LAUTERBACH
DEVELOPMENT TOOLS

Integration for Xilinx Vivado

MANUAL

Integration for Xilinx Vivado

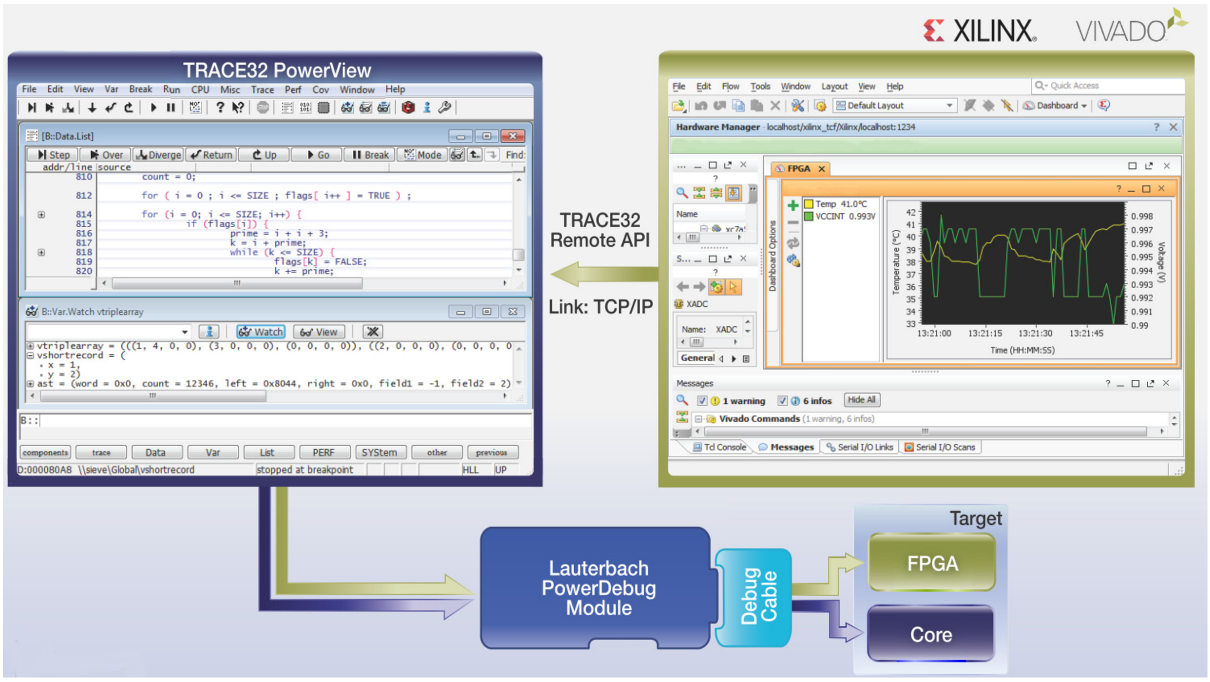
Integration for Xilinx Vivado

Version 06-Jun-2024

Introduction

The Xilinx Virtual Cable Daemon Bridge (XVCD Bridge) is an integration by Lauterbach that allows you to simultaneously use the Vivado Suite of Xilinx for hardware analysis and the Lauterbach TRACE32 infrastructure for software debugging - over a single (shared) connection to the target board via Lauterbach hardware.

Therefore the TRACE32 system is first set up as usual, i.e. the host PC with TRACE32 PowerView is connected to the target via a TRACE32 debugger. Vivado with its 'hw_server' then sets up a connection over the XVCD bridge, which synchronizes with TRACE32 PowerView over the TRACE32 API. This means that Vivado is able to hook its communication into an existing TRACE32 toolchain and use TRACE32 as its backend.



```
addr/line source
        count = 0;

        for ( i = 0 ; i <= SIZE ; flags[ i++ ] = TRUE ) ;

        for (i = 0; i <= SIZE; i++) {
                if (flags[i]) {
                        prime = i + i + 3;
                        k = i + prime;
                        while (k <= SIZE) {
                                flags[k] = FALSE;
                                k += prime;
```

```
vtriplearray = (((1, 4, 0, 0), (3, 0, 0, 0), (0, 0, 0, 0)), ((2, 0, 0, 0), (0, 0, 0, 0
vshortrecord = (
  x = 1,
  y = 2)
ast = (word = 0x0, count = 12346, left = 0x8044, right = 0x0, field1 = -1, field2 = 2)
```

```
D:000080A8  \\sieve\Global\vshortrecord    stopped at breakpoint    HLL  UP
```

Restrictions

As mentioned above, this integration allows simultaneous hardware analysis with Vivado, while performing software debugging with TRACE32. However, it is *not* possible to perform *software* debugging with Vivado at the same time. In order to synchronize the target access of both toolsets, the software debugging capabilities of Vivado need to be disabled (which is done automatically during the start-up process of the XVCD bridge).

System Requirements

TRACE32:

- The tool works with:
  - Windows
  - Linux
- Currently supported target architectures are:
  - MicroBlaze as of TRACE32 release 02/2016
  - ARM as of TRACE32 release 05/2016

Vivado:

- 2015.4 or newer
- It does **not** work with older versions of Vivado, as these come with an older version of the Xilinx hardware server.

Setting up the XVCD Bridge

Setting up the XVCD Bridge is a two-step process:

- **Installation**
- **Configuration**

The setup process is described in detail in the following sections.

Installation

The following paragraphs guide through the installation process. Depending on the operating system, either continue with section **"Windows Installation"** or with section **"Linux Installation"**.

NOTE:	The main directory of the XVCD bridge is ~~/demo/env/xilinx_vivado, where ~~ is your TRACE32 system directory, by default c:\t32 (Windows) or for example /home/t32 (Linux). If this folder is not available in your installation directory, ask Lauterbach support to send you the respective files.

Windows Installation

1. Install the Microsoft Visual C++ Redistributable 2015 from https://www.microsoft.com/en-us/download/details.aspx?id=48145
2. In addition to the files provided by Lauterbach, copy the following files from an existing Vivado installation to the ~~/demo/env/xilinx_vivado folder:
   - hw_server.exe
   - FTD2XX.dll

   Location of the above files in a 64 bit installation:
   - C:\Xilinx\Vivado\2015.4\bin\unwrapped\win64.o\hw_server.exe
   - C:\Xilinx\Vivado\2015.4\lib\win64.o\FTD2XX.dll

   32 bit installation:
   - The path may slightly vary in case of a 32 bit installation.
3. Continue at **"Configuration"**, page 8.

1. In addition to the files provided by Lauterbach, copy the following file from an existing Vivado installation to the ~~/demo/env/xilinx_vivado folder:
   - hw_server
   - **Location of the file**: *<xilinx_installation_directory>*/Vivado/*<version>*/bin/hw_server
2. Continue at **“Configuration”**, page 8.

Configuration

Configuring the environment is a two-step process:

1. Setting up the TRACE32 environment, by choosing *one* of the following two options:
   - **Setup via the T32Start application (only for Windows users)**
   - **Setup via the TRACE32 configuration file (*.t32)**
2. **Settings in the PRACTICE configuration script (*.cmm)**

The configuration process is described in detail in the following sections.

Making the Required Settings in T32Start

TRACE32 needs to be started with the right settings. Using the T32Start application, you can easily configure these settings, and then start TRACE32 from within T32Start.

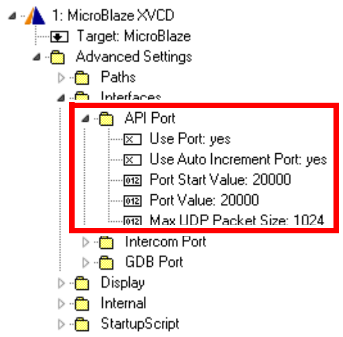
NOTE:	The T32Start application is currently only available for Windows. If you work with Linux or do not want to use T32Start, see the following section about **setting up a configuration file** instead.

For more information about T32Start, refer to **"T32Start"** (app_t32start.pdf).

To make the required settings in T32Start:

1. Navigate to ~~/bin/windows/ and double-click t32start.exe (where ~~ expands to your TRACE32 system directory, by default c:/t32).

2. In the **T32Start** window, navigate to the core for which you want to configure the XVCD bridge.



```
1: MicroBlaze XVCD
  Target: MicroBlaze
  Advanced Settings
    Paths
    Interfaces
      API Port
        Use Port: yes
        Use Auto Increment Port: yes
        Port Start Value: 20000
        Port Value: 20000
        Max UDP Packet Size: 1024
      Intercom Port
      GDB Port
    Display
    Internal
    StartupScript
```

3. Make the following settings:
   - **Advanced Settings** > **Paths** > **SystemPath**: Set the system directory of your TRACE32 installation.
   - **Advanced Settings** > **Interfaces** > **API Port** > **Use Port**: yes
   - **Advanced Settings** > **Interfaces** > **API Port** > **Max UDP Packet Size**: set to at least 1024

Making the Required Settings in the TRACE32 Configuration File

If you work on Linux or you do not want to use T32Start.exe, then you can manually create or edit your TRACE32 configuration file.

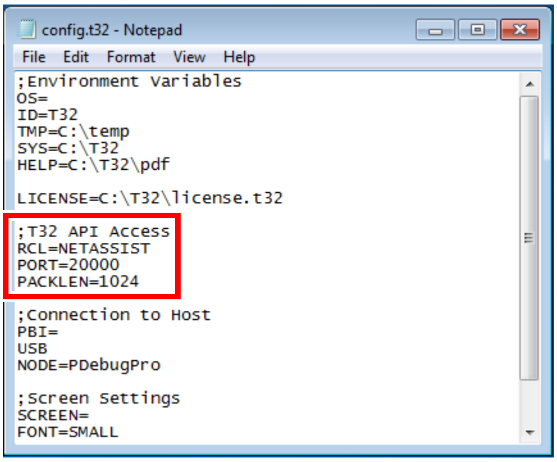
For more information about the TRACE32 configuration file, refer to **"Configuration File"** (training_debugger.pdf).

To make the settings in the configuration file:

1. Open your TRACE32 configuration file (by default it is ~~/config.t32, where ~~ expands to your TRACE32 system directory).

2. Make sure that the file has the following entries:

   - **RCL=NETASSIST**
   - **PORT=***<desired_t32api_port_number>* (default: 20000)
   - **PACKLEN=***<maximum_udp_packet_size>* (set to at least 1024)



```
;Environment Variables
OS=
ID=T32
TMP=C:\temp
SYS=C:\T32
HELP=C:\T32\pdf

LICENSE=C:\T32\license.t32

;T32 API Access
RCL=NETASSIST
PORT=20000
PACKLEN=1024

;Connection to Host
PBI=
USB
NODE=PDebugPro

;Screen Settings
SCREEN=
FONT=SMALL
```

NOTE:	Be aware that the configuration file layout, i.e. the position, sequence of the entries, and the blank lines between the configuration blocks need to correspond to the example given in the screenshot above.

Making the Required Settings in the Hardware Server Configuration File

The XVCD bridge does start and works in interaction with a “Xilinx hardware server” (hw_server). When starting the XVCD bridge, it automatically configures the hardware server with the *hw_server_init.txt* configuration file which can be found in the directory of the XVCD bridge.

Usually the settings in this file do not need to be changed. For details about configuration possibilities, please see the Xilinx documentation, such as the “Vivado Design Suite User Guide”.

Checking the Settings in the PRACTICE Configuration Script

Before starting the bridge, we need to make sure that the settings in the PRACTICE script xvcd_config.cmm are correct. These settings define certain port values for bridge communication. In most cases, these values do *not* need to be changed.

To check the settings in the script:

1. Navigate to the script file ~~/demo/env/xilinx_vivado/xvcd_config.cmm.
2. In the script file, check the settings described in the table below.

XVCDPORT	Port between hw_server and XVCD bridge.
TERMINATEPORT	Port that is used to terminate the bridge from TRACE32 PowerView.
PARKSTATE	The TAP parkstate of the XVCD bridge. Default: **7.** The parameter **7.** stands for **Select-DR-Scan**. The TAP parkstate of TRACE32 needs to match the TAP parkstate of the XVCD bridge. • The current TAP parkstate of TRACE32 can be looked up in the **SYStem.CONFIG.state /Jtag** window. • In order to change the TAP parkstate in TRACE32 to **Select-DR-Scan**, use **SYStem.CONFIG.TAPState 7.** in your start-up script (*.cmm). • For MicroBlaze and most ARM systems the **Select-DR-Scan** state is advisable.
STARTHWSERVER	Boolean value that determines whether the XVCD bridge should start the hw_server automatically or not. If set to **TRUE()**, the bridge will start the executable that is located in the xilinx_vivado folder with the according parameters. If set to **FALSE()**, then you should start the hw_server manually, as soon as the XVCD terminal shows an according message. For more information see chapter **Starting the Bridge with Manual Start of hw_server**.

Starting the XVCD Bridge

By default, the XVCD bridge **automatically** starts the hw_server executable during the startup process. However, there are scenarios during which a **manual** start of the hw_server may be necessary, such as:

- Wanting to start the hw_server with custom parameters
- When having problems during the startup process

Depending on which method you prefer, please continue with *one* of the following chapters:

- **Starting the Bridge with Automatic Start of hw_server** (default)
- **Starting the Bridge with Manual Start of hw_server**

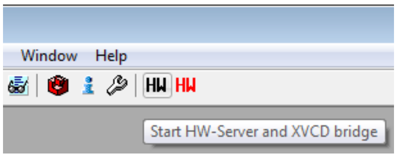
Starting the Bridge with Automatic Start of hw_server

To start the XVCD bridge:

1. Open the script xvcd_config.cmm and make sure that the parameter **STARTHWSERVER** is set to **TRUE()**.
2. Start TRACE32.
3. Add buttons for starting and terminating the XVCD bridge to the TRACE32 main toolbar by typing at the TRACE32 command line:

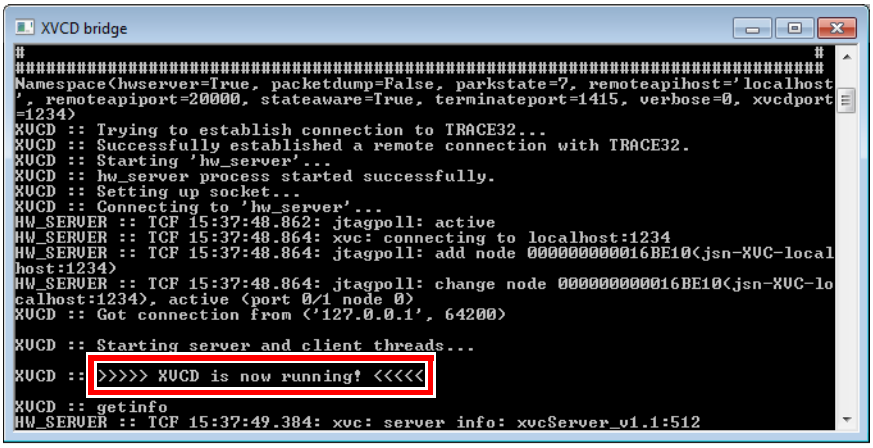
```
DO ~~/demo/env/xilinx_vivado/xvcd_add_buttons.cmm
```

Two buttons labeled **HW** are added to the TRACE32 main toolbar.



4. Click the black **HW** button to start the HW server and the XVCD bridge.

   The **XVCD bridge** terminal window opens, displaying the current status of the bridge. As soon as the **XVCD bridge** terminal window displays “XVCD is now running”, you can connect to Vivado with the steps described below.



```
#                                                                              #
################################################################################
Namespace(hwserver=True, packetdump=False, parkstate=7, remoteapihost='localhost
', remoteapiport=20000, stateaware=True, terminateport=1415, verbose=0, xvcdport
=1234)
XVCD :: Trying to establish connection to TRACE32...
XVCD :: Successfully established a remote connection with TRACE32.
XVCD :: Starting 'hw_server'...
XVCD :: hw_server process started successfully.
XVCD :: Setting up socket...
XVCD :: Connecting to 'hw_server'...
HW_SERVER :: TCF 15:37:48.862: jtagpoll: active
HW_SERVER :: TCF 15:37:48.864: xvc: connecting to localhost:1234
HW_SERVER :: TCF 15:37:48.864: jtagpoll: add node 000000000016BE10(jsn-XVC-local
host:1234)
HW_SERVER :: TCF 15:37:48.864: jtagpoll: change node 000000000016BE10(jsn-XVC-lo
calhost:1234), active (port 0/1 node 0)
XVCD :: Got connection from ('127.0.0.1', 64200)

XVCD :: Starting server and client threads...

XVCD :: >>>>> XVCD is now running! <<<<<

XVCD :: getinfo
HW_SERVER :: TCF 15:37:49.384: xvc: server info: xvcServer_v1.1:512
```

5. Proceed with chapter **Connecting to Vivado**.

Recommendation

We recommend that you call the xvcd_add_buttons.cmm script from your own start-up script. This will automatically include the buttons on the TRACE32 main toolbar on every start-up.

Example:

```
;code of your own start-up script (*.cmm)

DO ~~/demo/env/xilinx_vivado/xvcd_add_buttons.cmm

;your code
```

Variation

Alternatively, it is possible to open and close the bridge with the PRACTICE scripts xvcd_start.cmm and xvcd_end.cmm. They call the executable files xvcd_bridge.exe and xvcd_close.exe with the appropriate parameters. Do not call the executables directly, without passing any parameters.

The xvcd_start.cmm and xvcd_end.cmm scripts can also be integrated into your own PRACTICE start-up scripts, as shown in the example above.

Starting the Bridge with Manual Start of hw_server

By default, the XVCD bridge automatically starts the hw_server process during the startup process. However, there are scenarios during which a *manual start* of the hw_server may be necessary, such as:

- Wanting to start the hw_server with custom parameters
- When having problems during the startup process

The following steps will allow a manual start of the hw_server:

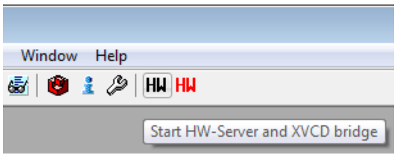
1. Open the script xvcd_config.cmm and set the parameter **STARTHWSERVER** to **FALSE()**.

2. Start TRACE32.

3. Add buttons for starting and terminating the XVCD bridge to the TRACE32 main toolbar by typing at the TRACE32 command line:

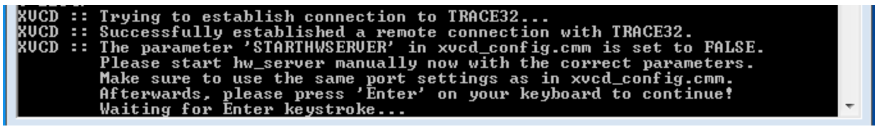
```
DO ~~/demo/env/xilinx_vivado/xvcd_add_buttons.cmm
```

   Two buttons labeled **HW** are added to the TRACE32 main toolbar.



4. Click the black **HW** button to start the XVCD bridge.

5. The **XVCD bridge** terminal window opens, displaying the current status of the bridge. The window output should stop with an entry "Waiting for Enter keystroke..."



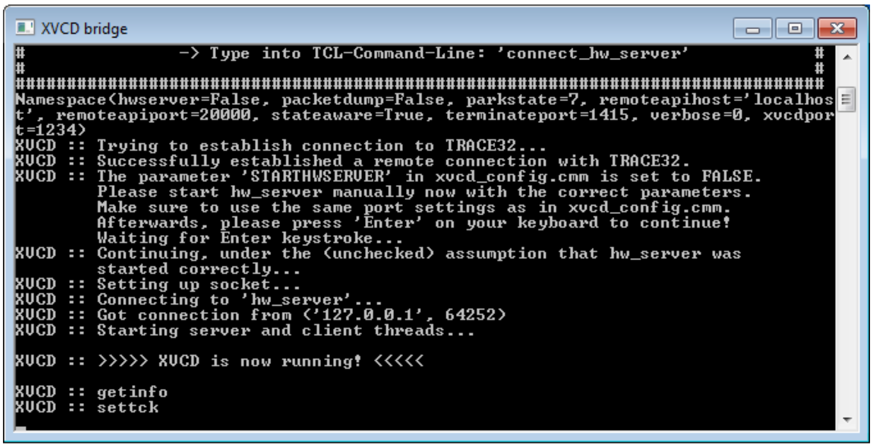
```
XVCD :: Trying to establish connection to TRACE32...
XVCD :: Successfully established a remote connection with TRACE32.
XVCD :: The parameter 'STARTHWSERVER' in xvcd_config.cmm is set to FALSE.
        Please start hw_server manually now with the correct parameters.
        Make sure to use the same port settings as in xvcd_config.cmm.
        Afterwards, please press 'Enter' on your keyboard to continue!
        Waiting for Enter keystroke...
```

6. Open a separate terminal window. Start the hw_server executable in this terminal, along with the desired parameters. The default parameters that are used by the XVCD bridge are:

```
hw_server.exe -L- -ljtag2
-e "set auto-open-servers xilinx-xvc:localhost:1234"
-e "set always-open-jtag 1"
-e "set xvc-log-level 1"
```

7. After you have started the hw_server, go back to the terminal of the XVCD bridge and press 'Enter'.

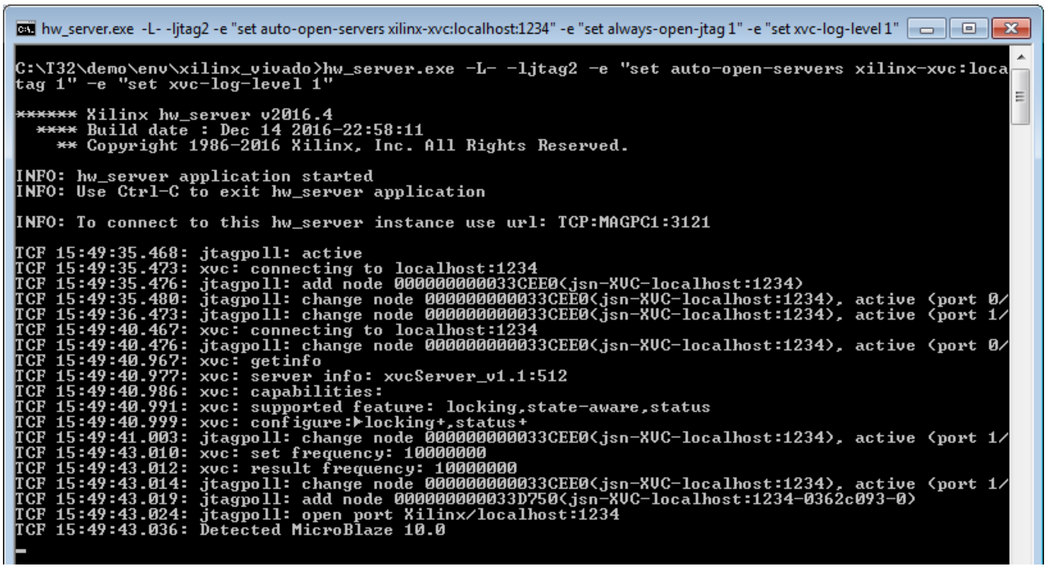
8. The XVCD bridge should continue and eventually finish its startup procedure. The output of the XVCD bridge should look like the following:



XVCD bridge

```
#              -> Type into TCL-Command-Line: 'connect_hw_server'              #
#                                                                              #
################################################################################
Namespace(hwserver=False, packetdump=False, parkstate=7, remoteapihost='localhos
t', remoteapiport=20000, stateaware=True, terminateport=1415, verbose=0, xvcdpor
t=1234)
XVCD :: Trying to establish connection to TRACE32...
XVCD :: Successfully established a remote connection with TRACE32.
XVCD :: The parameter 'STARTHWSERVER' in xvcd_config.cmm is set to FALSE.
        Please start hw_server manually now with the correct parameters.
        Make sure to use the same port settings as in xvcd_config.cmm.
        Afterwards, please press 'Enter' on your keyboard to continue!
        Waiting for Enter keystroke...
XVCD :: Continuing, under the (unchecked) assumption that hw_server was
        started correctly...
XVCD :: Setting up socket...
XVCD :: Connecting to 'hw_server'...
XVCD :: Got connection from ('127.0.0.1', 64252)
XVCD :: Starting server and client threads...

XVCD :: >>>>> XVCD is now running! <<<<<

XVCD :: getinfo
XVCD :: settck
_
```

The hw_server output should be similar to this:



hw_server.exe -L- -ljtag2 -e "set auto-open-servers xilinx-xvc:localhost:1234" -e "set always-open-jtag 1" -e "set xvc-log-level 1"

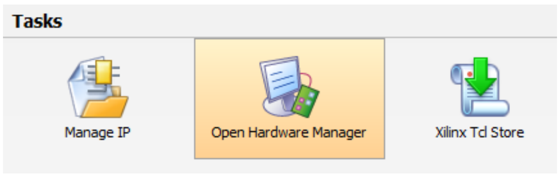
```
C:\T32\demo\env\xilinx_vivado>hw_server.exe -L- -ljtag2 -e "set auto-open-servers xilinx-xvc:loca
tag 1" -e "set xvc-log-level 1"

****** Xilinx hw_server v2016.4
  **** Build date : Dec 14 2016-22:58:11
    ** Copyright 1986-2016 Xilinx, Inc. All Rights Reserved.

INFO: hw_server application started
INFO: Use Ctrl-C to exit hw_server application

INFO: To connect to this hw_server instance use url: TCP:MAGPC1:3121

TCF 15:49:35.468: jtagpoll: active
TCF 15:49:35.473: xvc: connecting to localhost:1234
TCF 15:49:35.476: jtagpoll: add node 000000000033CEE0(jsn-XVC-localhost:1234)
TCF 15:49:35.480: jtagpoll: change node 000000000033CEE0(jsn-XVC-localhost:1234), active (port 0/
TCF 15:49:36.473: jtagpoll: change node 000000000033CEE0(jsn-XVC-localhost:1234), active (port 1/
TCF 15:49:40.467: xvc: connecting to localhost:1234
TCF 15:49:40.476: jtagpoll: change node 000000000033CEE0(jsn-XVC-localhost:1234), active (port 0/
TCF 15:49:40.967: xvc: getinfo
TCF 15:49:40.977: xvc: server info: xvcServer_v1.1:512
TCF 15:49:40.986: xvc: capabilities:
TCF 15:49:40.991: xvc: supported feature: locking,state-aware,status
TCF 15:49:40.999: xvc: configure:►locking+,status+
TCF 15:49:41.003: jtagpoll: change node 000000000033CEE0(jsn-XVC-localhost:1234), active (port 1/
TCF 15:49:43.010: xvc: set frequency: 10000000
TCF 15:49:43.012: xvc: result frequency: 10000000
TCF 15:49:43.014: jtagpoll: change node 000000000033CEE0(jsn-XVC-localhost:1234), active (port 1/
TCF 15:49:43.019: jtagpoll: add node 000000000033D750(jsn-XVC-localhost:1234-0362c093-0)
TCF 15:49:43.024: jtagpoll: open port Xilinx/localhost:1234
TCF 15:49:43.036: Detected MicroBlaze 10.0
_
```

9. The connection from the bridge to the hw_server has now successfully been established. Proceed with chapter **Connecting to Vivado**.

Connecting to Vivado

NOTE:	The Vivado version must match the version of hw_server(.exe) that you copied during the installation process.
NOTE:	Make sure to disconnect any "Xilinx JTAG USB cable" that is connected to the host PC.

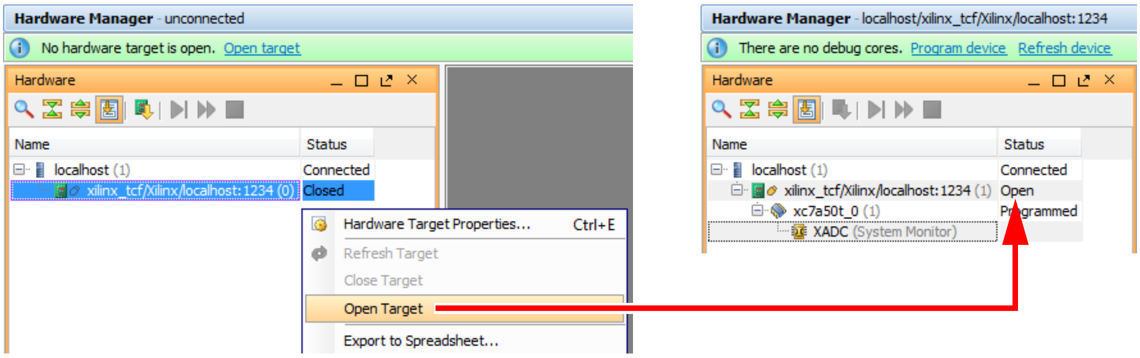
1. Start Vivado and click **Open Hardware Manager**.



2. At the Vivado TCL command line type: `connect_hw_server`

   The **Hardware Manager** should show a new entry in the **Hardware** window pane, which states a new connection with the name “localhost” and the status “connected”.

   This entry should also have a sub-entry, which represents the target (e.g. “xilinx_tcf/Xilinx/localhost:1234”, with the status “closed”).



Hardware Manager - unconnected

No hardware target is open. Open target

Name	Status
localhost (1)	Connected
xilinx_tcf/Xilinx/localhost:1234 (0)	Closed

Hardware Manager - localhost/xilinx_tcf/Xilinx/localhost:1234

There are no debug cores. Program device Refresh device

Name	Status
localhost (1)	Connected
xilinx_tcf/Xilinx/localhost:1234 (1)	Open
xc7a50t_0 (1)	Programmed
XADC (System Monitor)	

3. Connect to the target by right-clicking this sub-entry, and then select **Open Target** from the popup menu. Its status should change to “Open”, and show sub-entries for the JTAG components.

   You are now ready to use Vivado and TRACE32 simultaneously.

Troubleshooting

The following describes some possible error scenarios, along with suggestions how to resolve them:

‘VREF missing’ error

- When selecting "open target" in Vivado, the Xilinx hardware server ("HW_SERVER ::" prefix in the XVCD bridge window) throws the following error:

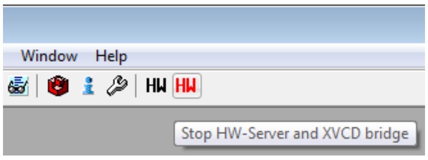
```
HW_SERVER :: TCF xx:xx jtagpoll: cannot open port xx:
open device failed: vref missing
```

- **Possible solutions**:
  - One possible cause is that there is something wrong with the Lauterbach JTAG debug cable. Please check the cable to make sure it is properly connected. If normal debugging of the target via TRACE32 (without the XVCD bridge) is possible, then the cable should be okay.
  - Another cause can be a "Xilinx JTAG USB cable" that is connected to the host PC. Disconnecting the cable should solve the issue.

Closing the XVCD Bridge

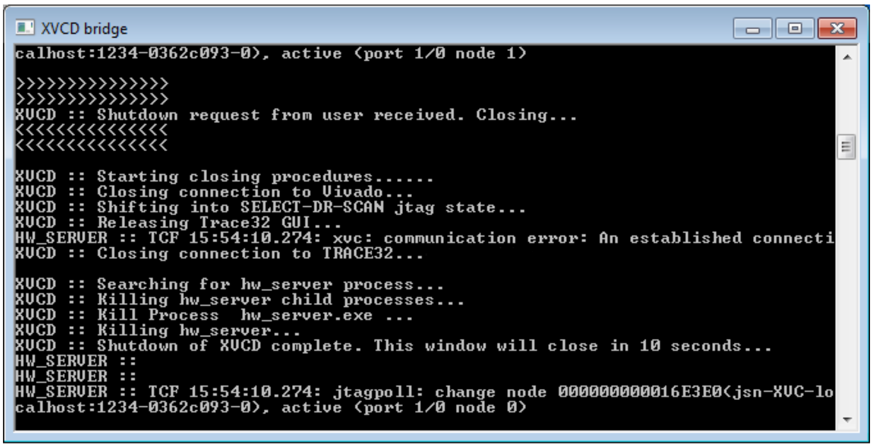
NOTE:	Do NOT close the **XVCD bridge** terminal window directly by clicking the **X** in the top right corner. Otherwise the shutdown procedure cannot be passed, which may result in unpredictable errors.

- **To close the XVCD bridge, do one of the following:**
  - On the TRACE32 main toolbar, click the red **HW** button.



  - Press **Ctrl+C** in the **XVCD bridge** terminal window.

The XVCD bridge now initiates a shutdown procedure. **XVCD bridge** terminal window closes 10 seconds after the shutdown procedure is complete.



```
calhost:1234-0362c093-0), active (port 1/0 node 1)
>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>
XVCD :: Shutdown request from user received. Closing...
<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<

XVCD :: Starting closing procedures......
XVCD :: Closing connection to Vivado...
XVCD :: Shifting into SELECT-DR-SCAN jtag state...
XVCD :: Releasing Trace32 GUI...
HW_SERVER :: TCF 15:54:10.274: xvc: communication error: An established connecti
XVCD :: Closing connection to TRACE32...

XVCD :: Searching for hw_server process...
XVCD :: Killing hw_server child processes...
XVCD :: Kill Process  hw_server.exe ...
XVCD :: Killing hw_server...
XVCD :: Shutdown of XVCD complete. This window will close in 10 seconds...
HW_SERVER ::
HW_SERVER ::
HW_SERVER :: TCF 15:54:10.274: jtagpoll: change node 000000000016E3E0(jsn-XVC-lo
calhost:1234-0362c093-0), active (port 1/0 node 0)
```

Error Behavior

If an error occurs during operation, the XVCD bridge automatically initiates a shutdown procedure. Possible errors are target power off, disconnected target, timeout of TRACE32 PowerView API, or Vivado timeout.

During the automatic shutdown, the **XVCD bridge** terminal window displays a message which helps you identify the error.

The automatic shutdown tries to ensure that TRACE32 can still work properly after the error. This, however, depends on the nature of the error.