TAS for DAS users

Using DAS tools and libraries with a TAS installation

About this document

Scope and purpose

Tool Access Sockets (TAS) is a standardized, lightweight interface for tools that utilizes Device Access Server (DAS). It provides the device connect, reset, and access functionality of DAS, and also supports additional features such as a session concept, channels, and continuous tracing.

The TAS installation includes a new das_api.dll which translates between DAS and TAS.

Intended audience

This document is intended for DAS tool users and developers of software and tools.

Table of contents

About this document **1**
Table of contents **2**
1 Introduction **3**
2 Comparison of DAS with TAS **4**
3 Troubleshooting **5**
3.1 Device not visible for DAS tool 5
3.2 TAS client error: session name or password 5
3.3 Problem: DAS remote access does not work 5
3.4 UDAS stops when a native TAS tool is started 5
3.5 Legacy DAS tool does not work 6
4 Emulator access with TAS **7**
5 DAS changes for TAS implementation **8**
Revision history **9**
Disclaimer **10**

1 Introduction

The TAS installation includes a das_api.dll file, which translates between DAS and TAS requests. This means that the usage of TAS will be transparent for DAS users.

Note: DAS V8.0 and its higher versions are based on TAS. Therefore, it is considered as a TAS installation.

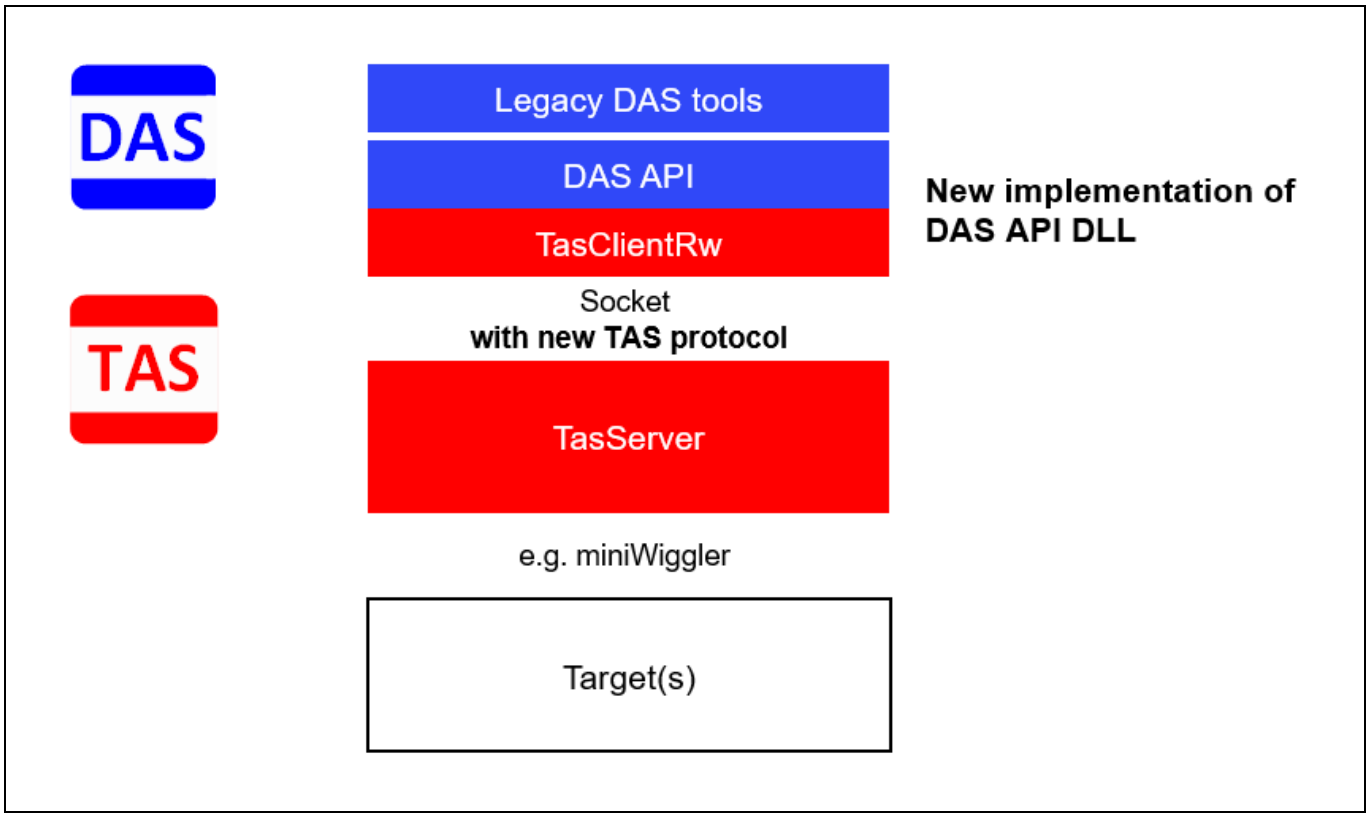


Figure 1 DAS tools layering on top of TAS infrastructure

Initially, TAS was designed for various host types such as Windows, Linux, and for others with minimum installation requirements. For a typical scenario, connecting a device using miniWiggler HW requires only the installation of USB drivers.

2 Comparison of DAS with TAS

DAS has a subset of TAS features. In addition to these features, TAS also includes a session concept, channels, and continuous tracing. DAS is standardized by the DAS API (das_api.h), while TAS is standardized by the client server socket protocol. DAS tools use the the DAS API DLL (das_api.dll), whereas TAS tools are compiled with the TasClient source code library.

The UDAS server of DAS is single threaded, whereas the TasServer uses its own thread for each client connection. Although this difference may not be significant for a single connected device, multi-threading can improve the performance and reduce the interference when several devices are accessed by multiple clients in parallel.

The TasServer has a fixed port number, eliminating the need for a DAS_HOME system variable and a running DASHPAS server. Therefore, the Tas Server can be started from any directory.

The following figure shows the information provided by DAS, MCD API (library on top of DAS), and TAS for two boards connected to a host computer.

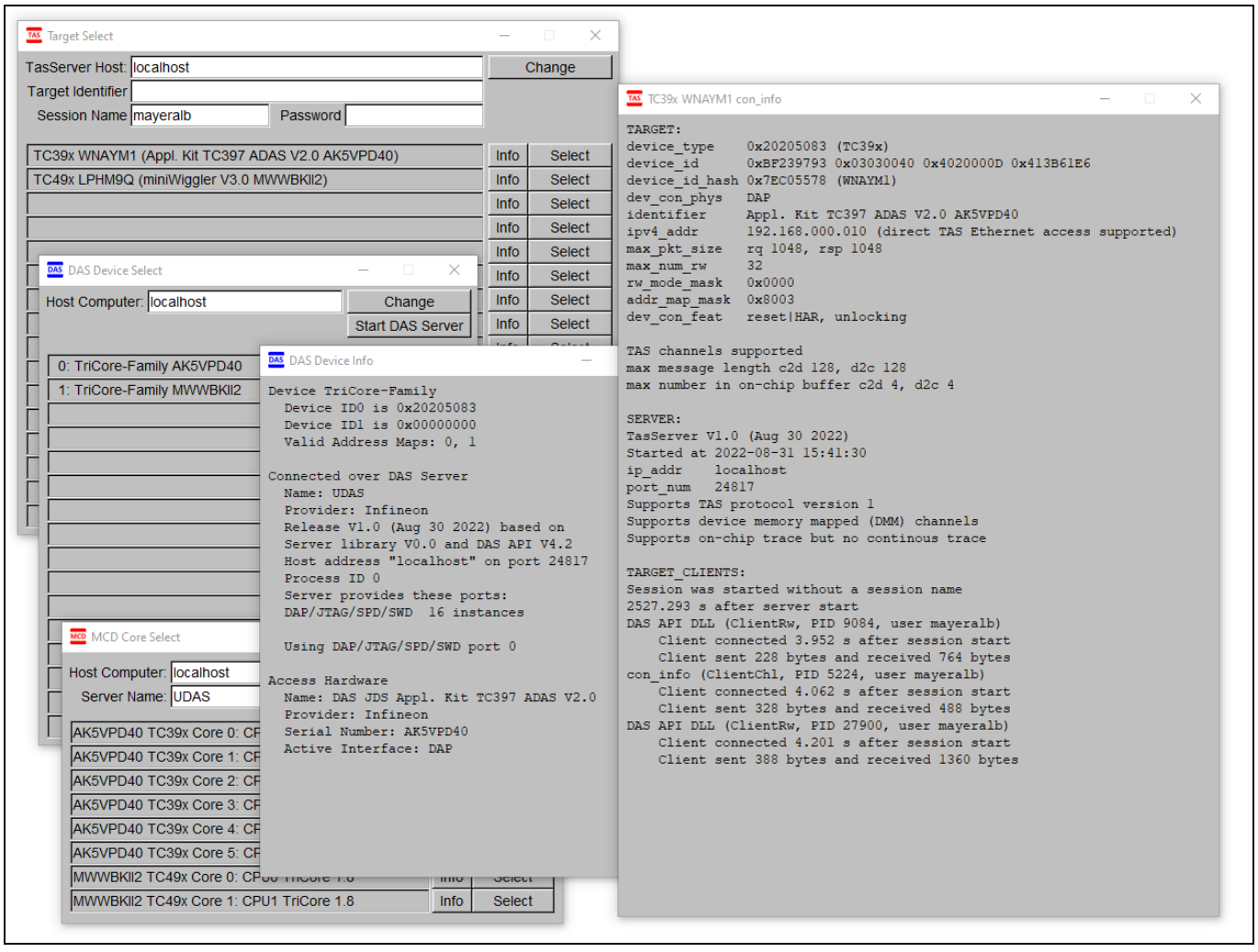


Figure 2 TAS, DAS, MCD scanner and info windows

3 Troubleshooting

The following recommendations assume that a TAS installer (or DAS V8.0) was used. In this case, a previous DAS installation is automatically removed. The TAS installation will provide DAS backward compatibility as described in this document.

3.1 Device not visible for DAS tool

Target visible in TAS Basic Client target select window but not by DAS tool

This issue will occur when the TAS session name is neither unnamed nor the username. To resolve this, terminate the TAS session by closing all involved clients and start a new unnamed session.

Target neither visible in TAS Basic Client nor by DAS tool

Firstly, investigate please HW-related issues like cables, power supply, and try with USB unplug, wait 2 seconds, and plug again.

Another possibility it that the TasServer is hanging. In this case, kill the TAS server (*tas_server.exe*) from Windows task manager. On Windows, it is automatically started again by the next client trying to connect.

Note that there is no DAS dashpas server anymore, which could have been crashed, and that the TasServer is displayed as “UDAS” server by DAS tools (Figure 2).

Only access HW (miniWiggler or on-board) visible

This means that the layers from tool to access HW are working properly. Hints for resolving this situation are not in the scope of this document.

3.2 TAS client error: session name or password

This will happen, for instance, when a DAS tool is started first and the TAS Basic Client afterwards. The DAS tool will start an unnamed session, while the TAS Basic Client sets the username as default session name (Figure 2). In this situation, the user just needs to delete the username from the session name entry in the GUI.

3.3 Problem: DAS remote access does not work

TAS and DAS use different protocols for the socket-based communication between client and server (Figure 1). This means that remote access will only work when all clients use a DAS API DLL that matches the protocol used by the server.

3.4 UDAS stops when a native TAS tool is started

This can only happen when a legacy DAS installation is active, which means that TAS was not installed in a regular way, because it would uninstall this legacy DAS installation. In this situation, the TAS Server, started manually from any directory, will automatically kill the legacy UDAS server. When using regular DAS and TAS installers this situation can, never happen.

3.5 Legacy DAS tool does not work

Tool uses a local das_api.dll

Since the local DLL will use the old DAS socket protocol (Figure 1), this setup cannot work. The simplest workaround is to delete the das_api.dll in the legacy tool directory. The DLL loading will then usually use the DLL search path and find the new installed version. However, this may not work if the DLL loading path is hardcoded in the tool. The option for this case is to replace the DLL by the version from the TAS installation directory. Note that the default DLL for TAS is 64-bit, but a 32-bit variant is also provided, which might be needed by older tools.

4 Emulator access with TAS

The emulator access with TAS does not require a specific server as it is established with the regular TasServer. The TasServer needs to know the IP address of the active emulator. It is provided as target identifier with the keyword edas or EDAS in front.

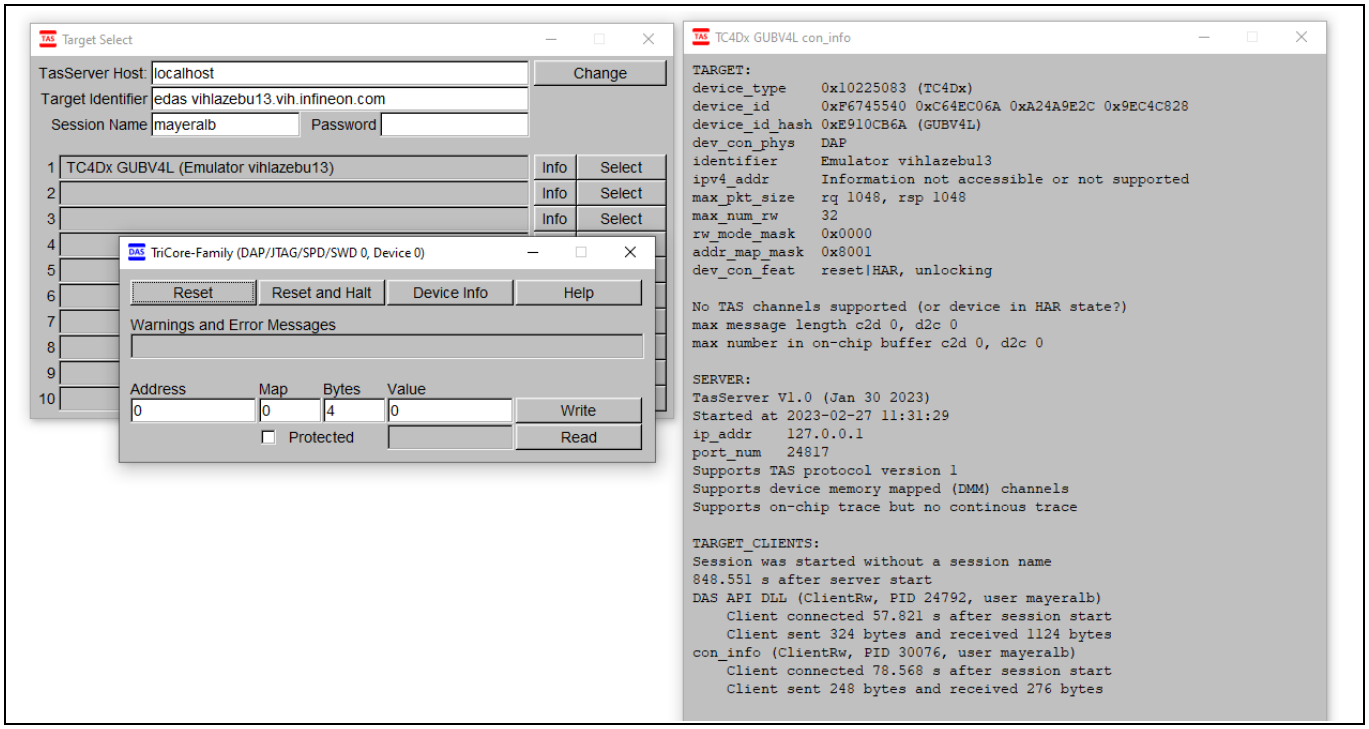


Figure 3 Established emulator access

Steps for establishing emulator access:

1. Open a TAS tool like a TAS Basic Client
2. Enter “edas <emulator IP address>” as target identifier in the target select window. If the port number is not 7745, it must be provided as well after the IP address
   Example: edas vihlazebu11.vih.infineon.com:1234

After a few seconds the connection is established, and the device is listed. At this point the device is visible for any DAS tool.

5 DAS changes for TAS implementation

The following key words from the DAS source code represent features that are not supported or have a different behavior for the TAS based implementation:

Not supported:

- DAE_AR_GET_LAST_ERROR
- DAE_AW_DAP_UART_MODE_xxx
- DAE_AW_DAP_TELEGRAM, DAE_AR_DAP_REPLY
- DAE_AW_DAPCON_PIN_CNTRL
- DAE_ARW_ECF_SERVER_ADDRESS
- DAE_AR_ECF_SOFTWARE_UPDATE_WWW_ADDRESS
- DAS_TS_RW_UNCHANGED

Different behavior:

- cancel_lists() has no effect
- DAS_LC_WILL_NOT_WAIT has no impact on wait_list() behavior if used
- DAS_ERROR_DEVICE_RESET notification:
  - If the reset was before the DAS list execution, it is notified with the list error code and the first transaction, if the address map of this transaction is ‘0’ or ‘1’
  - If it is during the list execution, it will be only notified with the list error code
- TasServer automatically disables Native JTAG Mode when the last client disconnects from this target

Revision history

Document revision	Date	Description of changes
V 1.1	2025-07-07	No functional change, update of template and typo fixing
V 1.0	2023-05-04	Initial release

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2025-07-07

Published by

Infineon Technologies AG

81726 Munich, Germany

© 2025 Infineon Technologies AG. All Rights Reserved.

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

AP32602

Important notice

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.