TRACE32 Installation Guide

Version 18-Nov-2020

02-Jul-19	New section “Parameters for PBI Drivers (Software-Only Solutions for Front-End plus Back-End)”.

WARNING:	To prevent debugger and target from damage it is recommended to connect or disconnect the debug cable only while the target power is OFF. Recommendation for the software start: 1. Disconnect the debug cable from the target while the target power is off. 2. Connect the host system, the TRACE32 hardware and the debug cable. 3. Power ON the TRACE32 hardware. 4. Start the TRACE32 software to load the debugger firmware. 5. Connect the debug cable to the target. 6. Switch the target power ON. 7. Configure your debugger e.g. via a start-up script. Power down: 1. Switch off the target power. 2. Disconnect the debug cable from the target. 3. Close the TRACE32 software. 4. Power OFF the TRACE32 hardware.

Important Information Concerning the Use of the TRACE32 Development System

Due to the special nature of the TRACE32 development system, the user is advised that it can generate higher than normal levels of electromagnetic radiation which can interfere with the operation of all kinds of radio and other equipment.

To comply with the European Approval Regulations therefore, the following restrictions must be observed:

1. The development system must be used only in an industrial (or comparable) area.
2. The system must not be operated within 20 metres of any equipment which may be affected by such emissions (radio receivers, TVs etc).

Prerequisites

TRACE32 supports these host computers and operating systems:

Host	OS	Company	Comment
AXP-STATION			
AXP-STATION	DIGITAL UNIX	DEC	Motif (SCU based SW only)
AXP-STATION	VMS/AXP 1.5	DEC	Motif (SCU based SW only)
HP-9000/700			
HP-9000/700	HP-UX 8.0	HP	Motif (SCU based SW only)
HP-9000/700	HP-UX 9.0	HP	CDE (SCU based SW only)
HP-9000/700	HP-UX 10.X	HP	CDE (SCU based SW only)
MACINTOSH			
MACINTOSH	LINUX/PPC	-	Motif/Lesstif
MACINTOSH	MAC OS-X/X86	Apple Inc.	Motif
MACINTOSH	MAC OS-X/X86	Apple Inc.	Qt
PC			
PC	WINDOWS XP	Microsoft Corporation	32 bit
PC	WINDOWS VISTA	Microsoft Corporation	32/64 bit
PC	WINDOWS 7	Microsoft Corporation	32/64 bit
PC	WINDOWS 8	Microsoft Corporation	32/64 bit
PC	WINDOWS 10	Microsoft Corporation	64 [32] bit
PC	LINUX	-	32/64 bit, Motif/Lesstif
PC	LINUX	-	32/64 bit, Qt
SPARC			
SPARC	SOLARIS 2.3	SUNSOFT	Open Windows or Motif
SPARC	SOLARIS 2.X	SUNSOFT	CDE
VAX-STATION			
VAX-STATION	VMS/VAX 5.5	DEC	Motif (SCU based SW only)

There are three different types of debugging:

- **Host-based**: TRACE32/PowerView runs on the host (e.g. a PC or Unix Workstation) and handles most of the user interaction and processing. At the start of a debug session, time-critical, target-related communication software is transferred to a POWER tool and then run there.

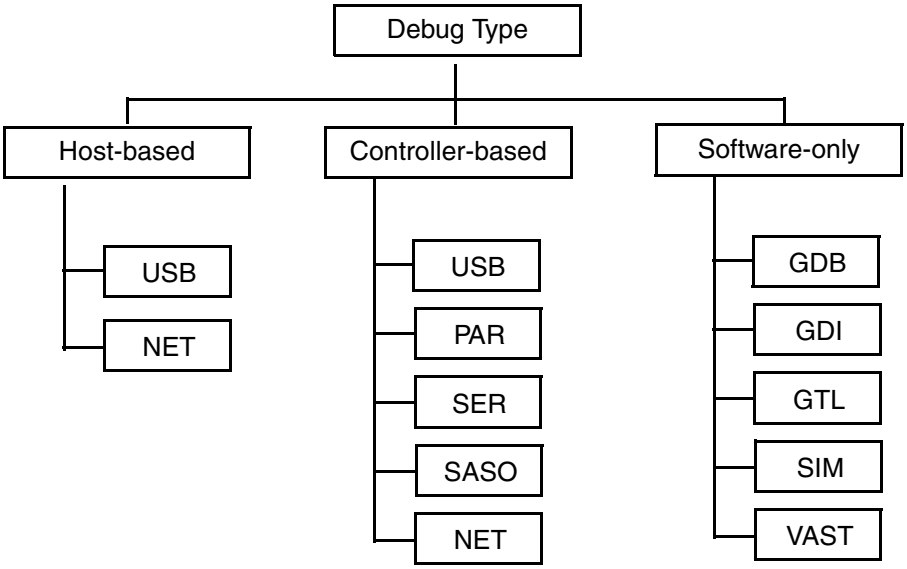
  Most customers use Host-based In-Circuit-Debugging (TRACE32-ICD) and tracing.

- **Controller-based**: TRACE32 software runs mostly on the SCU or PODBUS ETHERNET CONTROLLER (PODETH) unit. The system program and the target-related communication software must first be linked together and then downloaded to the SCU/PODETH unit. The host system (e.g. a PC or Unix Workstation) only runs a GUI interface program.

  This debugging type is mainly used for In-Circuit-Emulation (TRACE32-ICE).

  With PCs and Workstations sufficiently powerful for Host-based debugging, not many customers use a POWER ETHERNET CONTROLLER with a PODBUS-connected POWER DEBUG tool for Controller-based In-Circuit Debugging (TRACE32-ICD).

- **Software-only:** TRACE32 PowerView GUI is used as a debug front-end, or in simulation mode, and in some operation modes also includes the Back-End. No Lauterbach TRACE32 hardware is required (except in some cases for the licensing mechanism).



Host-based Interfaces

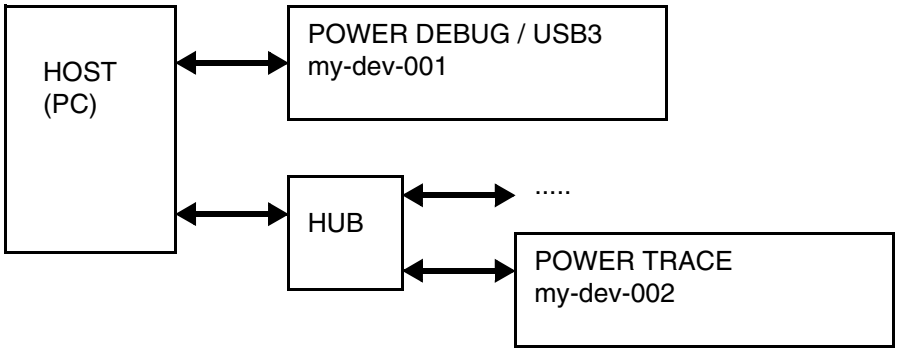
This chapter describes the host-based USB and Ethernet configurations. These types of configurations are commonly used for debugging and tracing.

USB Interface (TRACE32-USB)

The Universal Serial Bus (USB) is a standardized serial bus designed to connect peripherals to Personal Computers. The tiered-star topology allows simultaneous connection of up to 127 devices on the bus. Windows and Linux distributions provide full USB support.

If you need to debug with more than one TRACE32 POWER device, you can address these by using their **device name** in the USB configuration. In the TRACE32/PowerView GUI (abbreviated 'TRACE32' in this manual), you can set the device name in a dialog box that you open via the **Misc** menu > **Interface Config** (or via the TRACE32 command line using the **IFCONFIG.state** command).

Once you have set a device name, the option `NODE=<device_name>` in config.t32 tells TRACE32 to connect with the specific 'named USB device' that you want.



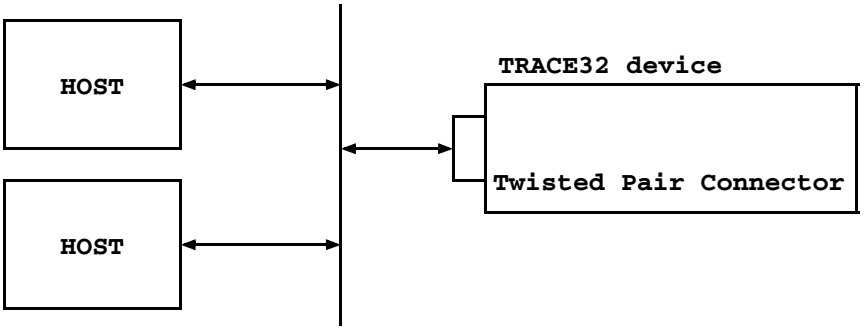
Ethernet is the physical standard for all connections to workstations or network-based PC configurations. The protocols UDP, ICMP (for ping), ARP, RARP and DHCP are supported by the TRACE32 device.

The Internet address of the device can be configured manually (e.g. using the USB Interface), with an RARP or DHCP server, or by using the `arp -s` command before making the first connection. It is not necessary to enter router information or a subnet mask.

The TRACE32 device contains a DHCP client to obtain its IP address from a DHCP server (your DHCP admin usually needs to configure this relation).

In TRACE32, you can set the device name in a dialog box that opens via the **Misc** menu > **Interface Config** (or via the TRACE32 command line using the **IFCONFIG.state** command). When your DHCP admin sets up a relationship for this name and a suitable IP address for your network, the DHCP server forwards this name-to-address information to the DNS server. With the DNS server 'knowing' the device name and IP address, TRACE32 can later use `NODE=<device_name>` in config.t32 to connect with the device.

Alternatively you can also use the **IFCONFIG.state** dialog to set the IP address manually. Then `NODE=<address>` points TRACE32 to your device.



Controller-based Interfaces

With the shift from in-circuit emulation to boundary-scan (JTAG) based debugging, controller-based interfaces have become much less common. Therefore, controller-based interfaces are not discussed here.

Minimal Manual Setup (no regular installation!)

This chapter describes a minimal manual setup of the TRACE32 software and hardware for the most widespread combinations of devices and operating systems. Please keep in mind that a manual setup may be error-prone and may not work in all debug environments. After all, it is not the installer which installs all files, but ***you*** who copies just the minimum number of files.

Even if you do not plan to implement a minimal manual setup, this little tour behind the scenes of TRACE32 helps you gain a better understanding of how to tailor the debug system to your needs.

No one likes a black-box environment - so let's switch on the light!

Use cases where a minimal manual setup makes sense:

- Help Lauterbach Support reproduce and solve any debug issue by quickly providing them with copies of your source and debug files, scripts and TRACE32 files in a minimal software setup (a.k.a. test-tube installation). Help us help you!
- Bundle your demo scripts and other files with a minimal TRACE32 software setup to form a self-contained demonstration kit for use at a trade fair or in a training session.
- Make portable on a USB stick - always a nice-to-have.
- In a space-restricted environment, e.g. a small Virtual Machine image.

For a regular installation:

- For Windows, a TRACE32 DVD installer is available.
  See **"MS Windows"** (icd_quick_installation.pdf).
- For Linux, see **"PC_LINUX"** (icd_quick_installation.pdf).

NOTE:	For a regular installation, always use the TRACE32 DVD installer. Currently there is only a TRACE32 DVD installer for Windows, but newer versions of this can also generate a file tree for Linux systems.

Prerequisites and Recommendations for the Minimal Manual Setup

Windows

The minimal manual setup for Windows requires:

- Access to an administrator account for driver installation
- Permission to create folders and files
- Permission to execute files from these locations

Linux

The minimal manual setup for Linux requires:

- Administrator rights

The TRACE32 PowerView GUI for Linux is available in two variants:

- **Qt GUI**

  If you choose to install a Linux Qt variant of TRACE32 (e.g. t32marm-**qt**), then you can skip the font installation.

- **Motif GUI**

  TRACE32 for Linux uses Motif libraries for the GUI. These require installation of the TRACE32 fonts. For information about the font installation, see **"Motif GUI specific steps"** in ICD Quick Installation, page 31 (icd_quick_installation.pdf).

Non-TRACE32 Hardware

- USB cable (even if you want to ultimately use an Ethernet configuration, for easy device setup you will temporarily want to have a USB cable)
- If you want to set up an Ethernet configuration, you will need either:
  - an Ethernet cross-over cable, or
  - an Ethernet hub or switch and two patch cables

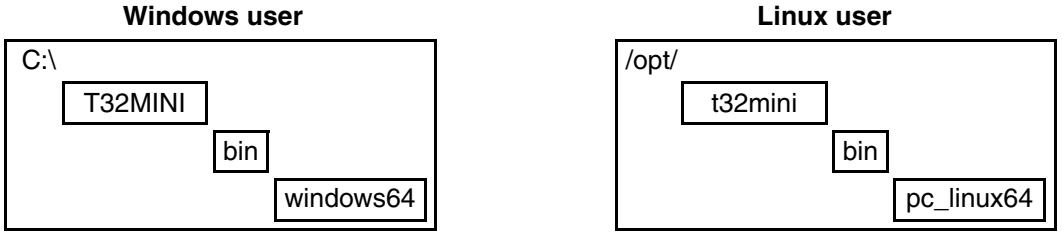
Copy the Required Files (USB and Ethernet)

The following step-by-step description is for a minimal manual setup of TRACE32 (64-bit version) on Windows and Linux. The file set copied from the TRACE32 installation DVD to a local folder structure is the minimal ARM debugging file set required for debugging PandaBoard.

The minimal debugging file set is identical for USB and Ethernet, only the content of the configuration file `config.t32` differs.

To copy the required files to their destination folders:

1. Create this folder structure:



2. From the TRACE32 installation DVD, copy these files into the **T32MINI** folder.

File	DVD Location	Description
`fcc.t32`	...\files	Low-level TRACE32 device communication software
`fccarm*.t32`	...\files	Running on the TRACE32 device, debug support for one or more cores
`help.t32`	...\files	Index, full-text search database, error/warning messages, and tooltips
`t32.men`	...\files	Top-level English menu file, which also initializes additional toolbar buttons.
`t32font.fon`	...\files	TRACE32 system font (Windows only)

3. Copy these files to the destination folder (**windows64** or **pc_linux64**):

   windows64 folder:

File	DVD Location	Description
`t32marm.exe` + `t32screenwin.dll`	...\files\bin\windows64	TRACE32 for Windows (TRACE32/PowerView GUI, 64-bit version)

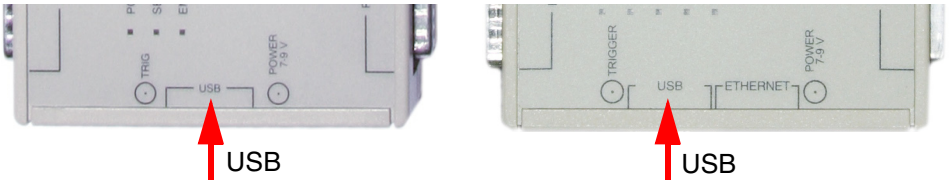
pc_linux64 folder:

t32marm-qt **+** t32screenqt5.so t32screenqt4.so	...\files\bin\pc_linux64	TRACE32 for Linux (64-bit version) • A file name suffix **-qt** marks the Qt GUI variant of the TRACE32/PowerView GUI. • The Motif GUI variant has no such suffix.

Reason for placing the binary into this **bin\<os>** structure: it resembles the file structure in our update archives. Then, when updating your minimal installation, you just need to unzip one file.

USB Configuration

The USB label is printed on USB-capable TRACE32 devices.

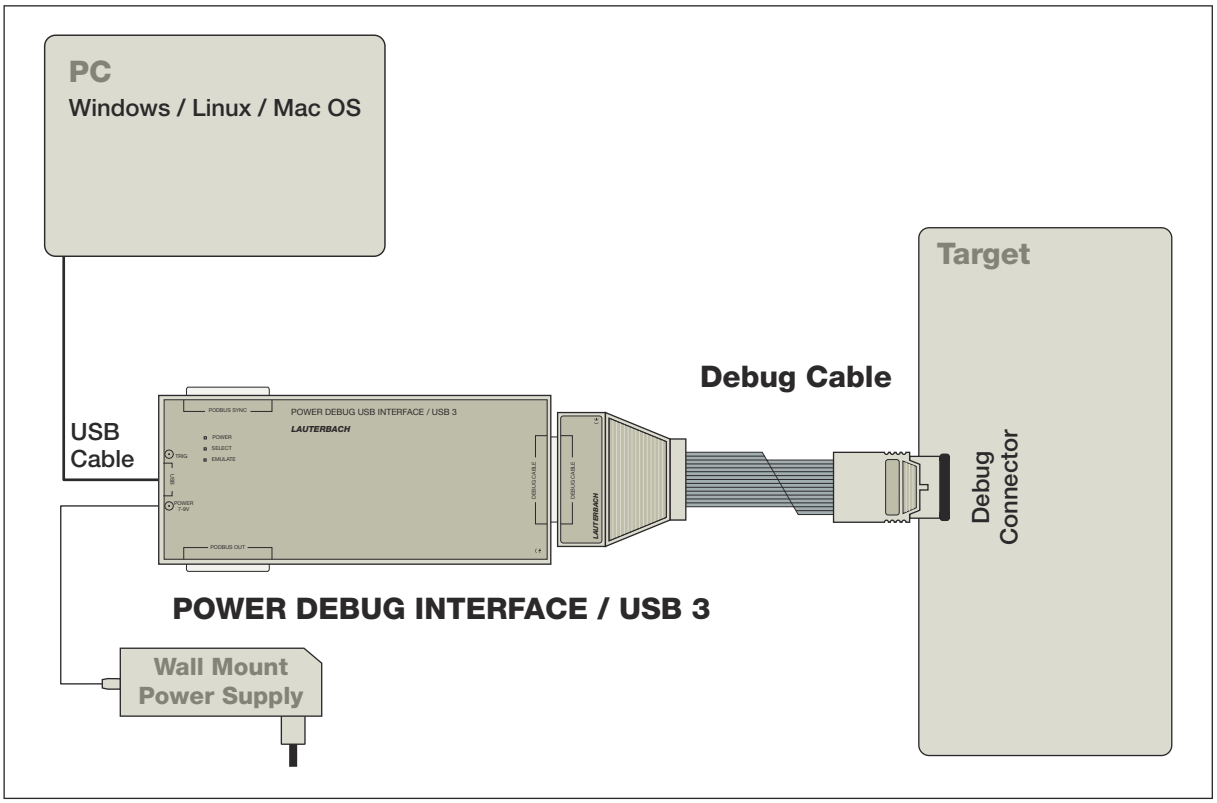


Typical devices include:

- POWER DEBUG INTERFACE / USB3, POWER DEBUG PRO,
  POWER DEBUG INTERFACE / USB2, POWER DEBUG INTERFACE / USB
- µTRACE
- POWER DEBUG II
- POWER TRACE / ETHERNET

Set up the Hardware (USB)

1. Connect the target-specific DEBUG CABLE to the POWER DEBUG INTERFACE / USB3.
2. Plug a USB cable into the POWER DEBUG INTERFACE / USB3 and the other end into a free USB port on your Windows / Linux PC (or Workstation).
3. Connect the power adaptor that came with your POWER device to mains and to the device.



Windows (USB)

Prerequisite for Windows is the installation of the “TRACE32 PODBUS USB driver for Windows” if not already installed from the standard TRACE32 software installer on the DVD.

1. On the TRACE32 installation DVD, please navigate to this file:

File	DVD Location	Description
`setup_t32_usb_driver.exe`	...\files\bin\windows64\drivers	USB driver installer

2. To install, double-click `setup_t32_usb_driver.exe`, and then follow the instructions of the installer.

Linux (USB)

For Linux, you don’t need to install any drivers, but you need to make sure that the USB devices have the proper name and access rights. The access rights and name are usually configured by **udev**. Please see the installation directory on the Lauterbach TRACE32 DVD for notes and a sample **udev rules file**.

File	DVD Location	Description
`readme.txt`	.../files/bin/pc_linux64/udev.conf	Instructions on how to configure Linux to make Lauterbach USB devices available

If you want to set up an Ethernet configuration on Windows or Linux

For Ethernet, you don’t need any drivers, but you need to make the device “visible” via a name or an IP address. For this, you need to set up a device name and configure your DHCP server.

(a) If you know ARP and can set up DHCP or configure a local hosts file, you can use this knowledge to map the MAC address of the TRACE32 POWER device (printed on the device) to an IP address and/or host name and then use this as the `NODE=` part of the configuration file.

(b) **An easier way** for the device configuration (to set up a device name or IP address in a TRACE32 device) is to use USB:

- For Windows + Ethernet, please install the USB driver. See **“Windows (USB)”**, page 15.
- For Linux + Ethernet, please configure your udev system. See **“Linux (USB)”**, page 15.

1. In the **windows64** or **pc_linux64** folder, right-click and create a file named `config.t32`.
2. Open the `config.t32` file with an ASCII editor.
3. Copy and paste one of the example configurations into the `config.t32` file:
   - For Windows, use **this example**.
   - For Linux, use **this example**.
4. Edit your `config.t32` file - i.e. use your own paths - and assign the ID you want (see `ID=`).
5. Save your `config.t32` file and close it.
   - If you want a USB configuration, please continue at **"Identify the Peripheral File"**, page 21.
   - If you want an Ethernet configuration, please continue at **"Ethernet Configuration"**, page 17.

Windows:

```
// TRACE32 configuration file for USB (Windows)
OS=
ID=T32TEST001
TMP=C:\temp                     ; temporary directory for TRACE32
SYS=C:\T32MINI                  ; system directory for TRACE32
HELP=G:\SERIAL\CD\files\pdf     ; help directory for TRACE32

PBI=
USB

PRINTER=WINDOWS
```

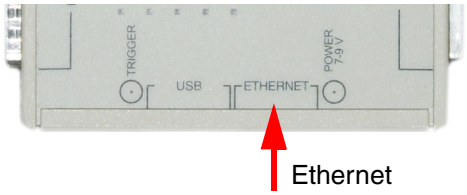
Linux:

```
// TRACE32 configuration file for USB (Linux)
OS=
ID=T32TEST001
TMP=/var/tmp                     ; temporary directory for TRACE32
SYS=/opt/t32mini                 ; system directory for TRACE32
HELP=/media/TRACE32/files/pdf    ; help directory for TRACE32

PBI=
USB
```

NOTE: For more complex configuration examples, refer to **"Parameters for the PBI Driver with LAUTERBACH Tools"**, page 55.

NOTE:	The **easiest way** to set the device name for an Ethernet configuration is to start with a USB connection. So, please continue at **"Copy the Required Files (USB and Ethernet)"**, page 12. Then complete the rest of this chapter.

The ETHERNET label is printed on Ethernet-capable TRACE32 devices.



Ethernet

Typical devices include:

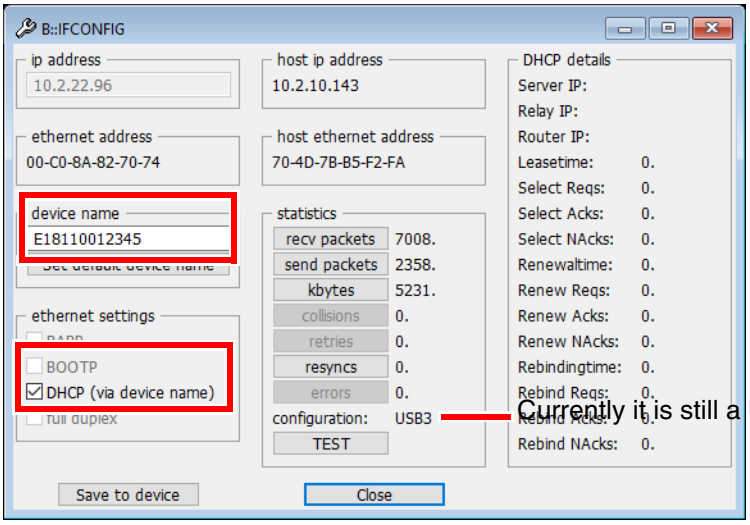
- POWER DEBUG PRO
- POWER DEBUG II
- POWER TRACE / ETHERNET

Changing an existing TRACE32 USB configuration to a TRACE32 Ethernet configuration involves these main steps:

- Assign a host name to the TRACE32 device.
- Modify the configuration file to change the connection from USB to Ethernet.
- Power-off the TRACE32 device.
- Disconnect USB from the TRACE32 device.
- Connect the TRACE32 device via an Ethernet cable.
- Power-on the TRACE32 device.

These steps are explained step by step in the sections below.

1. Make sure your TRACE32 device is connected via USB to your PC or Unix workstation, and you have a configuration file that is properly set up for USB.
2. Power on the TRACE32 hardware.
3. Start the TRACE32 PowerView GUI by double-clicking the TRACE32 executable in the **windows64** or **pc_linux64** folder.
4. From the **Misc** menu, select **Interface Config** to open the **IFCONFIG** window.
5. In the entry field below **device name**, type the host name you want to assign to the TRACE32 device.
6. Do one of the following:
   - If you want to use an Ethernet hub or switch, select the **DHCP (via device name)** checkbox.
   - If you to use a cross-over cable, clear the **DHCP (via device name)** checkbox, and enter the IP address you want to use for this TRACE32 device in the **ip address** box.
7. Select the **full duplex** checkbox only if auto-negotiations troubles occur with the used ethernet switch or hub.



8. Click **Save to device**.
   This saves the device name in the internal memory of the TRACE32 device.
9. Click **Close**.
10. Choose **File** menu > **exit** to close the TRACE32 PowerView GUI.

Most sites will use Ethernet with DHCP and DNS to assign IP addresses and host names to devices. Contact your system administrator for a host name, and then proceed as follows:

1. In the **windows64** or **pc_linux64** folder, open the `config.t32` file with an ASCII editor.
2. Modify the `config.t32` file:
   - For Windows, refer to this this **example**.
   - For Linux, refer to this **example**.

NOTE:	Do NOT use whitespaces around the operator **=** in the configuration files.

3. For `NODE=` enter the host name you have received from your system administrator.
4. Save your `config.t32` file and close it.

Example configuration for Windows:

```
// TRACE32 configuration file for NET
// (Windows)
OS=
ID=T32TEST002
TMP=C:\temp                     ; temporary directory for TRACE32
SYS=C:\T32MINI                  ; system directory for TRACE32
HELP=N:\TRACE32DVD\files\pdf    ; help directory for TRACE32

PBI=
NET             ; i.e. EtherNET
NODE=pod-hen01  ; host name assigned to the TRACE32 device

PRINTER=WINDOWS
```

Example configuration for Linux:

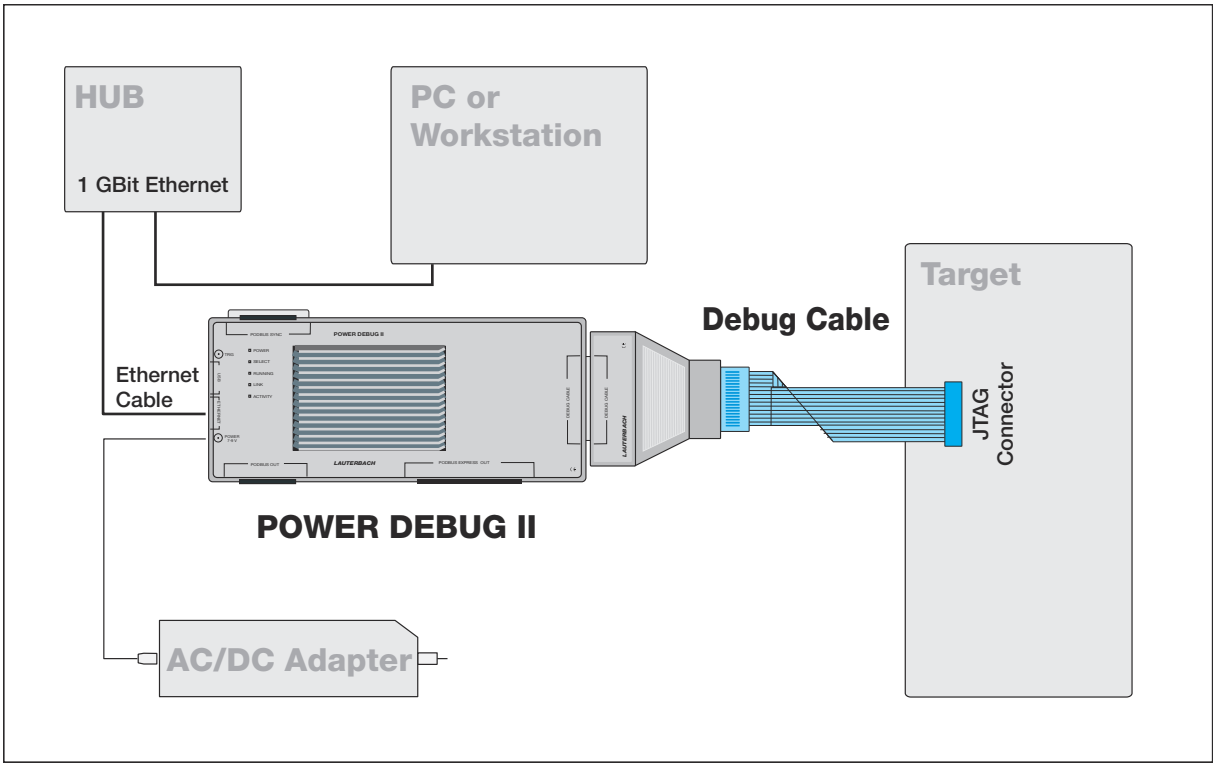
```
// TRACE32 configuration file for NET
// (Linux/MacOSX/Solaris)
OS=
ID=T32TEST002
TMP=/var/tmp
SYS=/opt/t32mini
HELP=/media/TRACE32/files/pdf

PBI=
NET             ; i.e. EtherNET
NODE=pod-hen01  ; host name assigned to the TRACE32 device
```

Remember that we started with a USB configuration, and then modified the TRACE32 configuration file for Ethernet. Now we are ready to re-configure the TRACE32 hardware for Ethernet as well.

A direct cross-over cable link between TRACE32 device and PC or Unix workstation can only be used if you have assigned an IP address to the TRACE32 device as described in **"Assign a Host Name to the TRACE32 Device (Ethernet)"**, page 18. Otherwise you need a hub (or switch) and patch cable(s).

To re-configure the hardware for Ethernet:

1. Unplug the USB cable.
2. Do one of the following:
   - Plug an Ethernet cable into the POWER TOOL / ETHERNET and the other end into a free port on an Ethernet hub or switch.
   - If you want a direct connection with your Windows / Linux PC (or Workstation), plug the Ethernet cross-over cable directly into the POWER TOOL and your PC.



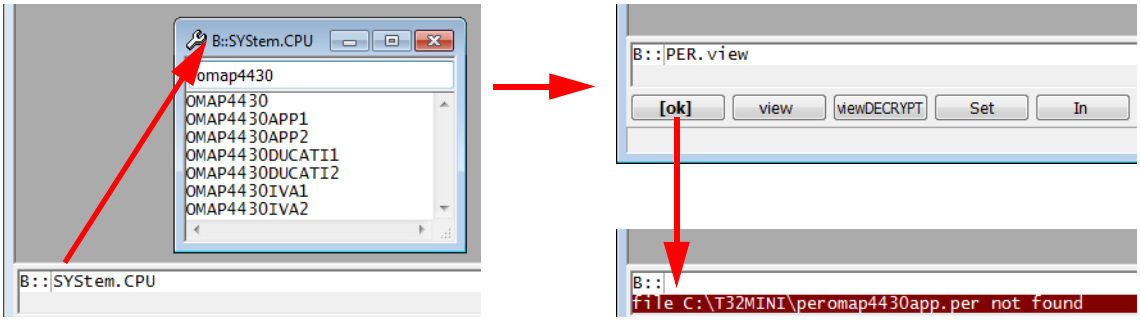
NOTE: Now continue with **"Identify the Peripheral File"**, page 21.

This step-by-step procedure describes how to identify the missing peripheral file (*.per) for a *minimal manual setup*. A full installation setup copies all available *.per files to the system directory, by default `C:\t32`.

To identify the missing *.per file in a minimal manual setup:

1. If they are not powered-on already, switch on the TRACE32 hardware, and then power on your target board.
2. Start TRACE32 by double-clicking the TRACE32 executable in the **windows64** or **pc_linux64** folder.
3. At the TRACE32 command line, type these commands:

```
SYStem.CPU ;Opens the SYStem.CPU window, displaying a list of CPUs.

;Double-click the CPU you want to use, e.g. OMAP4430.

PER.view   ;Opens the peripheral file for the selected CPU.

           ;If the file is not found, its name is displayed
           ;in the message bar, see screenshots below.

AREA.view  ;Allows you to copy the file name of the missing file.
```



Remember that we have not copied any peripheral file (*.per) yet. As a result, TRACE32 displays an error message.

4. From the TRACE32 installation DVD, copy the *.per file to the system directory; in our example, copy **peromap4430app.per** to the **T32MINI** folder.

File	DVD Location	Description
per*.per	...\files	Definitions for on-chip peripherals (text files)

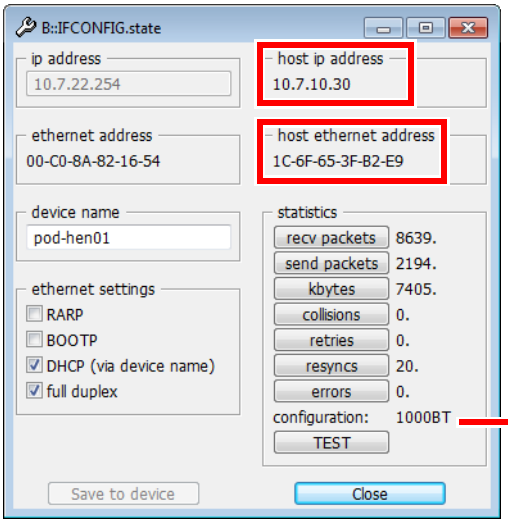
You do **NOT** need to re-start TRACE32.

5. To display the PER file contents, type these commands:

```
SYStem.Up
PER.view  ;Displays the peripheral file for the selected CPU,
          ;in our example, for OMAP4430.
```

This completes the minimal manual setup of TRACE32. For the curious reader and user:

The screenshot below is an example of the **IFCONFIG.state** window after we have changed the USB configuration into an ETHERNET configuration, as described in this chapter about the minimal manual setup:



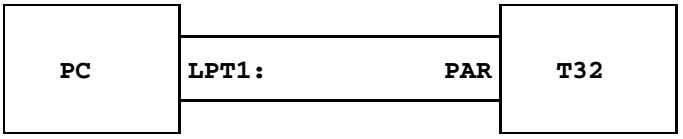
Gigabit Ethernet - BASE T

Legacy Host Interfaces

These interfaces have become obsolete or are no longer for sale.

Parallel Interface (TRACE32-PAR)

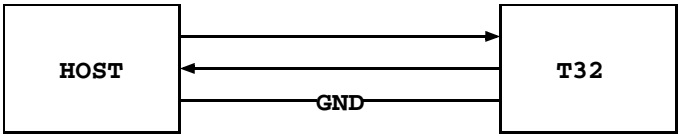
The PODBUS PARALLEL INTERFACE (host-based) was the former standard host interface for PC. With the widespread availability of faster peripheral connection methods like USB and Ethernet it has become obsolete.



Superseded by Ethernet and USB.

Serial Interface (TRACE32-SER)

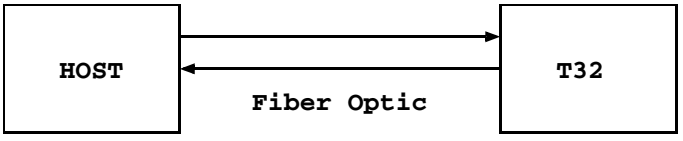
This interface is mainly used on non-standard hosts. The physical link may be RS232 or RS422. The maximum speed is 1 MBit/s with asynchronous transfer modes.



No longer available.

Fiber Optic (TRACE32-SER, TRACE32-NET)

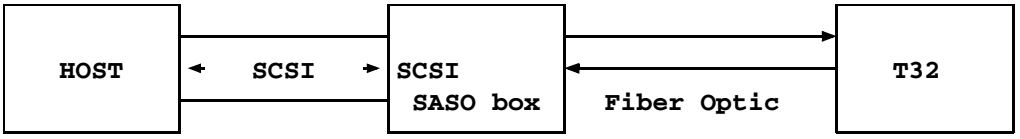
This interface is used on PCs only. A special interface card is needed (AT or MC bus). The max. speed is 2 MBit/s (ETHERNET card) or 1 MBit/s (SER).



No longer available.

SCSI Interface (TRACE32-SER, TRACE32-NET)

This interface is made by a special interface box, which connects the SCSI bus to the fiber optic interface. The interface supports all workstations (UNIX or VMS).



No longer available.

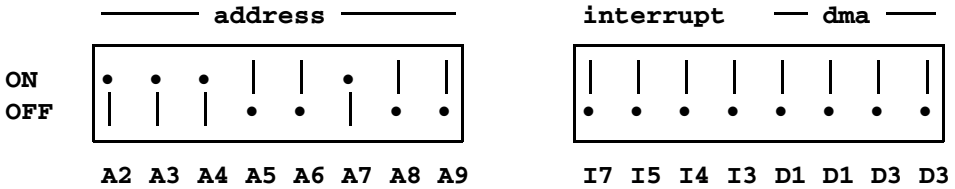
Host Interface Cards

Both interface cards are no longer available.

Fiber Optic Interface (PC-ISA)

This interface card occupies a short slot with an 8 bit bus connector. The address selector is set to 360H, interrupts and DMA are not used in the standard mode. No default settings must be changed on software installation. If a DMA based driver is used, the DMA switches will be set to ON (for faster download).

Default switch settings on the interface card:



switch open: 1
switch closed: 0

switch open: IRQ/DMA not used
switch closed: IRQ/DMA used

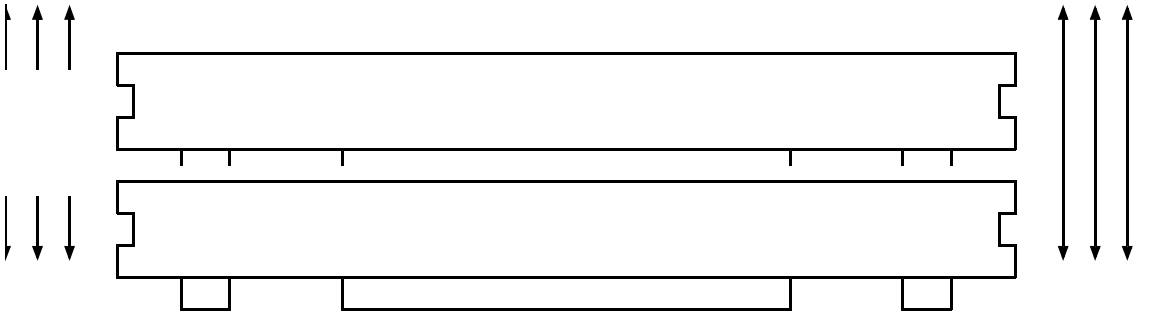
This card contains no DIP switches or selectors. The interface address is configured by the adapter configuration file. The address of the port must also be selected there.

To install the adapter in your Micro Channel computer, you have to complete the following steps:

1. Copy the file '@50DF.ADF' to the reference disk (you got this disk with your MicroChannel-Computer).
2. Turn off your computer and install the adapter in any available slot.
3. Insert the reference disk in floppy drive A:.
4. Turn on the system.
5. After booting the setup-program will ask you whether to do an auto- configuration or not. Answer this question with 'No'.
6. Choose the 'configuration' option in the main menu of the setup program.
7. Choose the 'auto configuration' option in the configuration menu.
8. If the auto-configuration fails, you must change the 'ADF' file. Call technical support for details.
9. If the configuration was successful, choose the option 'show configuration' and scroll through the slot-list until you see the entry for the TRACE32 Fiber Optic Interface and its Base I/O-Address. Record this base address.
10. If the base address is not 360H, change the line 'ADDRESS=' in the configuration file 'config.t32' to the correct address. NOTE: The address must be entered in decimal.

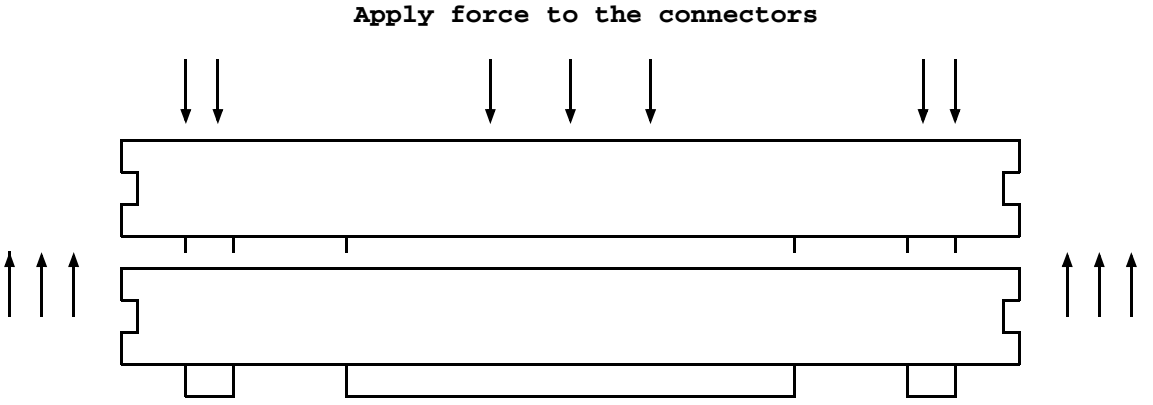
Remove Modules

Switch power off before opening the system. Remove the sliders between the module and other attached modules. Begin to apply force from the back end of the system, near the hole for the fan, and disconnect the module slowly.



Add Modules

Switch power off before opening the system. Check the connectors of the module to ensure proper alignment of the pins. Fit the connectors between the new module and the system. Be sure not to bend a pin when connecting the modules. Apply force to the connectors of the new module, beginning from the front of the system. If the module is snapped in, once apply force to all connectors to ensure proper contact. As the last step attach the cover plate and the sliders to the module.



System Memory (SCU/PODETH)

The system controller (SCU) or the PODBUS Ethernet interface (PODETH) consists of main processor, memory and the host interface. The main software and the user interface is running on this processor. Symbol information is also held in the SCU/PODETH memory. If the error message 'out of memory' is displayed during the download of a large program, it may be necessary to upgrade the memory.

- The max. size of memory is 8 MBytes for the SCU16 and 16 MBytes for the SCU32 and up to 64 MByte for the current SCU type.
- The max. size of memory for the PODETH interface is 128 MByte.

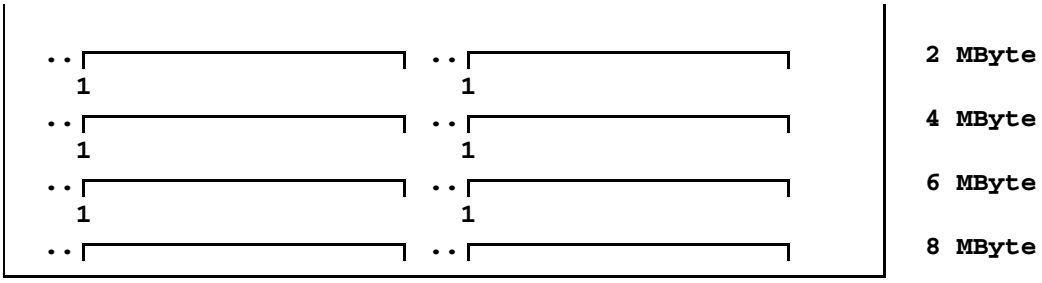
The currently used memory size can be displayed with the command **SETUP.MEMory**. As a worst case calculation for the required memory size for an HLL debug object file the following formula can be used:

MEMORY_SIZE = (FILE_SIZE - BINARY_CODE_SIZE) * 3

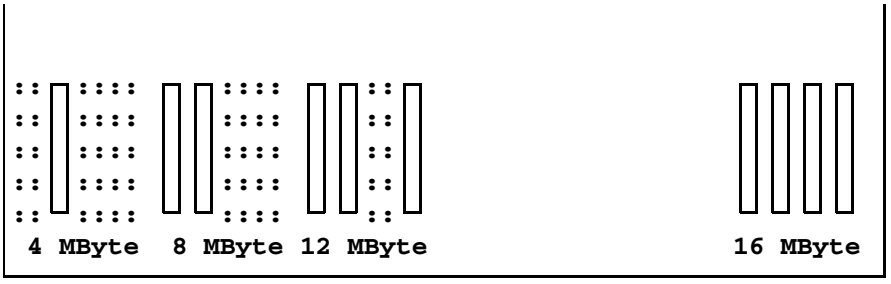
For a memory upgrade the memory module in the SCU/PODETH must be exchanged.

For SCU16, SCU-PAR, SCU32/15 and SCU32/22 memory extensions can be inserted in the SCU module after removing the cover plate with the fan.

SCU16-MX2



SCU32-MX4



Standard 1MBit*4 (60/80ns) ZIP memory chips can be used to upgrade the memory. Eight memories must be used for a 4 MByte extension. Memory must be plugged in according to the schematics above, e.g. for 12 MByte the three memories in position 4, 8 and 12 must be plugged in.

PowerTools

Within the PowerTools family we distinguish three types of hardware modules:

1. **Hardware modules with host interface which must be connected to the host (PODBUS SYNCH):**

   POWER DEBUG INTERFACE / USB 3

   POWER DEBUG PRO

   POWER DEBUG II

2. **Hardware modules with host interface which may be connected to the host (PODBUS IN):**

   POWER DEBUG INTERFACE / USB 2

   POWER DEBUG / ETHERNET

   POWER TRACE / ETHERNET

3. **Hardware modules without host interface**

   POWER TRACE PX

   POWER TRACE II

   POWER PROBE / LOGIC ANALYZER

   POWER INTEGRATOR

   POWER INTEGRATOR II

4. **Stand-alone hardware module with host interface**

   µTrace

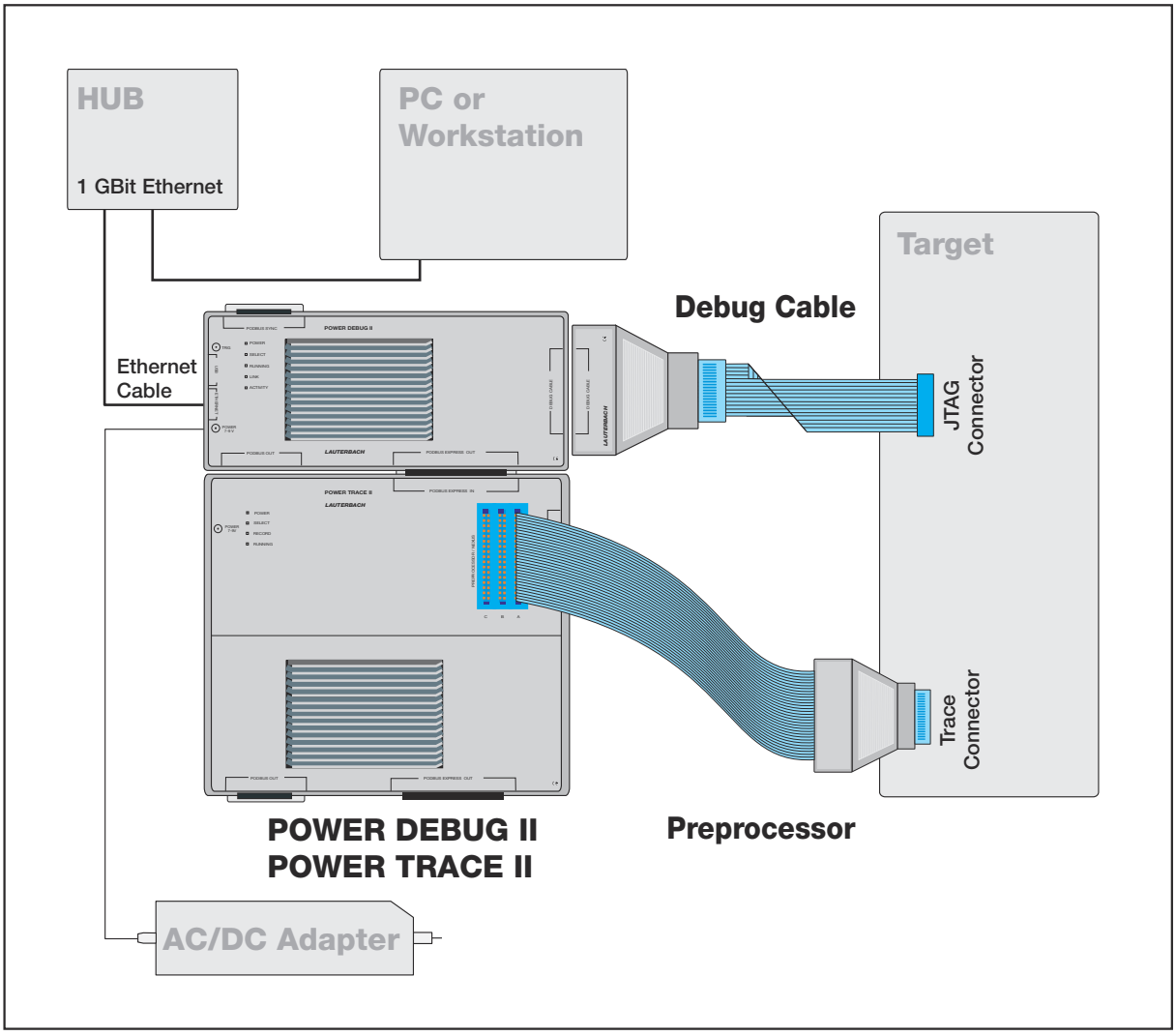
POWER LED

If the POWER LED is on, this indicates that a power supply is connected. This applies to all hardware modules except POWER TRACE II and POWER INTEGRATOR II.

POWER TRACE II/POWER INTEGRATOR II

POWER TRACE II/POWER INTEGRATOR II are only partially powered when a power supply is connected but the TRACE32 software is not started. This avoids unnecessary noise. The POWER LED is flashing to indicate this partly powered state.

POWER TRACE II/POWER INTEGRATOR II are completely powered when the TRACE32 software is started.



In the standard configuration (one POWER TRACE II/POWER INTEGRATOR II is assembled with a POWER DEBUG II module) the power supply connected to POWER DEBUG II is sufficient to supply both units. Any further module needs an extra power supply (more the 7 A).

A POWER LED that is still flashing after the TRACE32 software is started indicates that the current power supply is not sufficient. POWER TRACE II/POWER INTEGRATOR II provide their own POWER 7-9 V connector. This connector can be used to connect an additional power supply.

TRACE32 Software not Started

The following applies only to the hardware module that is connected to the host:

The SELECT LED flashes regularly when the self-test of the hardware module was successfully completed.

If the self test failed, the following error codes are flashed:

- Three flashes, then a pause: memory test failed
- Six flashes, then a pause: buffer overflow
- Nine flashes, then a pause: failed to load firmware
- Twelve flashes, then a pause: checksum error.

In any case, please contact your local Lauterbach support.

NOTE:	The uTrace does not have a SELECT LED, because it is a stand-alone hardware module. The RECORD LED is used here to indicate that a self-test failed.

TRACE32 Software Started

The SELECT LED is on when the TRACE32 software in communication with the hardware module.

NOTE:	The state of this LED has no meaning as long as the TRACE32 software is not started.

The following applies to all hardware modules except POWER TRACE II and POWER INTEGRATOR II:

- If a single-core processor is debugged, the RUNNING/EMULATE LED is ON, when the program execution is running. The RUNNING/EMULATE LED corresponds with running indicated in the Debug field of the **TRACE32 PowerView state line**.
- If a multi-core chip is debugged, the RUNNING/EMULATE LED is ON, when the program execution is running on the core that is the master of the debug communication (**SYStem.CONFIG Slave OFF**).

The following applied for POWER TRACE II:

- If the on-chip trace generation logic generates Nexus 5001™ compliant trace messages the RUNNING is ON, when the program execution is running. This is realizable because the trace generation logic generates Debug Status Messages at the start and the stop of the program execution.
- For all other trace protocols the RUNNING LED is on, if the POWER TRACE II hardware is ready to record trace information (see also RECORD LED description) and valid trace information is received.

The following applied for POWER INTEGRATOR II:

- For all other trace protocols the RUNNING LED is on, if the POWER TRACE II hardware is ready to record trace information (see also RECORD LED description) and valid trace information is received.

RECORD/RECORDING/TRACE LED

NOTE:	The state of this LED has no meaning as long as the TRACE32 software is not started.

The RECORD LED is on when the logic analyzer/ trace recording is armed. Armed means incoming trace data are recorded. The RECORD LED corresponds with ARM indicated in the Trace field of the **TRACE32 PowerView state line**.

NOTE:	The state of this LED has no meaning as long as the TRACE32 software is not started.

The TRIGGER LED is on when the logic analyzer encountered a trigger event. The TRIGGER LED corresponds with TRIGger indicated in the Trace field of the **TRACE32 PowerView state line**.

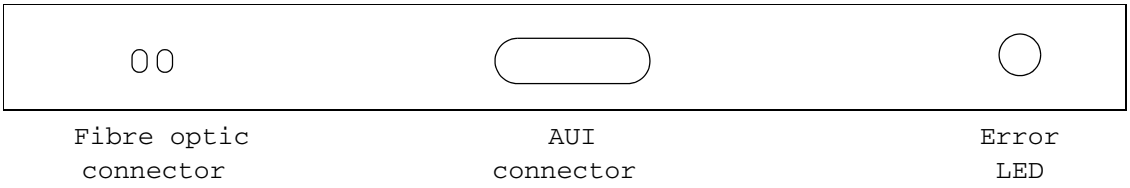
ERROR LED

The ERROR LED on the POWER INTEGRATOR has no function.

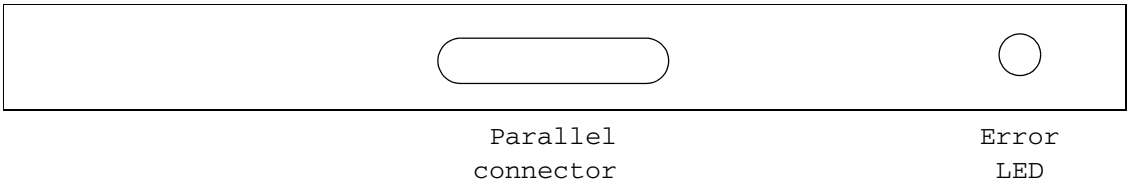
LEDs on POWER DEBUG II	
LINK	Physical link established.
ACTIVITY	Ethernet packets received respectively sent.

LEDs on POWER DEBUG / ETHERNET and POWER TRACE / ETHERNET	
CONNECT ERROR	Ethernet connection error occurred.
TRANSMIT	Ethernet packets sent.
RECEIVE	Ethernet packets received.
COLLISION	Ethernet collision occurred.

Rear panel (Fibre optic connector/AUI connector):

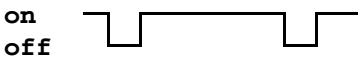


Rear panel (Parallel interface):

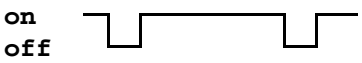


When the TRACE32 system is starting up, the Error LED on the rear of the System Control Unit (SCU) lights for 5 … 40 s, depending on the memory size. After this delay time the error codes shown below may be blinked:

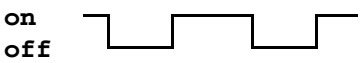
No carrier (fiber optic interface only)



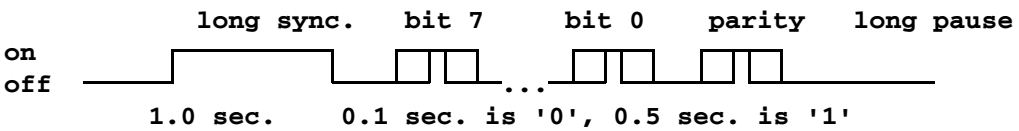
Waiting for connection (ETHERNET only)



Time-out of peripheral systems

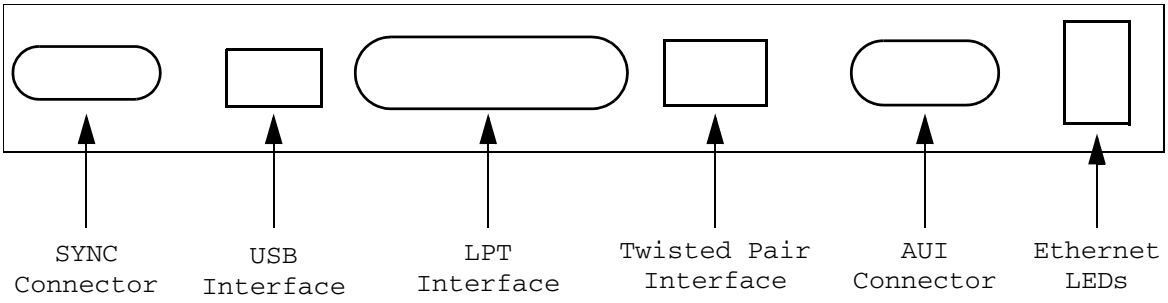


Fatal error



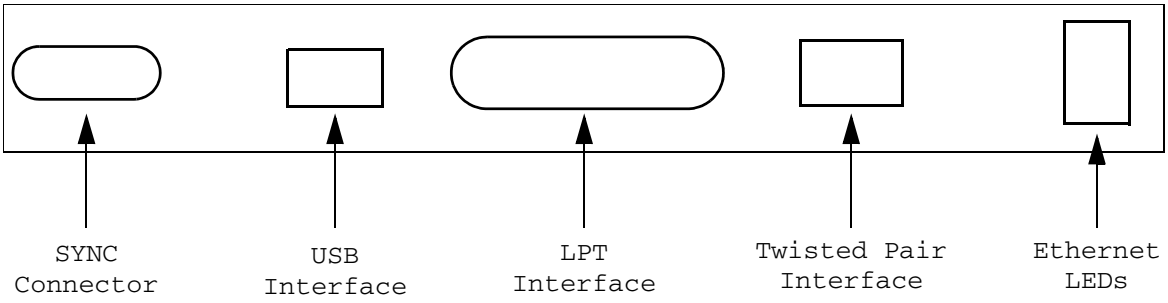
Selftest error with error code

Rear panel:



Ethernet LEDs	
T	Transmit activity.
R	Receive activity.
C	Collision activity.
L	Twisted pair link integrity.
S	Twisted Pair Polarity Error (receiver inputs TPRX+, TPRX- are reversed).
X	Twisted Pair Jabber condition detected.
E	In error case an error code is pulsed.
A	On if device is active, flashes if device is not used.

Rear panel:



Ethernet LEDs	
T	Transmit activity.
R	Receive activity.
C	Collision activity.
L	Twisted pair link integrity.
S	Speed, on if 100MBit Ethernet is used, off if 10MBit Ethernet is used.
F	Full duplex.
E	In error case an error code is pulsed.
A	On if device is active, flashes if device is not used.

LEDs	
POWER	External power is supplied.
ACTIVE	On if device is active, flashes if device is not used.
ERROR	In error case an error code is pulsed.
TRANSMIT	Transmit activity.
RECEIVE	Receive activity.
COLLISION	Collision activity.
LINK	Twisted pair link integrity.
POLARITY	Twisted Pair Polarity Error (receiver inputs TPRX+, TPRX- are reversed).
JABBER	Twisted Pair Jabber condition detected.

LEDs	
POWER	External power is supplied.
ACTIVE	On if device is active, flashes if device is not used.
ERROR	In error case an error code is pulsed.
TRANSMIT	Transmit activity.
RECEIVE	Receive activity.
COLLISION	Collision activity.
LINK	Twisted pair link integrity.
100 MBPS	On if 100 MBit Ethernet is used, off if 10MBit Ethernet is used.
AUX	Full duplex.

TRACE32-USB

Connector (USB 3.x)

Pin	Signal Name	Color
1	VCC	red
2	D-	white
3	D+	green
4	GND	black
5	StdB_SSTX-	blue
6	StdB_SSTX+	yellow
7	GND_DRAIN	shield
8	StdB_SSRX-	purple
9	StdB_SSRX+	orange

Connector (USB 2.0 and 1.x)

Pin	Signal Name	Color
1	VCC	red
2	D-	white
3	D+	green
4	GND	black

USB Interfaces (USB 3.x to 1.x)

TRACE32 Power Tools can be connected with standard USB cables to any free USB port on the PC itself or on a connected USB hub (stand-alone or embedded in a peripheral, e.g. a monitor or keyboard).

From the tables above you see that a USB connector can also provide power to a device. But the 2.5 Watts provided with USB 2.0, and the 4.5 Watts specified for USB 3.0 are not enough for the sophisticated hi-speed debugging hardware used in our TRACE32 tools. Therefore TRACE32 devices are “self-powered”, i.e. they get their power from an external power supply (that usually comes with the device).

USB 3.x

POWER DEBUG INTERFACE / USB3, POWER DEBUG PRO, and µTrace are Lauterbach TRACE32 Power Tools that support USB 3.0 SuperSpeed (max. bus transfer rate of 5 GBit/s). To take advantage of this high speed, please make sure you operate these TRACE32 devices on USB3 ports, and with matching USB3 cables.

The net transfer speed that can be obtained under real working conditions depends on a number of factors (including PC CPU and USB port chip, cable quality, but also the speed of the target JTAG interface or TRACE port) and has been observed to be roughly between 40 MBytes/s and 100 MBytes/s.

If you don't have any free USB 3.x port, you can also use an USB 2.0 port and USB 2.0 cable. Some loss of transfer speed is expected, but because of the improved internal data path layout introduced for USB 3.x, the devices still have higher transfer rates via USB 2.0 (between 20 and 35 MBytes/s) than our "pure USB 2.0" devices.

The USB 3.x standard does not specify any maximum cable length, but for SuperSpeed connections we recommend using 3 metres or less of high-quality cable, and to plug it directly into a PC USB3 port.

For comparison, a microwave oven uses 2.45 GHz frequencies, USB 3.0 uses 5.0 GHz. This means that interference from other devices, within your PC and from external devices, might affect USB transfers.

USB 2.0

POWER DEBUG INTERFACE / USB2, POWER DEBUG II and POWER DEBUG / ETHERNET all implement a combined USB 2.0 and USB 1.x interface. Connected to a USB 2.0 port, they run in Hi-Speed mode (480 MBit/s) with observed transfer rates of up to 12 MBytes/s.

Please note that the USB specification limits USB 2.0 cable length to 5 metres. This already includes any device-internal wiring, however, so e.g. a laptop plugged into a docking station might cause trouble when you use a 5 metre long USB cable. Also please note that 480 MBit/s means that the USB 2.0 bus has to handle 480 MHz frequencies. At these speeds, electromagnetic interference already becomes an issue.

USB 1.x

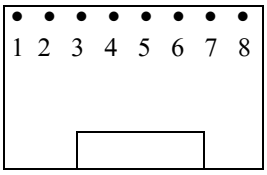
The POWER DEBUG INTERFACE / USB (as one example of this device category) is a full-speed device which uses the full USB 1.x bandwidth of 12 Mbps. USB 2.0 devices connected to a USB1.x port, will have transfer rates of up to 1.5 MBytes/s (some data can be additionally compressed, so in some cases you might observe slightly higher transfer rates).

Low-speed USB 1.x with 1.5 MHz transfer rate is not supported by TRACE32 devices.

Standard connector for the ethernet connection on TRACE32 development tools is a twisted pair connector.

8P8C-Connector (T568A/B, 'RJ45')

This is the standard connector for Ethernet interfaces, for use with twisted-pair cabling.

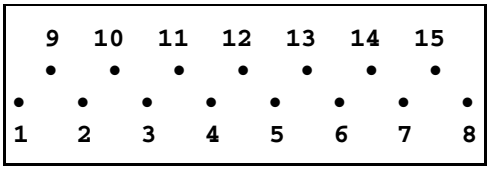


	1000BT	100BT
1	TRP1+	TRP1+
2	TRP1-	TRP1-
3	TRP2+	TRP2+
4	TRP3+	(termA)
5	TRP3-	(termA)
6	TRP2-	TRP2-
7	TRP4+	(termB)
8	TRP4-	(termB)

Termination (term):
75 Ohms to GND

AUI-Connector

This is a legacy interface that is not available on newer TRACE32 devices.



1	GND		
2	COL+	9	COL-
3	OUT+	10	OUT-
4	GND	11	GND
5	IN+	12	IN-
6	GND	13	+12V (0.25 A max)
7	N/C	14	GND
8	N/C	15	GND

Ethernet Interface

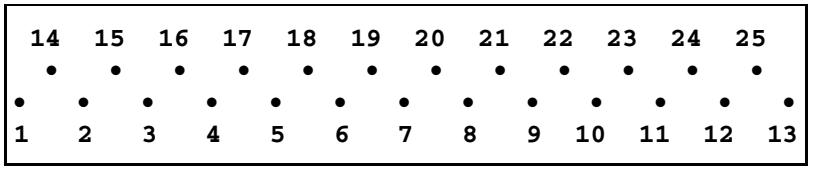
The MAC address of the Ethernet interface is recorded on the bottom of the system. The physical transfer speed is 10 MBit/s (legacy devices), 100 MBit/s or 1000 MBit/s. In idle mode, ETHERNET traffic is reduced to a minimum of 2 packets/s.
The command **SETUP.URATE** can limit the update speed of the window system, and thus reduce the Ethernet traffic, too. The command **IFCONFIG.PROfile** shows statistics about Ethernet usage.

Transfer protocol and interface are selected automatically. The interface, which first establishes a connection is selected for operation. The baud rate and protocol of the fiber optic interface is selected automatically, too. The fiber optic cable can be left connected if the Ethernet interface will be used.

TRACE32-PAR

This is a legacy interface that is not available on newer TRACE32 devices.

Connector



1	-STR	14	-AF
2	D0	15	-ERROR
3	D1	16	-INIT
4	D2	17	-SELIN
5	D3	18	GND
6	D4	19	GND
7	D5	20	GND
8	D6	21	GND
9	D7	22	GND
10	-ACK	23	GND
11	BUSY	24	GND
12	PE	25	GND
13	SELECT		

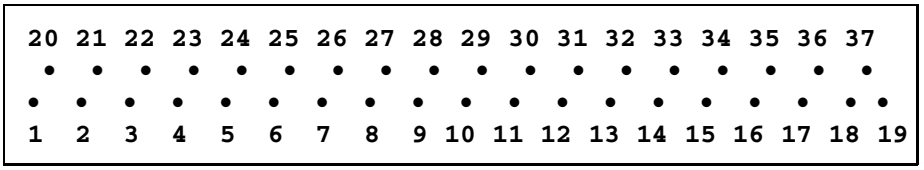
Parallel Interface

The interface cable is connected directly to the printer port of the PC. TRACE32 supports EPP and ECP mode. The effective download speed is approximately 350 KByte/s.

TRACE32-SER

This is a legacy interface that is not available on newer TRACE32 devices.

Connector



1	GND	20	CODE-0
2	TXD RS232-0	21	CODE-1
3	RXD RS232-0	22	CODE-2
4	RTS RS232-0	23	CODE-3
5	CTS RS232-0	24	CODE-4
6	TXD RS232-1	25	CODE-5
7	RXD RS232-1	26	CODE-6
8	RTS RS232-1	27	CODE-7
9	CTS RS232-1	28	CODE-8
10	+10V	29	CODE-9
11	-10V	30	CODE-10
12	GND	31	CODE-11
13	+5V	32	GND
14	GND	33	CLKIN+
15	RXD+ RS422	34	CLKIN-
16	RXD- RS422	35	CLKOUT+
17	TXD+ RS422	36	CLKOUT-
18	TXD- RS422	37	GND
19	GND		

RS232 Interfaces

Two serial interfaces are implemented. Both interfaces are no longer available.

RS422-Interface

This interface is very flexible. The interface consists of clock input, clock output, data input and data output lines. The transfer protocol is asynchronous. The clock rate may be up to 1 MHz. Clock input is the same for receiver and transmitter. Clock signal is either generated by TRACE32 or the host computer.

Fiber Optic Interface

This interface can be used for all PC-compatible host computers. For installation no change on the interface setup must be made. The physical transfer speed is 1MBit/s.

Selection of Transfer Protocol

The transfer protocol is selected by coding of the interface cable connector. No change on TRACE32 system must be made by changing between different host computers.

The transfer protocol is defined by the input lines CODE-0 to CODE-11. A logical 0 is defined by shortening the input pin to ground, logical 1 is set with an open input line.

If no interface cable is used, the fiber optic link will be selected by default.

Selection of Interface

CODE-11-10-9-8	
1111	Fiber Optic asynch. with handshake
1110	Fiber Optic asynch. DMA (boot EPROM version 2.3 or less)
1101	reserved
1100	RS232 no handshake
1011	RS232 software handshake
1010	RS422 no handshake
1001	RS422 software handshake
1000	reserved
….	
0000	reserved

CODE-2-1-0	baud rate
111	9600
110	19200
101	38400
100	76800
011	153600
010	307200
001	500000 (RS422 only, boot EPROM 3.x)
000	1MBit (RS422 only, boot EPROM 3.x)

CODE-4-5	(used on software handshake only)
00	mode 0
01	…
10	…
11	mode 3

CODE-4	(no handshake)
0	mode 0
1	mode 1

CODE-6	
0	block checksum, 8 bit data, no parity, 1 stop bit
1	parity, 8 bit data, even parity, 1 stop bit

All other pins must be left open. For details in selecting a protocol refer to the software part of the installation manual.

SASO

This is a legacy interface that is not available on newer TRACE32 devices.

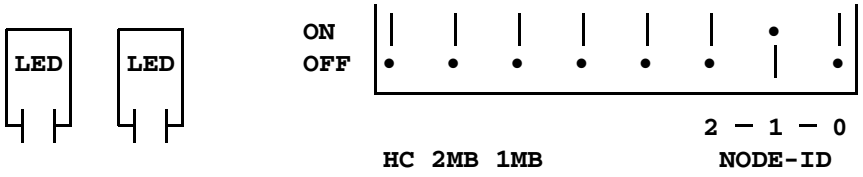
Connector

The SASO box has two 50-pin female high density connectors, following the SCSI-II standard. Cables to connect to other standards like SCSI-I, D-SUB or DEC are available.

Fiber Optic Interface

The transfer speed is 1, 2 or 4 MBit/s. The direct DMA protocol is used. SCUs with boot EPROM version 2.3 or less must select this protocol by shorting pin 28 and 32 on the 37-pin connector. SCUs with boot EPROM 3.x choose this protocol automatically. The interface system selects the highest transfer rate possible.

Selection of Interface



HC	High Current Fiber Optic (for 2/4 MBit or long distance)
2MB	Limit transfer speed to max. 2 MBit/s
1MB	Limit transfer speed to 1 MBit/s
NODE-ID	SCSI address (default: 2)

Files and Directories

The TRACE32 system needs two fixed directories. On the system directory all programs and data tables are stored. The temporary directory is used for temporary data tables. On multi-user systems with more than one TRACE32 device, all system files may be shared. Temporary directories must be kept separate for each user. Either different temp directories can be used, or the first characters of temporary file names have to be different for each user. These first characters are defined by the user ID code.

Example of the separation of paths:

Systemtype	ID code	System dir.	Temporary directory (example)
single user	T32	C:\T32*.*	C:\TMP\T32?????
multi user option 1	T32	C:\T32*.*	C:*userid*\TMP\T32?????
multi user option 2	userid	C:\T32*.*	C:\TMP*userid*?????

The paths name definition for system and temp directories can be done via environment variables (of the host operating system) or in the TRACE32 configuration file. The configuration of these path names in the config.t32 file is only useful when environment variables are hard to define.

Environment variables used by TRACE32:

Environment	config.t32 OS section	usage
T32ID	ID=	ID code for temporary files (prefix)
T32SYS	SYS=	Directory for system files
T32TMP	TMP=	Directory for temporary files
T32HELP	HELP=	Directory for the PDF help files
T32CONFIG		Pathname of configuration file (used instead of config.t32)
ACROBAT_PATH		Unix only: Installation directory of the Acrobat Reader (acroread)

If defined in the configuration file, the definitions must be made in the "OS=" section:

```
OS=
ID=ste
SYS=c:\t32
TMP=c:\tmp
HELP=c:\t32\pdf


PBI=
USB
;…
```

t32*(.exe)	TRACE32/PowerView GUI executable, e.g. t32mppc.exe, t32marm.exe, …
config.t32	Default name of configuration file
license.t32	Default name of software license file (only present if required)
boot.t32	boot loader and linker (downloaded to TRACE32 device by the TRACE32 PowerView GUI)
boot*.t32	boot files generated by the loader/linker
mcc*.t32	TRACE32 device software for SCU with NSC-CPU
mcp*.t32	TRACE32 device software (SCUPPC, PODBUS ETHERNET CONTROLLER)
fcc*.t32	TRACE32 device software for POWER TOOLS and FIRE
scc*.t32	TRACE32 device software for ICE
per*.per	definition file for on-chip peripherals (text files)
help.t32 *.pdf trace32.api	Help index, full text search database, tooltip text and error/warning messages Online manual files (in the subdirectory named as HELP= in the config file) TRACE32 interface file to Acrobat Reader API
men*.men	additional CPU- or task-specific menu files
t32.men	top-level English menu file
t32jp.men	top-level Japanese menu file
t32jp-utf-8.men	top-level Japanese menu file in UTF-8 format
autostart.cmm	primary start-up script provided by Lauterbach (do not edit)
system-settings.cmm	secondary start-up script for general/default settings (file can be created by the user)
system-preferences.cfg	optional global settings file
t32pro.ps	postscript header file (page and formatting pre-amble for postscript output)
t32font.fon	TRACE32 font file (WINDOWS only)
xplib.t32	library for FLASH programmer

The TRACE32/PowerView GUI executable binary should be installed in a directory which is in the environment search path, and the PDF files should be placed in the directory defined with HELP=.
All other files **must** exist in the TRACE32 System directory (T32SYS environment variable or SYS= option).

For Host-based configurations, the TRACE32 operating software is started by entering the name of the TRACE32 binary for your chip family, e.g. "t32marm".

For SCU-based configurations, you start “t32win.exe” (Windows) or “t32cde” (Unix). When the program is executed the first time after a hardware or software update, an automatic configuration process will be started. This process includes analyzing the hardware configuration and linking a bootable image file from the corresponding 'mcc' files. The linked image is saved in a file named 'bootxx'. For each hardware

configuration, one boot file is generated, so that different TRACE32 systems can share one system directory. When the t32win or t32cde program is started for the first time, it must have write-permission for the System directory. Once the bootxx.t32 file is generated, this write permission can be revoked.

If the TRACE32/PowerView GUI fails to save the bootable file in the System directory, it will attempt to use the temporary directory (defined by T32TMP) instead. This allows using the system directory in 'read-only' mode. On **UNIX** systems the keyword **NOLOCK** must be used after the header **BOOT=** to turn off the file locking for the 'boot.t32' file. Otherwise the 'boot.t32' file is accessed in read/write mode (for file locking).

```
LINK=NET
NODE=t32
PACKLEN=1024


BOOT=
NOLOCK
```

In UNIX environments, a third configuration option for the boot files is available. A dedicated user account can be defined, who owns the TRACE32 System directory and the TRACE32 device boot files. All other users only get read permissions to this directory. The TRACE32/PowerView GUI changes the user ID before accessing the boot files. This option is activated with **USER=**username in the configuration file.

Another alternative is to place the boot files in the temporary directory. This can be done by adding the keyword **TMPDIR** after the header **BOOT=** in the configuration file. The advantage over placing the boot files on the server is faster booting, and less network traffic during booting.

When all necessary boot files have been generated, all 'mcc' files can be deleted (if the disk space is needed for other applications).

If multiple TRACE32 devices are connected to one host, they can share the same system directory. To distinguish between the devices, the environment variable 'T32CONFIG' should be set to different configuration files. The configuration files should change the T32ID with the keyword 'ID=' (after OS=).

The following example of a legacy configuration connects two in-circuit emulators to a SUN workstation via Ethernet. An alternate method is to use one configuration file including different environment variables or to build an emulator pool. In an emulator pool the driver program connects to the first available emulator of a pool of emulators.

Two configuration files:

```
OS=                                    OS=
ID=_1                                  ID=_2


LINK=NET                               LINK=NET
NODE=t32_1                             NODE=t32_2
```

Batch file to connect to emulator #1:

```
T32CONFIG=/usr/t32/config_1.t32
export T32CONFIG
/opt/t32/bin/suns/t32cde
```

Environment variable in configuration file:

```
LINK=NET
NODE=${T32NODE}

SCREEN=
HEADER=ICE ${T32NODE}
```

Command Line Argument in configuration file:

```
LINK=NET
NODE=${1}

SCREEN=
HEADER=ICE ${1}
```

Emulator Pool:

```
LINK=NET
POOL=t32_1,t32_2,t32_3
```

File config.t32

The configuration file for the system defines the drivers to be installed in the host. By default the software running on the host, assumes that the configuration file is in the system directory and named 'config.t32'. But this can be overruled by specifying the configuration file in the environment variable T32CONFIG. On some systems (e.g. Windows) the configuration file can be defined by a command line option (-c).

Format:	*<driver>*=[*<file>*|*<name>*|**REMOTE**|**OFF**] *<special_commands>* **...** **empty line**
<driver>:	**OS** **PBI** **LINK** **SCREEN** **PRINTER** **LICENSE** **BOOT** **ALINK** (PC only) **ASCREEN** (PC only) **FILE** **KEY** **MOUSE** **SOUND** **REMOTE** **ASSIST** (Ethernet only)

The definitions in this file are used to select, to load and to configure all TRACE32 ‘drivers’. All commands and names must be used as shown above (especially blanks and upper/lower case characters).

All drivers can be defined by a line containing the driver definition and some optional commands related to the defined driver. Every configuration block has a headline defining the type of the driver, an equal sign and optionally the internal name or file name of the driver. If there is no file name given, no driver is loaded and all subsequent definitions are passed to the default driver. Each driver block must end with an **empty line**.

Comments in the configuration file must begin with a semicolon in the first column.

```
; Minimal USB configuration
OS=
ID=T32TEST1
SYS=/opt/t32
TMP=/opt/t32/tmp
HELP=/opt/t32/help

PBI=
USB
NODE=pod-hen3

;eof
```

Environment variables and command line parameters can also be used:

```
; Use node name from environment variable, title from command line
PBI=
NET
NODE=${T32NODE}

SCREEN=
HEADER=${1}

;eof
```

This will use the environment variable 'T32NODE' as the definition of the Ethernet node, and the first command line argument will be used as the window header. Comments must start with a semicolon in the first column.

You can also generate ‘hierarchical’ configuration files by using include files:

```
$INCLUDE /usr/t32/config.all

SCREEN=
PALETTE 0 = 12 34 56

;…
```

This allows to keep some configuration information local (e.g. in the home directory of the user by using the environment variable T32CONFIG), and share the rest of the configuration with other systems.

[Examples]

Format:	**PBI**= *<host_interface>* *<special_commands>* **...** **empty line**
<host_interface>:	**USB**
	USB **NODE=***<usb_device_name>*
	ADDRESS=*<address>*
	CITRIX
	PARPORT=*<number>*
	NET **NODE=***<node>* **CONNECTIONMODE=***<mode>*
<connection_mode>:	**AUTOABORT** **AUTOCONNECT** **AUTORETRY** **NORMAL** **QUERYCONNECT**
<special_commands>:	**AUTOABORT** (deprecated) **AUTOCONNECT** (deprecated) **AUTORETRY** (deprecated) **BROADCAST** **CORE=***<n>* **INSTANCE=***<n>* **ETHERNETADDRESS=***<phy_address>* **HOSTPORT=***<n>* **PORT=***<n>* **SMALLBLOCKS** **USE=***<bits>* **PROXYNAME**=*<ip_address>* **PROXYPORT**=*<port_number>* **DNSRETRIES=***<n>*

Connection Modes	not applicable for multicore debugging
AUTOABORT	If debugger module is already in use, the TRACE32 executable will be closed automatically without any user interaction.
AUTOCONNECT	The TRACE32 executable will automatically take over control over the debugger module, even if the debugger is already in use.
AUTORETRY	If debugger module is already in use, the TRACE32 executable will wait until the current TRACE32 session ends.
NORMAL	If PowerDebug device is already in use, warn user and close application after confirmation.
QUERYCONNECT	If PowerDebug device is already in use, ask user if connection shall be forced.

Special Commands	
AUTOABORT AUTOCONNECT AUTORETRY	obsolete - please use CONNECTIONMODE=<*mode*> instead
CORE	Defines the default chip coordinate. In case INSTANCE is left out it, CORE will be also used as INSTANCE parameter. `CORE=0` means the TRACE32 executable that uses this setting in a config file has exclusive access to a particular PowerDebug module. No other TRACE32 executable can connect to this particular PowerDebug module. You can return the value of CORE with the **SYStem.USECORE()** function.
INSTANCE	Defines the internal communication channel between TRACE32 executable and debug interface. Set to 0 for single-core debugging (default if not specified). For multi-core debugging (several instances of TRACE32 connect to the same debugger module), use a unique number between 1..16 for each instance. You can return the value of INSTANCE with the **SYStem.INSTANCE()** function.
HOSTPORT	Defines the UDP communication port from the debugger module to the PC (default is PORT+n)

Special Commands	
PORT	Sets the UDP communication port from the PC to the debugger module (default is 20000). For AMP, all instances must use the same port number.
SMALLBLOCKS	It restricts the default communication block size to 4.5 KBytes instead of 16KB. This may avoid problems with network equipment which have resource restrictions (e.g. internal buffer overflows)
USE	The USEMASK selects the Lauterbach device, when several devices are connected to each other via PODBUS IN/OUT. For a description and an example, see **SYStem.USEMASK()** function.
PROXYNAME=*<ip_address>* PROXYPORT=*<port_number>*	TRACE32 allows to communicate with a POWER DEBUG INTERFACE USB from a remote PC. For an example, see **"Example: Remote Control for POWER DEBUG INTERFACE / USB"**, page 60.
DNSRETRIES=*<n>*	Default: DNSRETRIES=0 Sets the number of retries when trying to resolve a DNS name of a TRACE32 device on start-up of the host software. Before each retry there is a 1 second wait time.

Examples

In this section:

Examples for <host_interface>

[Back to Examples]

```
PBI=                    ; TRACE32 development tool is connected to the
USB                     ; PC via USB
```

```
PBI=                    ; TRACE32 development tool is connected to the
USB                     ; PC via USB
NODE=T32-ARM
                        ; the TRACE32 development tool is identified by
                        ; a name (IFCONFIG)

                        ; a name is required if several TRACE32
                        ; development tools are connected via USB

                        ; the manufacturing default device name is the
                        ; serial number of the debug module
                        ; e.g. NODE=E18110012345

                        ; requires USB-Flash V8.0 and higher, requires
                        ; TRACE32 software build 8000. and higher
```

```
PBI=                    ; TRACE32 development tool is connected to
NET                     ; the host via ethernet
NODE=T32-ARM
```

```
PBI=                    ; TRACE32 development tool is connected to
PARPORT=1               ; the PC via parallel interface
```

```
PBI=                    ; TRACE32 development tool is connected to the
ADDRESS=632             ; PC via parallel interface

                        ; address of parallel port in decimal
```

```
PBI=                    ; TRACE32 development tool is connected to the
CITRIX                  ; PC via Citrix VD (Virtual Channel Driver,
                        ; for USB only)
```

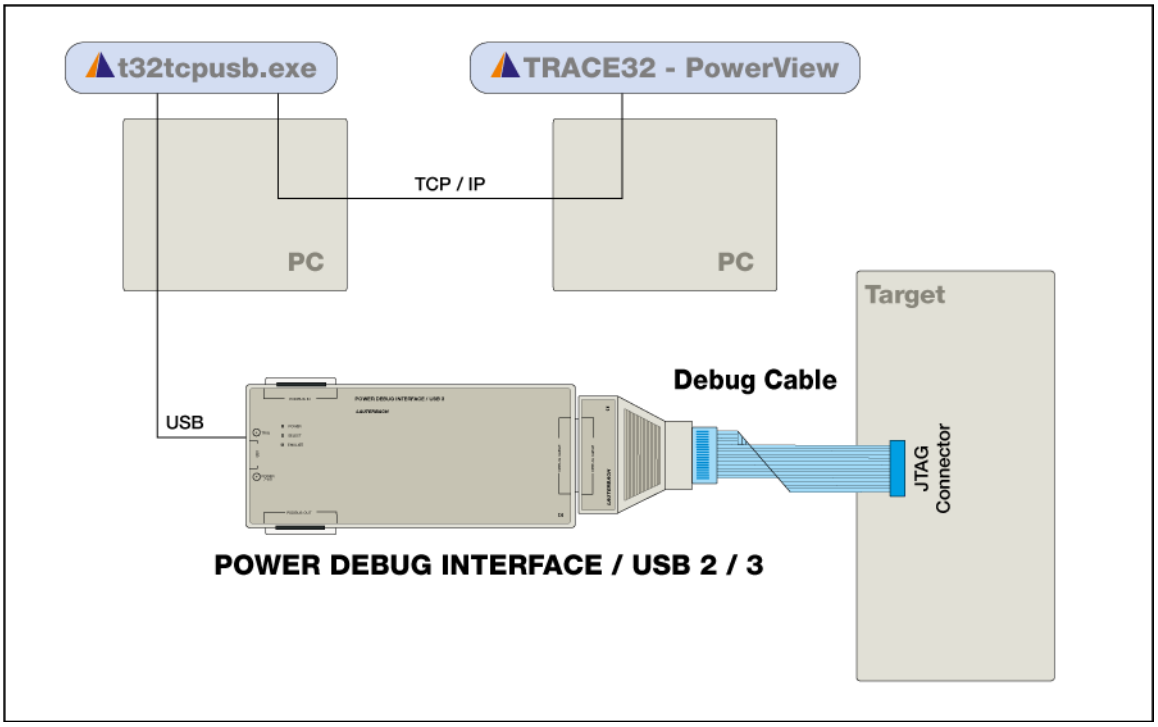
[Back to Examples]

```
PBI=                            ; Quit TRACE32 without any dialog when
USB                             ; connection to debugger fails
CONNECTIONMODE=AUTOABORT
```

```
PBI=                            ; The TRACE32 executable will automatically
NET                             ; take control over the debugger module,
NODE=T32-ARM                    ; even if the debugger is already in use.
CONNECTIONMODE=AUTOCONNECT
CORE=0                          ; necessary for DVD 09/2018 and newer
                                ; or use alternative: INSTANCE=0
```

[Back to Examples]

TRACE32 allows to communicate with a POWER DEBUG INTERFACE USB from a remote PC.



In order to implement this communication, the command line tool **t32tcpusb.exe** has to be started on the PC to which the POWER DEBUG INTERFACE USB is connected. **t32tcpusb.exe** can be found in the `bin/<target_os>` directory of your TRACE32 installation (e.g. `bin/windows64`).

t32tcpusb.exe *<port_number>*	The default port number is 8455

```
t32tcpusb.exe 8866
```

On the remote host computer the configuration file for USB has to be extended as follows:

Parameter	Syntax	Description
Host interface	PBI= USB PROXYNAME=*<ip_address>* PROXYPORT=*<port_number>*	 *<ip_address>* of PC that runs t32tcpusb.exe *<port_number>* that was specified when t32tcpusb.exe was started

```
; Host interface
PBI=
USB
PROXYNAME=10.2.20.142
PROXYPORT=8866

; Environment variables
OS=
ID=T32
TMP=C:\temp                     ; temporary directory for TRACE32
SYS=C:\t32                      ; system directory for TRACE32
HELP=C:\t32\pdf                 ; help directory for TRACE32

; Printer settings
PRINTER=WINDOWS                 ; all standard windows printer can be
                                ; used from the TRACE32 user interface
```

Format:	**PBI**=*<driver>* **…** **empty line**
<debug_ monitor_ driver>:	**COM***<n>* **DLL** *<dll>* **NET** *<address>*
<instr._set_ simulator_ driver>:	**SIM** ***SIM**
<virtual_ target_ driver>:	**CADI** [*<library_file>*] **GDI** *<library_file>* **MCD** *<library_file>* **MDI** **VAST** *<library_file>* **VDI** [*<library_file>*]
<3rd_party_ target_ server_ driver>:	**DAS** *<type>* **GDB** [*<communication_parameters>*] **GDB-EPOC** **OSE**
<3rd_party_ simulator_ driver>:	**GDI** *<dll>* **MDI** **SCS** **SIMTSI**

The TRACE32 PowerView GUI can be used without any LAUTERBACH hardware as debug front-end for:

- Debugging via a **TRACE32 debug monitor**
- Debugging via a **TRACE32 Instruction Set Simulator**
  - Customers who have a hardware-based TRACE32 debugger can use it to license the TRACE32 Instruction Simulator. For more information about `PBI=*SIM`, click **here**.
- Debugging via a **third-party target server**
- Debugging via a **third-party simulator**
- Debugging a **virtual target**
- Debugging **PPC Target Boards**

```
PBI=COM2 baud=38400      ; start TRACE32 for debugging via a monitor
                         ; program on the target

                         ; use RS232 as communication interface
```

```
PBI=NET 10.2.0.208       ; start TRACE32 for debugging via a monitor
                         ; program on the target

                         ; use ethernet as communication interface
```

```
PBI=DLL C16xcan.DLL      ; start TRACE32 for debugging via a monitor
                         ; program on the target

                         ; communication is performed with a user-defined
                         ; DLL
```

TRACE32 Instruction Set Simulator

```
PBI=SIM                  ; start TRACE32 to debug via a TRACE32
                         ; instruction set simulator
```

TRACE32 Instruction Set Simulator Licenced via a Hardware-based TRACE32 Debugger

If the hardware-based TRACE32 debugger is connected to the host computer by USB, the following lines have to be added to the TRACE32 configuration file:

```
PBI=*SIM                 ; start TRACE32 to debug via a TRACE32
USB                      ; instruction set simulator
```

If the hardware-based TRACE32 debugger is connected to the host computer by ethernet, the following lines have to be added to the TRACE32 configuration file:

```
PBI=*SIM                 ; start TRACE32 to debug via a TRACE32
NET                      ; instruction set simulator
NODE=<node_name>
```

```
PBI=GDB                  ; start TRACE32 for Linux debugging via
                         ; gdbserver or T32server
```

```
PBI=GDB COM1 baud=115200     ; start TRACE32 for Linux debugging via
                             ; gdbserver or T32server and use RS232 as
                             ; communication channel
```

```
PBI=GDB 127.0.0.1:30000      ; start TRACE32 for Linux debugging via
                             ; gdbserver or T32server and use TCP/IP as
                             ; communication channel by specifying
                             ; <remote_target_ip_address>:<gdb_port>
```

```
PBI=GDB localhost:30000      ; start TRACE32 for Linux debugging via
                             ; gdbserver or T32server and use TCP/IP as
                             ; communication channel by specifying
                             ; <host_name>:<gdb_port>
```

```
PBI=GDB-EPOC COM1 baud=115200          ; start TRACE32 for EPOC debugging
                                       ; via EPOC gdbstub
                                       ; use RS232 as communication
                                       ; interface
```

```
PBI=OSE                  ; start TRACE32 to debug OSE via the OSE debug
                         ; server
```

Third-Party Simulator

```
PBI=GDI tsim.dll         ; start TRACE32 to debug via the Infineon
                         ; TriCore simulator
```

```
PBI=MDI                  ; start TRACE32 to debug via the MDI (MIPS debug
                         ; interface) simulator
```

```
PBI=SCS                 ; start TRACE32 to debug via the SCS (StarCore)
                        ; simulator
```

```
PBI=SIMTSI              ; start TRACE32 to debug via the Target Server
                        ; Simulator from Texas Instruments
```

Virtual Targets

For examples of configuration files, see:

- **"Modifying the Configuration Files in Windows"** (virtual_targets.pdf).
- **"Modifying the Configuration File in Linux"** (virtual_targets.pdf).

PPC Target Boards

```
PBI=ADS                 ; decimal address of interface port
ADDRESS=256
```

```
PBI=EBDI                ; decimal address of parallel port
ADDRESS=888
```

Format:	**PBI**=*<driver>* *<special_commands>*
<driver>:	**MCISERVER**
<special_commands>:	**CORE=1**…**254** **INSTANCE=AUTO** | **1**…**16** **NODE=***<ipv4>* | *<host_name>* **PORT=***<typ_port>* **DEDICATED** **AUTOSTART**

The TRACE32 PowerView GUI can be used without any LAUTERBACH hardware as debug back-end. This section describes the configuration file options for debug back-ends.

Description of <special_commands>: PBI=MCISERVER

AUTOSTART	starts an instance of t32mciserver when the server has not started yet at local host and the option DEDICATED is also set. Do use this option wen t32mciserver must be started at a certain point in time or special parameters.
CORE	Defines the default chip coordinate. In case INSTANCE is left out it, CORE will be also used as INSTANCE parameter. `CORE=0` means the TRACE32 executable that uses this setting in a config file has exclusive access to a particular PowerDebug module. No other TRACE32 executable can connect to this particular PowerDebug module. You can return the value of CORE with the **SYStem.USECORE()** function.
DEDICATED	Prohibits to start the integrated MCI-server when there is no response from an already started MCI-server at localhost. Use this option when t32mciserver shall be started explicitly at localhost.
INSTANCE	Defines the internal communication channel between TRACE32 executable and debug interface. Set to 0 for single-core debugging (default if not specified). For multi-core debugging (several instances of TRACE32 connect to the same debugger module), use a unique number between 1..16 for each instance. You can return the value of INSTANCE with the **SYStem.INSTANCE()** function.

NODE	IP address or hostname of MCI-Server. In ideal case this must be a machine with very low latency (<100us) to the target. The localhost is used as hostname when NODE is left out. An integrated MCI-server is started when NODE is localhost and the option DEDICATED is left out. The integrated MCI-server runs in the first started GUI. An instance of t32mciserver as dedicated MCI-server is started when NODE is localhost and the options DEDICATED and AUTOSTART are used. A dedicated MCI-server should be used when interaction with the console is required or to keep the latency small by running it at the simulation/emulation host.
PORT	TCP-port number of MCI-Server. All started PowerView GUIs that belong to one target system must use the same PORT and NODE in order to connect to the same MCI-server. The used port of the dedicated MCI-Server is passed by a command line parameter of t32mciserver[.exe]. The port number 30000 is used as TCP-port when the PORT option is left out.

Other Configuration Scenarios

Other TRACE32 configuration parameters are described in the manuals pertaining to the various TRACE32 products.

What do you want to do?	Configuration is described here:
To start TRACE32 as GDB front-end	**"TRACE32 as GDB Front-End"** (frontend_gdb.pdf)
To set up communication with TRACE32 via remote control	**"TRACE32 as GDB Back-End"** (backend_gdb.pdf)
To start TRACE32 as front-end and GTL back-end	**"PowerView System Configurations"** (backend_gtl.pdf)
To start TRACE32 as front-end and XCP back-end	**"PowerView System Configurations"** (backend_xcp.pdf)
To start TRACE32 as front-end and DAS back-end	**"Debugging via Infineon DAS Server"** (backend_das.pdf)
To start TRACE32 as front-end and DCI DbC back-end	• **"Connecting to an Intel® SoC using DCI DbC"** (dci_intel_user.pdf) - or - • **"Connecting to an Intel® Client or Server System using DCI DbC"** (dci_intel_user.pdf)

Software Installation

In this chapter:

For information about floating licenses, refer to **"Floating Licenses"** (floatinglicenses.pdf).

Floating Licenses

For information about floating licenses, refer to **"Floating Licenses"** (floatinglicenses.pdf).

Quick Installation for controller-based debugging

The installation can be done by starting “setup.bat” or "files\bin\setup64\setup.exe" (WIN2000/XP/WIN Server 2003/WIN Vista/WIN7/WIN8/WIN10).

NOTE:	**To install TRACE32 on WINDOWS, you have to be logged in as an administrator.**

Three different driver programs are available for Windows:

t32win.exe (formerly t32w95.exe) for Windows 2000/XP/Vista/7/8/10

Two emulators can share one PC and the WINDOWS environment. The name of the configuration file can be defined in the command line with the option '-c'. The command line for an application may be defined by WINDOWS. The command line

```
t32win.exe -c c:\t32\configw.t32
```

will start the driver with the configuration file 'configw.t32'.

NOTE:	You can configure the TRACE32 help system with a few mouse-clicks to display the PDF help files in your favorite PDF viewer; see **“Configure the Help System”** (ide_user.pdf).

Example of a configuration script for a TRACE32 ICE:

```
LINK=PAR

SCREEN=
HEADER=TRACE32-68020
MWI

PRINTER=WINDOWS
```

USB Interface

The driver must be selected. Windows 2000 / XP / Vista / 7 / 8 or 10 is required.

When the device is first connected to the system, the hardware assistant detects a new USB device and asks for a driver directory.

If the TRACE32 software is already installed, the required file (t32usb.inf) can be found in the TRACE32 installation directory (e.g. c:\t32\). Otherwise please insert the TRACE32 installation CD and let the system search for it or navigate to the directory \bin\windows\drivers..

Configuration Command:

LINK=USB	Select USB connection

Prerequisite for Windows 7 and higher: Install the “TRACE32 PODBUS USB driver for Windows”.
The USB driver installer is located in ~~/files/bin/windows/drivers or ~~/files/bin/windows64/drivers.
To install, double-click `dpinstselect.exe`.

Controller-based Ethernet setup

The Ethernet connection requires the driver program with networking capabilities 't32w32.exe'. This program requires that a WINSOCK compatible TCP/IP provider is installed. First a new node must be created for TRACE32. The Ethernet address of the emulator is on a sticker located on the bottom or back side of the system. The following line must be added to the file HOSTS:

```
192.9.200.5    t32
```

Note, the above used INTERNET address is an example only. Contact your network administrator for a new INTERNET address for TRACE32. When an RARP server is used, the Ethernet address of the system must be entered in the file ETHERS:

```
0:c0:8a:0:0:0   t32
```

The INTERNET address is requested by a RARP protocol by TRACE32. If no RARP server is running, the address for the first connect must be set in the host table. After the first successful connect the INTERNET address is stored in nonvolatile memory within TRACE32. The following command sets the host translation table:

```
arp -s 192.9.200.5 0-c0-8a-0-0-0
```

NOTE:	**Windows 95 has a bug, that may cause the arp command to fail, when the arp cache is empty. In this case 'ping' another host before executing the arp command, this will fill the arp cache.**

NOTE:	**On Windows NT the ARP command is only available if you are logged in as an administrator.**

If the ARP command is not available the internet address must be set by connecting the system via fiber optic link/parallel interface.

To use the network access, the net driver must be activated. The node name can be changed, when not identical to 't32'.

Configuration Command:

LINK=NET	Used for controller-based debugging
NODE=<*node_name*>	(default: t32)
PACKLEN=1024	Limits the size of the UDP packages to 1024
HANDSHAKE	Use handshake after each packet send to host
SMALLBLOCKS	Limits the size of packet bursts to 4.5 KBytes (useful when switches or routers cannot buffer larger packets)

This driver must be selected. As a default the address 378 (hex) is used.

NOTE: If starting-up the system immediately after power-up the message "LINK ERROR …" will be displayed. This message can be ignored, the boot sequence will be started normally after the self-test has been finished.

If the interface is not working, check the BIOS setup for the Parallel Port. For the standard configuration it should be set to EPP 1.9.

Configuration Command:

LINK=PAR	Used for controller-based debugging

Commands:

LPT2	Use LPT2 instead of LPT1 (Preferable over ADDRESS = xxx)
LPT3	dto.
ADDRESS = *<address>*	Use address to access the parallel interface. Some common used addresses are: 378 (hex) 888 (decimal) for LPT1 278 (hex) 632 (decimal) for LPT2 3bc (hex) 956 (decimal) for LPT on MGA
EPP	Use EPP mode.
ECP	Use ECP mode.
SLOW	Changes the handling to a more reliable, but slower method.
FAST	Changes the handling to a faster method. This may not work on all PCs.
SLOWEPP	Uses EPP mode to speed up transfer to TRACE32.
FASTEPP	Same as above and use of FAST mode for to receive data from TRACE32.
SLOWFIFO	Uses ECP buffered mode to speed up transfer.
FASTFIFO	Same as above and use of FAST mode for to receive data from TRACE32.

The efficiency of the parallel port (and the optimum configuration) can be checked by the **IFCONFIG.PROfile.TESTRECV** and **IFCONFIG.PROfile.TESTSEND** commands. The **IFCONFIG.PROfile.RESYNC** command can show errors during the transfer. NOTE: To reach optimum performance the software is "self learning". This will result in varying transfer rates during the learn process.

When the Parallel Interface is working in the standard mode, try out the following modes:

FASTFIFO	This will give the best performance on ECP compatible cards. NOTE: You must select the ECP port mode in your BIOS setup!
SLOWFIFO	This mode can be used, when FASTFIFO fails. The download speed is the same, only the upload speed may be slower (only about 10%).
FASTEPP	If you have no ECP Port, but only an EPP Port, this mode can give a performance increase over the standard modes. Check the performance by repeating the **PROFILE.TESTRECV** command at least ten times.
SLOWEPP	Can be used, when FASTEPP does not work. The download speed is the same, only the upload speed may be slower (only about 10%). Check the performance by repeating the **PROFILE.TESTRECV** command at least ten times.
FAST	Can be used, when neither the FIFO nor the EPP modes work. Check the performance by repeating the **PROFILE.TESTRECV** command at least ten times.
SLOW	Must be used when no other modes (including the default mode) work. May also be required when long or bad printer cables are used.

Fiber Optic Interface

The driver for the fiber optic interface in polling mode is the default driver. The DMA and interrupt switches on the interface card must be set to off. This configuration can be used for ISA and MCA interface cards.

Configuration Command:

LINK=	Used for controller-based debugging

Commands:

ADDRESS = 864	Defining interface address (default is 864 dec, 360 hex)
BAUDRATE = 1	Limit the baud rate to 1 or 2 Mbit

The special fonts will be loaded by the driver program, if they are not already (fixed) loaded. They should be manually loaded when more than one debugger is used simultaneously.

Configuration file (config.t32) commands:

Configuration Command:

SCREEN=

Commands:

MDI	MDI Multiple Document Interface (default) All child windows appear inside the TRACE32 main window.
FDI	Floating Document Interface All child windows can be placed on any position on the desktop independently form the main window. Minimizing the main window, minimizes also the child windows. Only the main window appears in the task bar.
MTI	Multiple Top-level window Interface All child windows can be placed on any position on the desktop independently form the main window. Minimizing the main window, minimizes none of the child windows. Both the main and all child windows appear in the task bar.
MWI	Multiple Window Interface (obsolete alias for FDI). See FDI
OFF or INVISIBLE	TRACE32 as a hidden instance. For details, see **below**.
PALETTE <*n*> = <*red*><*green*><*blue*>	Change color value, the intensities will vary from 0 to 255.
FONT=SMALL	Use small fonts (13x7 instead of 16x8). Alternatively, you can use MEDIUM or LARGE.
TEXTFONT=<*font_name*>	Use the specified font for text windows. The specified font must either match the size of the regular TRACE32 fonts, or must be a TrueType fixed pitch font. Examples for such fonts are "Courier New" or "Lucida Console".
CHARSET=<*number*>	Text windows and menus use the specified character set.
MINCHO	Use MS-Mincho character set (Japanese)
DPI=AUTO (default) Linux only	TRACE32 uses the OS DPI rate for font scaling.

DPI=<*value*> Linux only	User-defined DPI rate for font scaling. Range: `48` to `448`
DPI=NONE Linux only	No DPI scaling.
VLINES=<*lines*>	Startup size lines (default 35)
VCOLUMNS=<*col*>	Startup size columns (default 81)
VICON	Startup iconized (minimized)
VFULL	Startup full screen (maximized)
HEADER=<*name*>	Header text for the TRACE32 main window
LINES=<*lines*>	Max. number of lines (default 55)
COLUMNS=<*col*>	Max. number of columns (default 144)
other	Other commands are available for special purposes, they are not used in standard environments.

TRACE32 as a Hidden Instance

There are three ways to configure TRACE32 to start as a hidden instance. Choose one of the configuration options:

Option 1:
The main window of TRACE32 remains hidden. However, dialogs and other window of TRACE32 can still be opened. This is useful, for example, if an error occurs during a regression test.

```
SCREEN=
INVISIBLE
```

NOTE:
- INVISIBLE must be written below SCREEN= in the config.t32 file.
- On UNIX machines, the configuration option 1 requires a running X server.

Option 2:
The main window of TRACE32 and all other dialogs and windows of TRACE32 remain hidden - even if an error occurs.

```
SCREEN=OFF
```

NOTE:
- OFF must be written in the same line as SCREEN= in the config.t32 file.
- On Unix machines, the configuration option 2 does NOT require a running X server.
- However, the X libraries must be installed.

Option 3:
The main window of TRACE32 is added to the toolbar of the host computer. It can be fully displayed in case of an error.

```
SCREEN=
VICON
```

Japanese Font

The following example uses a Japanese font for text display.

```
SCREEN=
TEXTFONT=@MSxxxx      ; the Japanese font name cannot be reproduced here
CHARSET=128
```

You can also define the window position and size of TRACE32 using the **FramePOS** command or the **CmdPOS** command.

Configuration Command:

PRINTER=<*type*>	printer type (**RAW**,**NEC**,**LJ**,**PS**,**WINDOWS**) RAW: simple printer without graphics NEC: NEC compatible printer LJ: Laserjet compatible printer PS: Postscript printer WINDOWS: Windows native printer

Commands:

DEV=<*path*>	device name or file name
SPOOL=<*cmd*>	command is executed after printing
IBMSET	use PC-8 character set for HP-Laserjet II
NOR	Don't use reverse feed (e.g. CANON LJ-600) Only for NEC printer driver.

Quick Installation

In the following example the directory **/opt/t32** is used as the system directory.

The system directory is created with the following command:

```
mkdir /opt/t32                          # or similar
```

The files are extracted from the CD to the system directory with the following commands:

```
mount /mnt/cdrom                            # or similar
cd /opt/t32
cp -r /mnt/cdrom/files/*              ./

cp ./demo/practice/autostart.cmm      ./

mv bin/pc_linux/config.t32            ./    # not necessary if the TRACE32
                                            # executable is called with
chmod -R u+w *                              # configuration filename
                                            # parameter
                                            # e.g. t32marm -
                                            c/opt/t32/bin/pc_linux/config.
                                            t32

/mnt/cdrom/files/bin/pc_linux/filecvt ./    # converts all filenames to
                                            # lower case and files into
                                            # UNIX format
                                            # and uncompresses all files if
                                            # necessary
```

The following environment variables must be set (e.g. in .bashrc for the BASH-shell):

```
export T32SYS=/opt/t32
export T32TMP=/tmp
export T32ID=T32
```

Prepare and install the fonts:

Since TRACE32 software release April 2010 the font installation is simplified.
It's necessary to place a subdirectory named fonts (e.g. /opt/t32/fonts) under the TRACE32 system directory (e.g. /opt/t32). The TRACE32 PowerView software automatically searches for the required TRACE32 fonts in this directory if the fonts are not provided by the host operating system.

When bitmap fonts are blocked/locked from the host operating system, a usage overwrite can be activated by adding the following lines inside the actual used TRACE configuration file e.g. config.t32.

```
SCREEN=                         ; bitcoded values (0..3 allowed)
FONTMODE=3                      ; bit0: bitmap system fonts activated
                                ; bit1: bitmap TRACE32 client fonts
                                ; activated
```

Font installation for TRACE32 software releases older than April 2010:

```
cd /opt/t32/fonts
mkfontdir ./
xset +fp /opt/t32/fonts          # only temporary adding of TRACE32
xset  fp rehash                  # fontdirectory

chkfontpath -a /opt/t32/fonts    # permanent adding of the fontdirectory
                                 # not available under SUSE distribution
```

The TRACE32 fonts can be added alternatively to an existing font server configuration.

The **TRACE32 online help system** uses an external PDF viewer for displaying the information in PDF format.

Please execute the TRACE32 command **SETUP.PDFViewer.state** inside the TRACE32 PowerView GUI once.
If the autodetection fails, a manual setting will be necessary.

Legacy information for Acrobat Reader usage:

Download Acrobat Reader from http://www.adobe.com and install it if not already installed on the system. Usually, you have to be root for the installation!

```
tar -xvzf linux-508.tar.gz            # or similar filename
./INSTALL                             # run the install script
```

Set the environment variable “ACROBAT_PATH” to the Acrobat installation path:

```
export ACROBAT_PATH=/opt/Acrobat5          # added in ~/.bashrc for BASH

export ACROBAT_PATH=/opt/Adobe/Reader8     # added in ~/.bashrc for BASH
```

Copy the TRACE32 plug-in in the Acrobat plug_ins folder (without new line):

```
cp /mnt/cdrom/files/bin/pc_linux/trace32.api
                                  $ACROBAT_PATH/Reader/intellinux/plug_ins
```

If applicable, disable the Acrobat Reader configuration feature: Use only certified plug-ins

Verify that you have write permission to the system directory and prepare the configuration file **config.t32**:

```
cd /opt/t32/bin/pc_linux        # depends on the location of the actual
#   or                          # used configuration file - the default
cd /opt/t32                     # file location is /opt/t32 (==$T32SYS)

vi config.t32

…

LINK=NET                        # executable t32cde
NODE=t32
                                # please replace t32 with the actual
                                # assigned nodename for the ICE system

PBI=                            # executables t32m*
NET
NODE=t32                        # please replace t32 with the actual
                                # assigned nodename for the ICE system
```

Uncompress the executable files before usage (not necessary when filecvt was used before):

```
cd /opt/t32/bin/pc_linux
gzip -d t32*.gz                                  # or   gunzip t32*.gz
```

Include the executable file in the PATH variable:

```
export PATH=$PATH:/opt/t32/bin/pc_linux          # added in ~/.bashrc for
                                                 # BASH - preferred solution
```

NOTE:	You can configure the TRACE32 help system with a few mouse-clicks to display the PDF help files in your favorite PDF viewer; see **"Configure the Help System"** (ide_user.pdf).

Before the installation a new node must be created. The Ethernet address of the system is placed on the bottom side of the system. The following line must be added to the file /etc/hosts:

```
192.168.0.5    t32
```

Note that the INTERNET address given here is an example only. Contact your network administrator for a new INTERNET address for TRACE32. The Ethernet address of the system must be entered in the file /etc/ethers:

```
0:c0:8a:0:0:0   t32
```

The INTERNET address is requested by a RARP protocol by TRACE32. If no RARP server is running, the address for the first connect must be set in the host table. After the first successful connect the INTERNET address is stored in nonvolatile memory within TRACE32. The following command sets the host translation table:

```
arp -s t32 0:c0:8a:0:0:0
```

This command must be executed **immediately before** the first startup of the emulator. It is not required for future startups because the INTERNET address is stored in the emulator. The arp cache table should be checked just before the first startup with the command
'arp -a'.

The net driver must be activated. The node name can be changed, when not identical to 't32'

Configuration Command:

PBI= NET	driver section read from executables t32m*
NODE=<*node_name*>	Node name of TRACE32 (default: t32)
POOL=<*node_name*>, …	Define a set of nodes, which are scanned for connection.
LINK=NET	driver section read from executables t32cde
NODE=<*node_name*>	Node name of TRACE32 (default: t32)
POOL=<*node_name*>, …	Define a set of nodes, which are scanned for connection.

Quick Installation

In the following example the directory **/home/t32** is used as the system directory. The connection is made by Ethernet and the software is installed for Solaris 2.X.

The system directory is created with the following command:

```
mkdir /home/t32                     # or similar
```

The files are extracted from the CD to the system directory with the following commands:

```
mount -F hsfs -o ro /dev/dsk/c0t6d0s2 /cdrom/trace32
                                    # or similar

cd /home/t32

cp -r /cdrom/trace32/files/*        .

cp   ./demo/practice/autostart.cmm  .

mv bin/suns/config.t32 .            # not necessary if the TRACE32
                                    # executable is called with
                                    # configuration filename parameter
                                    # e.g. t32marm -c/home/t32/bin/suns/
                                    # config.t32

chmod -R u+w *

/cdrom/trace32/files/bin/suns/filecvt .
                                    # converts all filenames to lower

                                    # case and files into UNIX format and

                                    # uncompresses all files if necessary
```

Please modify the file config.t32 to your needs.

In the login script (e.g. .cshrc in the home directory for the C-shell) the following lines must be added:

```
setenv T32SYS /home/t32
setenv T32TMP /tmp
setenv T32ID  T32
```

Prepare and install the fonts:

```
cd /home/t32/fonts
mkfontdir .
xset +fp /home/t32/fonts
xset  fp rehash
```

The **TRACE32 online help system** uses an external PDF viewer for displaying the information in PDF format.

Please execute the TRACE32 command **SETUP.PDFViewer.state** inside the TRACE32 PowerView GUI once.
If the autodetection fails, a manual setting will be necessary.

Legacy information:

Download Acrobat Reader from http://www.adobe.com and install it if not already installed on the system. Usually, you have to be root for the installation!

```
gzip -d sol-508.tar.gz                # or similar filename
tar -xvf sol-508.tar
./INSTALL                             # run the install script
```

Set the Environment variable “ACROBAT_PATH” to the Acrobat installation path:

```
setenv ACROBAT_PATH /opt/Acrobat5     # added in ~/.cshrc for C-shell
```

Copy the TRACE32 plug-in in the Acrobat plug_ins folder (without new line):

```
cp /cdrom/files/bin/suns/trace32.api       $ACROBAT_PATH/Reader/sparcsol
                                           aris/plug_ins
```

If applicable, disable the Acrobat Reader configuration feature: Use only certified plug-ins

Verify that you have write permission to the system directory and the boot.t32 file and prepare the configuration file:

```
cd /home/t32/bin/suns        # depends on the location of the actual used
   # or                      # configuration file - the default file
cd /home/t32                 # location is /home/t32 (==$T32SYS)
vi config.t32
…

LINK=NET
NODE=t32                     # please replace t32 with the actual
                             # assigned nodename for the ICE


SCREEN=
WMGR=OW16
```

Uncompress the executable files before usage (not necessary when filecvt was used before):

```
cd /home/t32/bin/suns
gzip -d t32*.gz              # or gunzip t32*.gz
```

Include the executable file in the PATH variable:

```
setenv PATH $PATH:/home/t32/bin/suns    # added in ~/.cshrc for C-shell
     # or                               # preferred solution
```

Add the Ethernet node name to /etc/hosts and /etc/ethers (**the INTERNET address here is only an example**):

```
/etc/hosts                              /etc/ethers

192.9.200.5    t32                      0:c0:8a:0:12:34     t32
```

Execute the ARP command from the system administration level to define the translation between INTERNET and Ethernet address:

```
su
…
arp -s t32 0:c0:8a:0:12:34
^D

ping t32
…
t32 is alive.
```

Start the emulator driver program and study the other configuration options in this manual.

Example of a configuration script for SUN 3/60:

```
LINK=SCSI

SCREEN=
HORSPEED=110
VERTSPEED=120
BARSPEED=75
MIDISRIGHT

PRINTER=LJ
DEV=/tmp/lstfile
SPOOL=lp -c /tmp/lstfile
IBMSET
```

Example of a configuration script for Sparc station in a network:

```
LINK=NET
NODE=trace32_15

SCREEN=
WMGR=OW16

BOOT=
USER=t32

PRINTER=LJ
DEV=/tmp/lstfile
SPOOL=lp -c /tmp/lstfile
IBMSET

PRINTER=PS
DEV=/tmp/lstfile
SPOOL=prt -lpostscript /tmp/lstfile
```

If the TRACE32 system files resides in one directory for all systems, and not everybody should have 'write permission' to that directory, it is possible to define an extra user for TRACE32. Only this user may have the right to write into the system-directory, all 'regular' users read within this directory only. This special 'user' is defined in the configuration file:

Configuration Command:

BOOT=

NOLOCK

USER=<*user_name*>

This special user must be the owner of the executable file 't32' and the system directory. The 'Set-User-Id' bit of the 't32' file must be set:

```
chmod ugo+s t32
```

Another solution to this problem is to store the boot files in the temporary directory. In this case the keyword **NOLOCK** must be used after BOOT= . The rights to write to the system directory must be disabled.

The environment variable T32CONFIG can be used to define individual configuration scripts for each user.

Before installation a new node must be created. The Ethernet address of the emulator is on a sticker located on the bottom or back side of the system. The following line must be added to the file /etc/hosts:

```
192.9.200.5    t32
```

Note, the above used INTERNET address is an example only. Contact your network administrator for a new INTERNET address for TRACE32. The Ethernet address of the system must be entered in the file /etc/ethers:

```
0:c0:8a:0:0:0   t32
```

The INTERNET address is requested by a RARP protocol by TRACE32. If no RARP server is running, the address for the first connect must be set in the host table. After the first successful connect the INTERNET address is stored in nonvolatile memory within TRACE32. The following command sets the host translation table:

```
arp -s t32 0:c0:8a:0:0:0
```

This command must be executed **immediately before** the first startup of the emulator. It is not required for future startups because the INTERNET address is stored in the emulator. The arp cache table should be checked just before the first startup with the command 'arp -a'.

The net driver must be activated. The node name can be changed, when not identical to 't32'.

Configuration Command:

LINK=NET	
NODE=<*node_name*>	Node name of TRACE32 (default: t32)
POOL=<*node_name*>, …	Define a set of nodes, which are scanned for connection.

For the SCSI interface a new device node must be created in the /dev directory. Check that the chosen combination isn't currently in use, otherwise the standard address cannot be used. The creation of the node must be done from the system administration level.

```
cd /dev
mknod t32 c 17 16
chmod a+r t32
chmod a+w t32
```

Solaris 2.X recognizes SCSI devices during the boot phase and creates automatically device nodes. This behavior can be achieved with the command

```
touch /reconfigure
```

which forces a scanning of the SCSI bus during the next booting or just boot the system with

```
boot -r
```

After this only the access rights of the node has to be changed with

```
chmod a+rw /dev/rdsk/c0t2d0s0
```

This device node is only the example for the use of the standard SCSI ID 2.

Configuration Command:

LINK=SCSI

If the boot EPROM version is 2.3 or older, then the **pins 28 and 32** of the SER module must be shortened (selection of fiber optic with DMA). The workstation should be turned off, when connecting the SASO interface. The SCSI address should be selected according to the table below. The address must be selected on the SASO module (see also description SASO module).

SCSI address	diskname	workstation	mknod
1	sd2	SUN-3	17 16
2	sd4	SUN-3	17 32
2	sd2	Sparc	17 16
0	sd3	Sparc	17 24

Example for the configuration command under Solaris 2.X

```
LINK=SCSI
DEV=/dev/rdsk/c0t2d0s0
```

The cable between the 37-pin TRACE32 connector and a 25-pin D-Sub connector on the host side must be connected in the following way:

(TRACE32-SER)	Pin 1, 21, 24, 26, 28 and 29 connected, Pin 1 with Host Pin 7, Pin 2 with Host Pin 3, Pin 3 with Host Pin 2.

Configuration Command:

LINK=RS232

Commands:

DEV=<*path*>	device name (default /dev/ttya)
BLOCKSIZE=<*size*>	block size (default 32)
HANDSHAKE=<*mode*>	handshake mode (default off)
BAUDRATE=<*code*>	baud rate code (default 38400) Common used codes (for SUN) are: 13 -> 9600 14 -> 19200 15 -> 38400 See the /use/include/sys/ttydev.h file for the specific codes of your host.

The block size should be set to 256 bytes on SUN.

Sample config.t32 file for 9600 baud:

LINK=RS232

DEV=/dev/ttya

BLOCKSIZE=256

BAUDRATE=13

Prior to starting the driver the special fonts of the TRACE32 system must be installed. There are two methods for doing this. The first method is to copy the files to the standard directory and include it in the system archive. This is the preferable method, when many users of a network system should be able to access the emulator. The second method is to keep the fonts in an extra directory. This method will be used, if there is no permission to write into the font directory.

On SunOS 4.1.x the fonts from the BDF format must be converted to the local format:

```
cd /home/t32/fonts
../bin/sun4/genfont.bat
rm *.bdf
```

The result should be a set of fonts in the X11/NEWS format (*.fb). NOTE: This conversion is not required on Solaris 2.x.

First method (adding the fonts to the system archive):

```
cd /usr/openwin/lib/X11/fonts/misc
cp /home/t32/fonts/*.bdf .
mkfontdir .
```

or for SunOS 4.1.x:

```
cd /usr/openwin/lib/fonts
cp /usr/t32/fonts/*.fb .
/usr/openwin/bin/bldfamily -f 20
```

The above commands must be entered from the system administration level, and the OpenWindow environment must be restarted after installing the fonts.

Second method (using an extra directory and set the FONTPATH):

The environment variable FONTPATH should refer to the './fonts' directory. It could be set in the '.profile' or '.login' batch, that is executed in the login procedure, but it must be set before the window system is started.

```
FONTPATH=${FONTPATH};/home/t32/fonts
```

or for SunOS 4.1.x:

```
FONTPATH=${FONTPATH};/usr/t32/fonts
```

The 'bldfamily' command in this directory must be executed:

```
cd /home/t32/fonts
mkfontdir .
```

or for SunOS 4.1.x:

```
cd /usr/t32/fonts
/usr/openwin/bin/bldfamily -f 20
```

Restart the OpenWindows environment.

Third method (using an extra directory and xset command):

The 'bldfamily' command in this directory must be executed:

```
cd /home/t32/fonts
mkfontdir .
```

or for SunOS 4.1.x:

```
cd /usr/t32/fonts
/usr/openwin/bin/bldfamily -f 20
```

The 'xset' commands add this directory to the FONTPATH:

```
xset +fp /home/t32/fonts
xset fp rehash
```

or for SunOS 4.1.x:

```
xset +fp /usr/t32/fonts
xset fp rehash
```

Notes for Solaris 2.X

A substitution for the third method of including a separately TRACE32 font path is to use a user specific file '$HOME/.OWfontpath' or a host specific file '$OPENWINHOME/lib/locale/$LOCALE/OWfontpath' with the TRACE32 font path inside.$LOCALE is the current selected language for the host (e.g. C for German).

Notes for CDE (Common Desktop Environment)

Normally the commands 'WMGR' and 'STYLE' aren't necessary. With the command **TRANSIENT** the system uses Transient Shell Windows as children. The default is to use Top Level Shell Windows. Transient Shell Windows are closer coupled together but cannot be minimized or maximized.

Notes for OpenWindows

OpenWindows cannot display colored icons. Use the configuration item **MONOICON** to select monochrome icons.

Configuration file (config.t32) commands:

Configuration Command:

SCREEN=

Commands:

TRANSIENT	Create TransientShells for client windows. The default is to use TopLevelShells. Transient Shells cannot be iconized or maximized. The visibility and stacking order is controlled by the main window.
XFORCE	Force window position of client windows by an extra X call, bypassing the window manager. This can be used if the windows cannot be placed at the wanted location (WINPOS command).
FONT=SMALL FONT=MEDIUM FONT=LARGE	Window Manager for Motif, smaller characters (13x7) dto., but larger characters (20x10). MEDIUM is the default setting.
FONT=DEC	alternative character size used (even) - could be used together with other FONT commands like FONT=SMALL
WMGR=MOTIF13 WMGR=MOTIF16 WMGR=MOTIF20	Window Manager for Motif, (13x7) dto., but larger characters (16x8) dto., but larger characters (20x10)
STYLE=GREY	Use own color scheme with grey background
STYLE=BLUE	White text on blue background for HP-Motif
STYLE=SAND	'Sand' colors for DEC systems

PALETTE <*n*> = <*red*><*green*> <*blue*>	Change color value, the intensities can vary from 0 to 65535.
VLINES=<*lines*> VCOLUMNS=<*col*> VICON HEADER=<*name*>	start-up size lines (default 35) start-up size columns (default 81) start-up as Icon header text for window
LINES=<*lines*> COLUMNS=<*col*>	max. number of lines (default 55) max. number of columns (default 144)
MONO	monochrome display on color screen
MONOICON	monochrome icons on color screen
EDITTRANSLATION=<*line*>	Adds the line to the translations for the editor windows
FIXVDESK	Workaround for problems with virtual desktops (minimized client windows after desktop switching)
other	other commands are available for special purposes, they are not used in standard environments

The command **SETUP.EXT BAK %** can be used to append a '**%**' to the backup files, instead of replacing the extension by '.bak'.

Terminal

The TRACE32 system can also be operated from a standard VT220 compatible terminal. The keyboard can be either VT220 or PC compatible. The following lines are required to use a VT220 terminal:

```
SCREEN=VT220

KEY=VT220

MOUSE=OFF

SOUND=TERM
```

Printer

Accessing a postscript printer with a customer specific command.

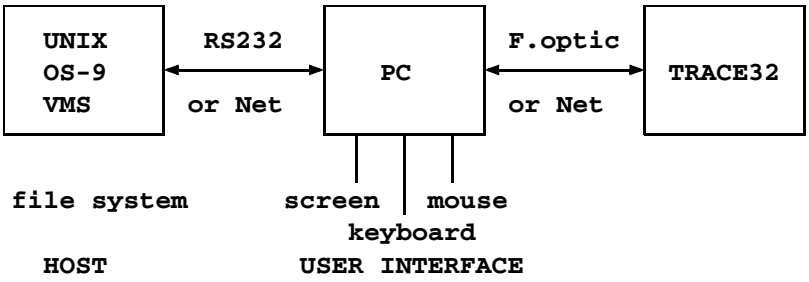
Configuration Command:

PRINTER=<*type*>	printer type (**RAW**,**NEC**,**LJ**,**PS**)

Commands:

DEV=<*path*>	device name or file name
SPOOL=<*cmd*>	command is executed after printing
IBMSET	use PC-8 character set for HP-Laserjet II
NOR	Don't use reverse feed (e.g. CANON LJ-600) Only for NEC printer driver.

```
PRINTER=PS
DEV=/tmp/${T32ID}lstfile
SPOOL=prt -lpostscript /tmp/${T32ID}lstfile
```



The REMOTE system allows to use two hosts for control, one for user interface and the other for the file system. This achieves a fast and comfortable working environment on OS-9 or UNIX systems together with terminals. An MS-DOS PC or another workstation can be used between the host and the emulator as an operating console. The connection between PC and emulator can be done with the fiber optic interface. The connection between the PC and the UNIX/OS-9 system is done by RS232 or via TCP/IP. On the MS-DOS configuration script the command

```
REMOTE=drvser.t32
```

will activated the REMOTE interface and selecting the serial interface. The remote configuration block must be the last one in the configuration script. The configuration of the serial interface is documented in the “MS-DOS installation procedure”. By this standard selection, the operating system and file interface from the host system is used. With the commands

```
KEY=REMOTE
BOOT=REMOTE
PRINTER=REMOTE
```

these interfaces can also be redirected to the UNIX system (normally not used).

In the UNIX system (file server) the screen and keyboard interface should be turned off to allow the driver to run in background mode:

```
SCREEN=OFF
MOUSE=OFF
KEY=OFF
SOUND=OFF
```

To execute programs on the MS-DOS system (**OS** commands) an '!' has to be added in front of the command:

```
E::OS !dir
```

The 'dir' command is executed on MS-DOS.

Example OS/9 together with PC

The following configuration describe a connection to an OS-9 system. The keyboard, screen and mouse functions are used on the PC. The printer is driven by to OS/9, and booting is done from the PC directly. The mouse will be connected to COM1, and the link to the OS-9 has to be done with COM2. For higher transmission rates an NS16550 should be used within the PC.

```
SCREEN=drvherc.t32

MOUSE=drvsumma.t32

PRINTER=REMOTE

REMOTE=drvser.t32
ADDRESS=760
BAUDRATE=19200
LIMIT=1000
```

configuration script for MS-DOS

```
SCREEN=OFF

KEY=OFF

MOUSE=OFF

SOUND=OFF

LINK=
DEV=/t1
BLOCKSIZE=256

PRINTER=RAW
DEV=/p1
```

configuration script for OS-9

Example VAX/VMS and Workstation

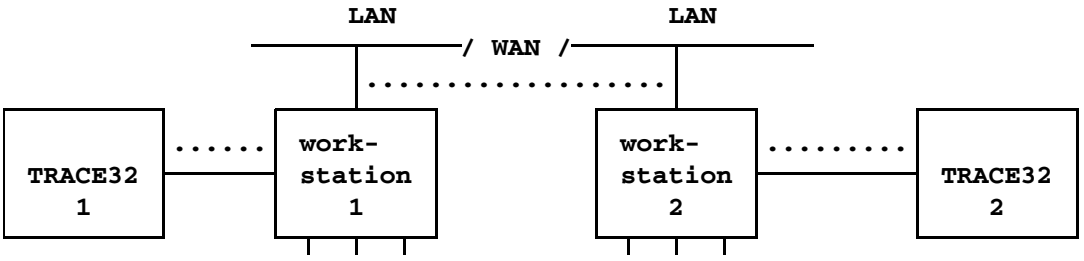
The following configuration text describes a connection to an VAX/VMS system. The keyboard, screen and mouse are used on the workstation and booting is done from the workstation directly. The emulator is connected to the workstation via Ethernet.

```
SCREEN=
WMGR=MOTIF16

LINK=NET
NODE=t32_emu

REMOTE=NETASSIST
```

configuration script for the Workstation

```
SCREEN=OFF

KEY=OFF

MOUSE=OFF

SOUND=OFF

LINK=NET
NODE=workstation1
```

configuration script for VAX/VMS



The configuration script contains the following lines:

```
IC=NETASSIST
PORT=20000
```

The port number is the UDP port number which is used to communicate with the other system. The default is 20000. If this port is already in use, try one higher than 20000. The **InterCom** system cannot be used together with Remote, Assist or RCL systems.

If you can not solve your problem with the following hints, please contact our support line:

telephone: ++49 8102/9876-555

facsimile: ++49 8102/9876-999

e-mail: support@lauterbach.com

System doesn't response to ping on Ethernet

Internet address already setup in system, or arp used?

When arp is used, it must be used on the same workstation short before.

Ethernet address correct?

System on the correct subnet?

Cables and transceiver o.k.?

Ethernet software in host (PC) configured correctly?

Executable program does not start or gives fatal error

When transferring between different OS-systems, files copied in binary mode?
Access rights to file in directory o.k.?

Configuration file contents o.k.?

Executable program displays 'FATAL ERROR selecting device-driver …'

Using configuration file for MS-DOS for the WINDOWS-Driver?

WINDOWS and workstation drivers cannot load new drivers.

Environment variable 'T32CONFIG' and/or 'T32SYS' correctly set?

Executable program displays 'error reading config.t32':'

Configuration file contents o.k.?

Commands in file in uppercase?

Blanks inserted/not inserted?

Device specific commands placed after device header?

Device configuration blocks separated by empty lines?

Environment variable 'T32CONFIG' and/or 'T32SYS' correctly set?

Executable program stops without message, but with window opened

Access rights to directory o.k.?

On UNIX host, try with 'NOLOCK' feature.

When using the RS232 interface: Is a login process active on the tty?

xset +fp fontpath gives error 'bad value …'

Does the font directory exist?

Does the fonts.dir file exist (created by mkfontdir)?

Is the directory seen under the same name by the X-server?

Have all directories that lead to the font directory read and execute permissions for everybody?

Program stops with message 'font xxxx not found'

Do fonts appear in the 'xlsfonts' command?

Can one font (e.g. t32-lsys-16) be displayed by 'xfd -fn t32-lsys-16'?

Fonts added to X-Windows FONTPATH?

Fonts converted, when required, and .bdf files removed?
(E.g. for TinyX you can use bdftopcf to convert to PCF and gzip to pack them.)

Command to generate font directory executed with correct parameters?

Fonts installed on the X-Windows server, not client?

If using an X-Terminal, use the conversion programs for the X-Terminal?

Executable program displays 'boot.t32 not found'

Access rights to directory o.k.?

Read and write access to boot.t32 (write required on UNIX without NOLOCK)?
Configuration file contents o.k.?

Environment variable 'T32SYS' correctly set?

Executable program stops after displaying 'error reading boot.t32'

When transferring between different OS-systems, files copied in binary mode? Access rights granted?

Try again after switching off the TRACE32 system?

Executable program displays 'illegal command <filename>'

Environment variable "T32CONFIG" correctly set?

Executable program stops after displaying 'booting …' or 'finished.'

When transferring between different OS-systems, files copied in binary mode? Packet size set correctly on Ethernet, handshake set when required?

Bootloader stops with message 'fatal error …'

When transferring between different OS-systems, files copied in binary mode? Mixing different versions of the software, e.g. MCC.T32 and MCCxxx.t32?

Bootloader displays 'cannot save image …'

Write access right on system directory?

Disk full?

Existing read-only file?

Software crashes or stops after booting is finished

Switches in the network path that cannot handle large bursts (try with SMALLBLOCKS)?

Boot image file maybe destroyed, remove all boot0x.t32 files?

Connection of modules o.k., connector bend?

Software doesn't work stable

Boot image file maybe destroyed, remove all boot0x.t32 files?

Connection of modules o.k., connector bend?

Check connection of Fibre Optic, Ethernet or Parallel interface.

On Ethernet try with smaller packet size and/or handshake.

Emulation system doesn't work correctly

Check Emulation Probe Manual in 'Targets' part of the manual.

Parallel Port not working stable

Check that the port is on the correct mode. Choose either EPP 1.9 or compatible mode. The mode selection can usually be done in the BIOS setup (can be activated during booting).

Executable program displays "file help.t32 could not be read"

The index file help.t32 should be in the trace32 system folder (SYS=<t32 system folder>), but was not found, was damaged or was too old. Download the actual help files from https://www.lauterbach.com

Executable program displays “Acrobat Reader could not be started”, please set §ACROBAT_PATH or run acroread manually

First check if the Acrobat Reader is installed properly and the environment variable "ACROBAT_PATH" is set correctly to the Acrobat installation directory. Then run ACROBAT_READER (./opt/Acrobat5/bin/acroread) and check if the plug-in “TRACE32” was loaded correctly: There should be a entry “About Trace32…” in the menu “help->About Plugins”. If not, copy the file “trace32.api” to the plug_ins folder of Acrobat Reader.

If the message is displayed only at Acrobat Reader startup, but then no further problems with the help occur, your system may be too slow, and you can ignore the message!

Fixed width font t32sys not found under WINDOWS

When you start the TRACE32 executable the fonts are loaded. If a SW update will be done, which replaces the TRACE32 font file named t32font.fon, the new fonts will not be activated as long as the old fonts are loaded.

This happens even if both font files are identical.

Please reboot your Windows PC to solve this issue.

FAQ

Please refer to our Frequently Asked Questions page on the Lauterbach website.