKT[4] | I                |
|                  | NC     | 23  | 24  | TRACEPKT[3] | I                |
|                  | NC     | 25  | 26  | TRACEPKT[2] | I                |
|                  | NC     | 27  | 28  | TRACEPKT[1] | I                |
|                  | NC     | 29  | 30  | TRACEPKT[0] | I                |
|                  | NC     | 31  | 32  | TRACESYNC   | I                |
|                  | NC     | 33  | 34  | PIPESTAT[2] | I                |
|                  | NC     | 35  | 36  | PIPESTAT[1] | I                |
|                  | NC     | 37  | 38  | PIPESTAT[0] | I                |

*8-bit ARM ETM target pinout*

Blue colored signals are required for trace.

---

Note: Mictor 38-pin ARM ETM Cable Adapter features resettable fuses on pins 9, 11, 12, 15, 17, 19 and 21. These protect debug signals against overcurrent and cycle back to a conductive state after the excessive current fades away. Signals on pins 6, 16, 18, 20, 22, 24, 26, 26, 28, 30, 32, 34, 36 and 38 are protected via 47 ohm serial resistors.

---

The adapter connects to the target via a 38-pin Mictor connector (Tyco Electronics 5767055-1). A target should feature a matching part (for example Tyco Electronics 5767081-1 in SMT technology).

## ***Troubleshooting***

Make sure you read the technical notes document for your target microcontroller architecture before contacting iSYSTEM technical support. It contains all the information related to the debugging including some troubleshooting tips.

According technical notes document gets open if you select Emulation Technical Notes under Help menu, considering that winIDEA workspace is configured for your target microcontroller.

OCD Cortex M0-M1 and OCD Cortex M3-M4 technical notes documents are also available for download at [www.isystem.com](http://www.isystem.com) under Supported MCUs.

---

Disclaimer: iSYSTEM assumes no responsibility for any errors which may appear in this document, reserves the right to change devices or specifications detailed herein at any time without notice, and does not make any commitment to update the information herein.

**© iSYSTEM. All rights reserved.**