LAUTERBACH
DEVELOPMENT TOOLS

Native Process Debugger

Native Process Debugger

Native Process Debugger

Version 06-Jun-2024

Operation Theory

The TRACE32 Native Process Debugger is a software only debugger that allows to debug applications running on a Windows host. The minimum supported Windows version is Windows XP.

The TRACE32 Native Debugger is free and does not require any license.

Related Documents and Tutorials

- **“Training Simulator and Demo Software”** (demo.pdf) is a quick start tutorial that should make familiar with most important features of TRACE32 PowerView.

Configuration

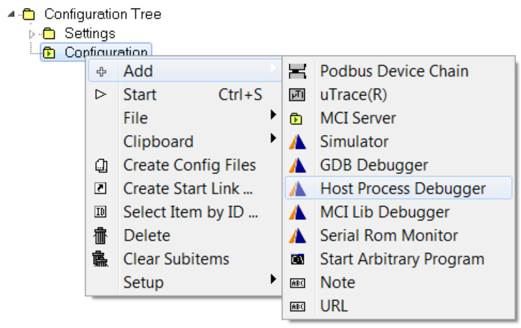
To start TRACE32 as Native Process debugger, the configuration file has to contain the following lines. The default configuration file is config.t32 and is located in the TRACE32 system directory.

```
                                          <- mandatory empty line
PBI=HOST
                                          <- mandatory empty line
```

!	**An empty line is mandatory before and after the PBI block in the TRACE32 configuration file, otherwise, a "syntax error" will be reported when starting TRACE32.**

For more information about the TRACE32 configuration, please refer to **"Training Basic Debugging"** (training_debugger.pdf).

In case you are using the t32start.exe utility to start TRACE32, you can simply add a new configuration by doing a right mouse click on **Configuration Tree** and select **Add > Configuration**. You can then add a **Host Process Debugger** to the new configuration.



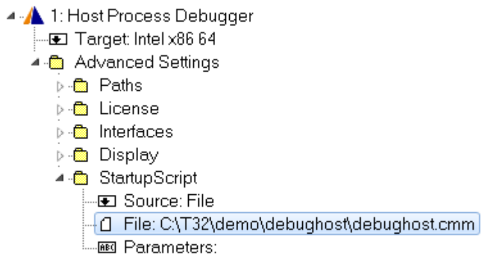
Please refer to the **"T32Start"** (app_t32start.pdf) manual for more information about the T32Start utility.

A collection of scripts is provided by Lauterbach to simplify the usage of the TRACE32 Native Debugger. These scripts can be found in the TRACE32 installation directory at `~~/demo/x86/etc/hostdebug`

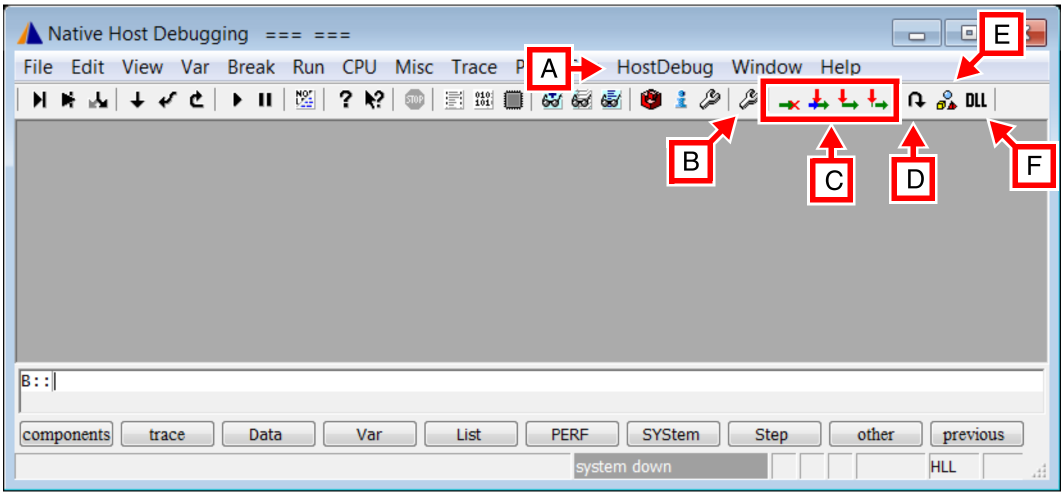
To configure the TRACE32 Native Debugger, run the script `hostdebug.cmm` using the TRACE32 menu **File > Run Script** or by executing the following command in the TRACE32 command line

```
DO ~~/demo/x86/etc/hostdebug/hostdebug.cmm
```

You can also automatically execute this script when starting TRACE32 by adding the command above to the **system-settings.cmm** file in the TRACE32 system directory (create the **system-settings.cmm** if it does not exist). See also **"Automatic Start-up Scripts"** (practice_user.pdf). In case you are using the t32start.exe utility, you need to specify the script at **Host Process Debugger > Advanced Settings > StartupScript > File**.


1: Host Process Debugger
Target: Intel x86 64
Advanced Settings
Paths
License
Interfaces
Display
StartupScript
Source: File
File: C:\T32\demo\debughost\debughost.cmm
Parameters:

The script adds buttons to the main tool bar of TRACE32 PowerView as well as a **HostDebug** menu.


Native Host Debugging === ===
File Edit View Var Break Run CPU Misc Trace P
HostDebug Window Help
A
B
C
D
E
F
DLL
B::
components trace Data Var List PERF SYStem Step other previous
system down
HLL

A HostDebug menu

B Project Setup

C Buttons to attach, detach or start a new process. From left to right:

1. **Disconnect/ Down**: detach from debugged process
2. **Attach...**: open process list window to attach to a running process
3. **Go**: start the process configured in the **Project setup** window. The process is directly resumed after start
4. **Up**: start the process configured in the **Project setup** window. The process is directly stopped after start

D Return to application code if stopped in a Windows DLL

E Display the thread of the debugged process

F Display the DLLs loaded by the debugged process

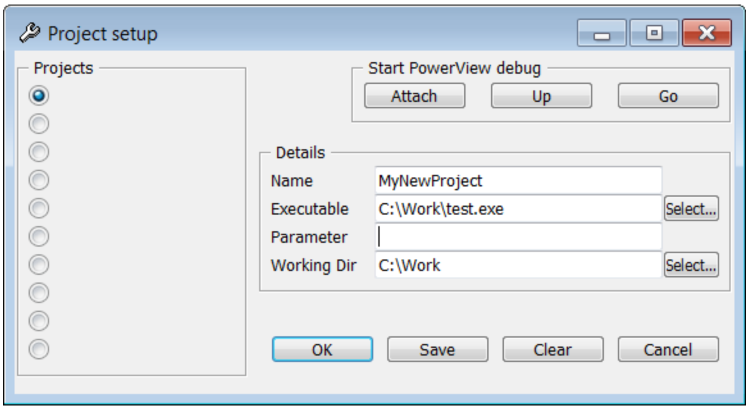
The same actions can be performed using the **HostDebug** menu. This menu contain the following items:

- **Attach...**: open process list window to attach to a running process
- **Up**: Start the process configured in the **Project Setup** window. The process is directly stopped after start
- **Go**: Start the process configured in the **Project Setup** window. The process is directly resumed after start
- **Disconnect/ Down**: detach from debugged process
- **Return to Application**: return to application if stopped in a Windows DLL
- **Project setup**: open the **Project Setup** window
- **User configuration**: open window to configure user preferences

Starting a New Process

If you have not already created a project, set up a new project using the following steps:

1. Open the **Project setup** window by selecting the menu **HostDebug > Project setup** or using the corresponding button from the main tool bar.



Name: MyNewProject
Executable: C:\Work\test.exe
Parameter:
Working Dir: C:\Work

2. Enter the project name in the **Name** field.
3. Enter the name of the executable in the **Executable** field.
4. Optionally specify the executable parameters and working directory in the **Parameter** and **Working Dir** fields.
5. Save the project using the **Save** button.

The process can be then started by selecting the menu **HostDebug > Up** or using the **Up** button from the main tool bar. The process will be executed and stopped at its first instruction. If you want to directly resume the process after start, use the menu **HostDebug > Go** or the **Go** button instead.

The process debug symbols will be automatically loaded by the TRACE32 symbol autoloader.

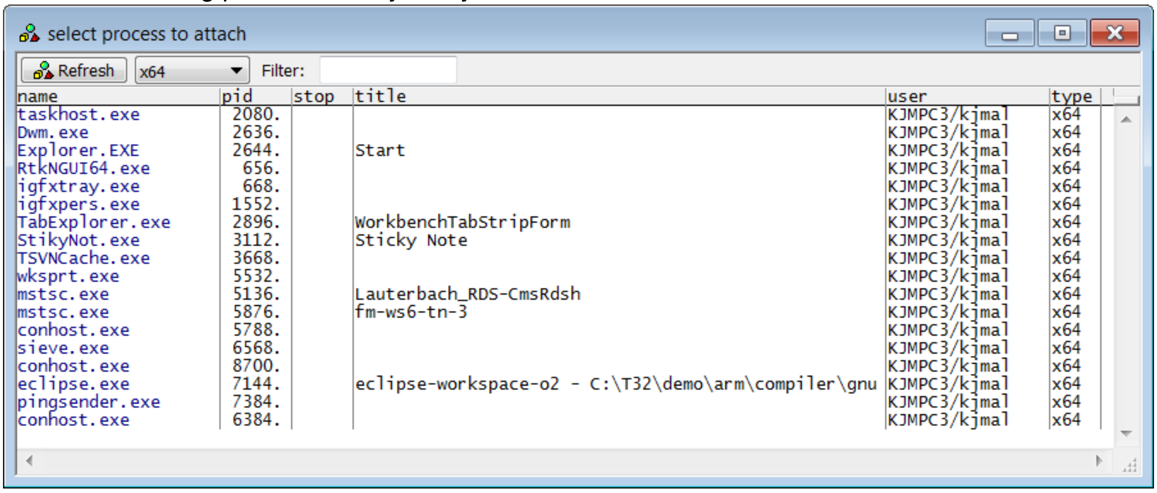
You can alternatively use a PRACTICE script to start a new process for debugging e.g.

```
SYStem.PROCess C:\Debugging\hello.exe arg1 arg2

; set up the autoloader script to load the symbols automatically after
; start
sYmbol.AutoLOAD.CHECKCMD "do C:\T32\debughost\nat_autoload.cmm "

SYStem.Mode Up       ; process is directly stopped after loading
; or
; SYStem.Mode Go     ; process is running after loading
```

Attach to a Running Process

To attach to a running process:

1. Select the menu **HostDebug > Attach...** or push the **Attach...** button in the main tollbar t open a list of all running processes on your system



name	pid	stop	title	user	type
taskhost.exe	2080.			KJMPC3/kjmal	x64
Dwm.exe	2636.			KJMPC3/kjmal	x64
Explorer.EXE	2644.		Start	KJMPC3/kjmal	x64
RtkNGUI64.exe	656.			KJMPC3/kjmal	x64
igfxtray.exe	668.			KJMPC3/kjmal	x64
igfxpers.exe	1552.			KJMPC3/kjmal	x64
TabExplorer.exe	2896.		WorkbenchTabStripForm	KJMPC3/kjmal	x64
StikyNot.exe	3112.		Sticky Note	KJMPC3/kjmal	x64
TSVNCache.exe	3668.			KJMPC3/kjmal	x64
wksprt.exe	5532.			KJMPC3/kjmal	x64
mstsc.exe	5136.		Lauterbach_RDS-CmsRdsh	KJMPC3/kjmal	x64
mstsc.exe	5876.		fm-ws6-tn-3	KJMPC3/kjmal	x64
conhost.exe	5788.			KJMPC3/kjmal	x64
sieve.exe	6568.			KJMPC3/kjmal	x64
conhost.exe	8700.			KJMPC3/kjmal	x64
eclipse.exe	7144.		eclipse-workspace-o2 - C:\T32\demo\arm\compiler\gnu	KJMPC3/kjmal	x64
pingsender.exe	7384.			KJMPC3/kjmal	x64
conhost.exe	6384.			KJMPC3/kjmal	x64

2. Select a process from the list by a double-click or using a right mouse click then select **Attach to process.** The process will be then highlighted in the process list window. The process debug symbols will be automatically loaded after attaching.

The steps described above can be achieved using the following PRACTICE commands

```
TASK.ListPROC                 ; display a list of running processes

; set up the autoloader script to load the symbols automatically after
; attach
sYmbol.AutoLOAD.CHECKCMD "do C:\T32\debughost\nat_autoload.cmm "

TASK.Attach 6568.             ; attach to process with pid 6568
```

Troubleshooting

No information available until yet.

Native Process Debugger Specific SYStem Commands

SYStem.Mode — Establish the communication with the process

Format:	**SYStem.Mode** <*mode*>
	SYStem.Down (alias for SYStem.Mode Down) **SYStem.Up** (alias for SYStem.Mode Up)
<*mode*>:	**Down** **Up**

Default: Down

Down	Default mode, no debugging.
Up	TRACE32 loads the process with the selected command line. Process is stopped.

SYStem.Option.IMASKASM — Disable interrupts while single stepping

Format:	**SYStem.Option.IMASKASM** [**ON** | **OFF**]

Default: OFF.
If enabled, the interrupt mask bits of the CPU will be set during assembler single-step operations. The interrupt routine is not executed during single-step operations. After a single step, the interrupt mask bits are restored to the value before the step.

SYStem.Option.IMASKHLL — Disable interrupts while HLL single stepping

Format:	**SYStem.Option.IMASKHLL** [**ON** | **OFF**]

Default: OFF.

If enabled, the interrupt mask bits of the CPU will be set during HLL single-step operations. The interrupt routine is not executed during single-step operations. After a single step, the interrupt mask bits are restored to the value before the step.

SYStem.PROCess — Set the process command line

Format: **SYStem.PROCess** <*cmdline*>

This command selects the command line of the process that will be loaded with **SYStem.Mode Up**. This step is not needed if you attach to a running process

```
SYStem.PROCess C:\Debugging\hello.exe arg1 arg2
```

SYStem.CurrentDir — Set the current directory for the process

Format: **SYStem.CurrentDir** <*path*>

Select the full path to the current directory for the process. If this is not set, the new process will have the same drive and current directory as the debugger.

```
SYStem.CurrentDir C:\Debugging
```

Native Process Debugger Specific TASK Commands

The Native Process Debugger supports the following TASK commands:

TASK.List.tasks	Display the list of running execution threads for the selected process.
TASK.ListPROC	Display the list of running processes on the system.
TASK.LIB	Display a list of loaded libraries for the selected process.
TASK.RUN	Start a new process for debugging.
TASK.ATTACH	Attach to a running process.
TASK.DETACH	Detach from process.
TASK.Go	Start the process execution.
TASK.Break	Stop the process execution.
TASK.select	Select a thread from the **TASK.List.tasks** window.
TASK.KILL	Kill a running process.
TASK.FREEZE	Freeze thread.
TASK.THAW	Resume a frozen thread.

TASK.ListPROC — Display the list of running processes

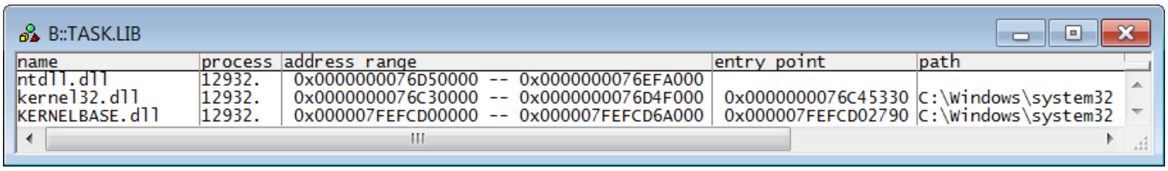
Format:	**TASK.ListPROC**

Display the list of running processes on the system.

TASK.LIB — Display the list of loaded libraries

Format:	**TASK.LIB**

Display the list of libraries loaded by the debugged process.



```
name            process address range                                      entry point        path
ntdll.dll       12932.   0x0000000076D50000 -- 0x0000000076EFA000
kernel32.dll    12932.   0x0000000076C30000 -- 0x0000000076D4F000   0x0000000076C45330 C:\Windows\system32
KERNELBASE.dll  12932.   0x000007FEFCD00000 -- 0x000007FEFCD6A000   0x000007FEFCD02790 C:\Windows\system32
```

TASK.FREEZE

Freeze a selected thread

Format:	**TASK.FREETZE** <*tid*>

TASK.THAW

Resume a frozen thread

Format:	**TASK.THAW** <*tid*>