Faculty of Science
Department of Computer Science
Chair for Embedded Systems
Prof. Dr. O. Bringmann

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Digital Design and Synthesis of Embedded Systems

Summer Term 2025

Exercise 1 : Deadline 11.05.2025 - 23:59 o'clock

The solutions are submitted via ILIAS as a packed archive (ZIP/TAR.GZ). This ZIP contains the written answers to questions, the source code (not as a listing in the PDF), the simulation results as VCD files, and the Makefile. Each of your programming tasks **must** be executable with your Makefile or with a standard python interpreter. If your python solution uses non-standard packages you have to provide a requiretmens.txt file. If you do not follow this format, your handing may be graded with 0 points!

Exercise 1 Design methods of embedded systems [10 Points]

Please answer the following questions briefly.

(a) Why are there different levels of abstraction and how does the transformation between the levels take place?

(b) Explain the difference between the behavioral and structural view in the Y diagram. Use an made-up example for your explanation.

(c) What is an Intellectual Property (IP)? What types of IPs are there and how do they differ? Which chip development method makes frequent use of IPs?

(d) What is the difference between an FPGA and an ASIC implementation with standard cells?

Exercise 2 Simulation of 1-bit adders with Verilog and Xcelium [10 Points]

The following tasks require the execution of Cadence Xcelium. Please work your way carefully through the appendix of this exercise sheet, where both login onto the exercise computers and the simulation of circuits are briefly explained.

(a) Given the following SystemVerilog code for a one-bit halfadder at logic level. Find and correct the errors!

```
1  module halfadder (
2    input logic a, b,
3    output logic s, c,
4  );
5
6    assign s = a | b;
7    assign c = a & b;
8
9  endmodule
10
```

(b) Simulate the corrected one-bit halfadder completely via the command line of Xcelium with the commands: `force <signal> <value>` und `run 10`. Pay attention to the correct input format of the `force`-command (see example below). Save your code in a file `halfadder.sv`. Submit your simulation results as a VCD (Value Change Dump) file. Instructions for generating value change dumps can be found in the appendix of this exercise sheet.

Expected submission: vcd file

(c) Create the structural description of a one-bit full adder in SystemVerilog using the one-bit halfadder from task 2(b). **Expected submission:** sv file

(d) Simulate the structure description of the one-bit full adder completely via the command line of Xcelium as in task 2(b). Submit your simulation results as a VCD (Value Change Dump) file.

Expected submission: vcd file

(e) Create the behavioural description of a one-bit full adder in SystemVerilog.

Expected submission: sv file

(f) Simulate the behavioural description of the one-bit full adder completely and manually via the command line of Xcelium as in task 2(b). Submit your simulation results as a VCD (Value Change Dump) file.

Expected submission: VCD-Datei

(g) Are the simulation results of the structural description and the behavioural description identical? Explain what differences/similarities you see?

(h) Create a file `and.sv` and a SystemVerilog module that describes the behaviour of a logical AND function. Create another file `mult.sv` and develop a structural description of a 3-bit multiplier. Use the following code as the basis of the multiplier module:

```
1      module mult (
2        input  logic a0, a1, a2, b0, b1, b2,
3        output logic s0, s1, s2, s3, s4, s5
4    );
5
```

(i) Simulate your design as before and check for correctness. **Expected submission:** VCD-Datei

General information for your submission

Each submission that with programming taks must adhere to the following layout:

```
├──folder_which_will_be_zipped
        ├── Submission.pdf
        ├── Programming_task_0 (if available)
                ├── Makefile
                ├── Source file 0
                ├── Source file-1
                ├── Source file-n
       ├── Programming_task_1 (if available)
                ├── Makefile
                ├── Source file 0
                ├── Source file-1
                ├── Source file-n
```

Note that coding-submissions that are not executable with your provided Makefile **will not be graded and result in 0 point for this exercise!**

Use Xcelium on the exercise computers

In order to start the tools required for the exercises, you need to log in to our network and use one of the pool computers listed below:

- samba
- goten
- ducky
- dacky
- veku
- yabara
- cvraman
- mambo
- bolero
- bump
- habanera
- veritas
- goku
- vegeta

All tool installations are located on these computers in the NFS network. The license server for all tools runs on the computer `menelaos.cs.uni-tuebingen.de` and the port `1701`. You should be able work on each exercise without making changes to exercise-setup. If you encounter any issue, feel free to use the forum in the Ilias course ASAP.
Below we present solutions for the most common operating systems, how you can log in to one of the pool computers and how you can run Xcelium with the help of the pool computers. All solutions work under the three major operating systems Windows, Mac OS X and Linux.

Apache Guacamole

To use a pool PC with Apache Guacamole, you do **not** need to install additional software, as you only require a web browser. If something does not work, you can of course contact us at any time - preferably via the forum. As a first step, however, it makes sense to take a look at the corresponding FAQ.
The following steps are necessary for connection and use:

1. Visit the Guacamole-Instance of the Chair of Embedded Systems.
2. Log in with your CS account.
3. Use your smartphone to configure 2-factor authentication, which is mandatory for use. To do this, you need an OTP app on your smartphone. Below you will find a list of apps for Android and iOS. Please note that these are only examples and we do not make any recommendations:
   - andOTP (PlayStore)
   - Google Authenticator (PlayStore)
   - FreeOTP Authenticator (App Store)
   - Google Authenticator (App Store)
4. You should now see the list of available pool PCs (as shown in figure 1). Note: The PCs available do not have to be exactly the same as in the illustration.

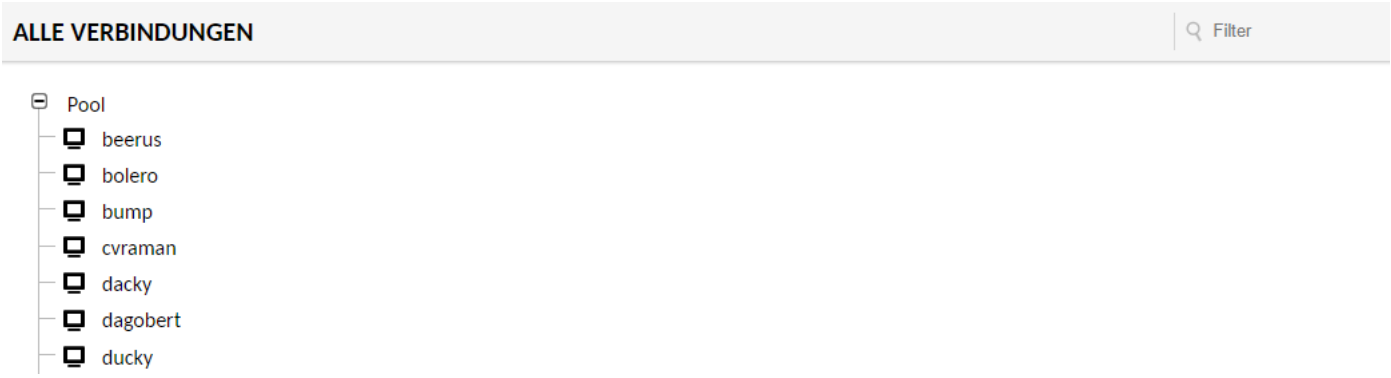


Figure 1: List of Pool-PCs in Guacamole

5. Select one of the computers. Selection causes a connection to be established to the computer. The pool PCs are switched off overnight, so there may be a delay of up to 2 minutes during the first connection. During this time, Guacamole will report the error that the host is not available. However, a connection will be established automatically as soon as it is available.

Note:

- If Guacamole is not able to establish a connection, you should try to wake up the computer manually in the next step. How to do this is described in section General information on the use of SSH. If it does not work via Guacamole, please use the instructions for the connection via SSH.
- If the login via Guacamole works, but you only see a black screen, you must log in via SSH on the corresponding computer (see section General information on the use of SSH and following) and execute the following commands:

```
1 # kinit
2 # aklog
3
```

  You must then start a new session on the corresponding computer via Guacamole.

General information on the use of SSH

- If you receive an error message similar to *No route to host* or *Remote session unexpectedly closed*, then the corresponding computer is still off and must first be switched on. You can do this from the SSH gateway as follows:

```
# sudo ether-wake mac-address

```

  You can find an assignment of pool-computer to MAC address here. To boot *samba* the following command is required:

```
# sudo ether-wake 90:1b:0e:c6:eb:6c

```

- The error message

```
Could not chdir to home directory /home/<cs-account>: Permission denied
```

  is usually not a problem. You should simply be able to use the current session. To access your home directory, you must then execute the command `cd` once in the console. Otherwise, you can simply end this session and start a new one.

- If you receive a message such as `TERM undefined` when logging on to the pool-computer, you must log in again to **both** computers. In other words, on the gateway and on the computer.

- You can upload the required materials to the pool-computer either via `scp` or MobaXTerm.

Starting an SSH connection under Linux and Mac OS X

This section covers starting Xcelium from a Linux computer. You will need a graphical installation with X-Server and OpenSSH.

1. Connecting to the SSH gateway
In the first step, you must connect to the SSH gateway – which you can reach at the address `sshgw.cs.uni-tuebingen.de`
Enter the following into a Linux console of your choice (e.g. xterm, konsole, gterminal):

```
# ssh -X username@sshgw.cs.uni-tuebingen.de
```

Then enter your WSI password (which you initially set via the application). You should now see a Screen like the following.

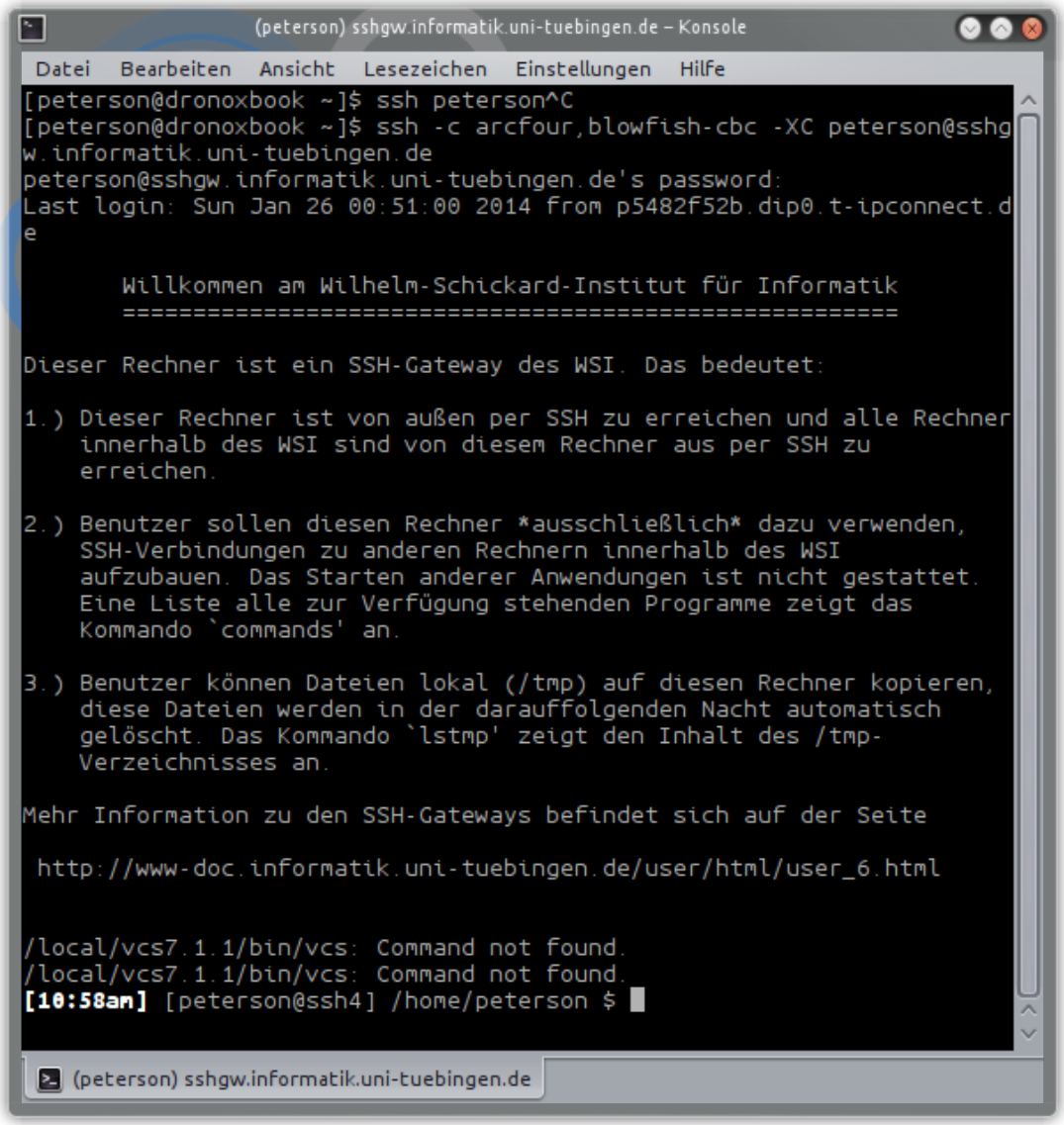


Figure 2: Screen after successful connection to the SSH gateway

The option `-X` activates the X-forwarding mechanism. Please replace `username` with your WSI account!

2. Connect to the pool-computer

In the second step, select a computer from the list above (e.g. samba) and log in to this computer from the SSH gateway. To do this, enter the following in the console:

```
# ssh -X username@poolPC
```

The field `username` still corresponds to your WSI account and `poolPC` to the corresponding host name of the pool computer. An example of these fields would be the following:

```
ssh -X peterson@samba
```

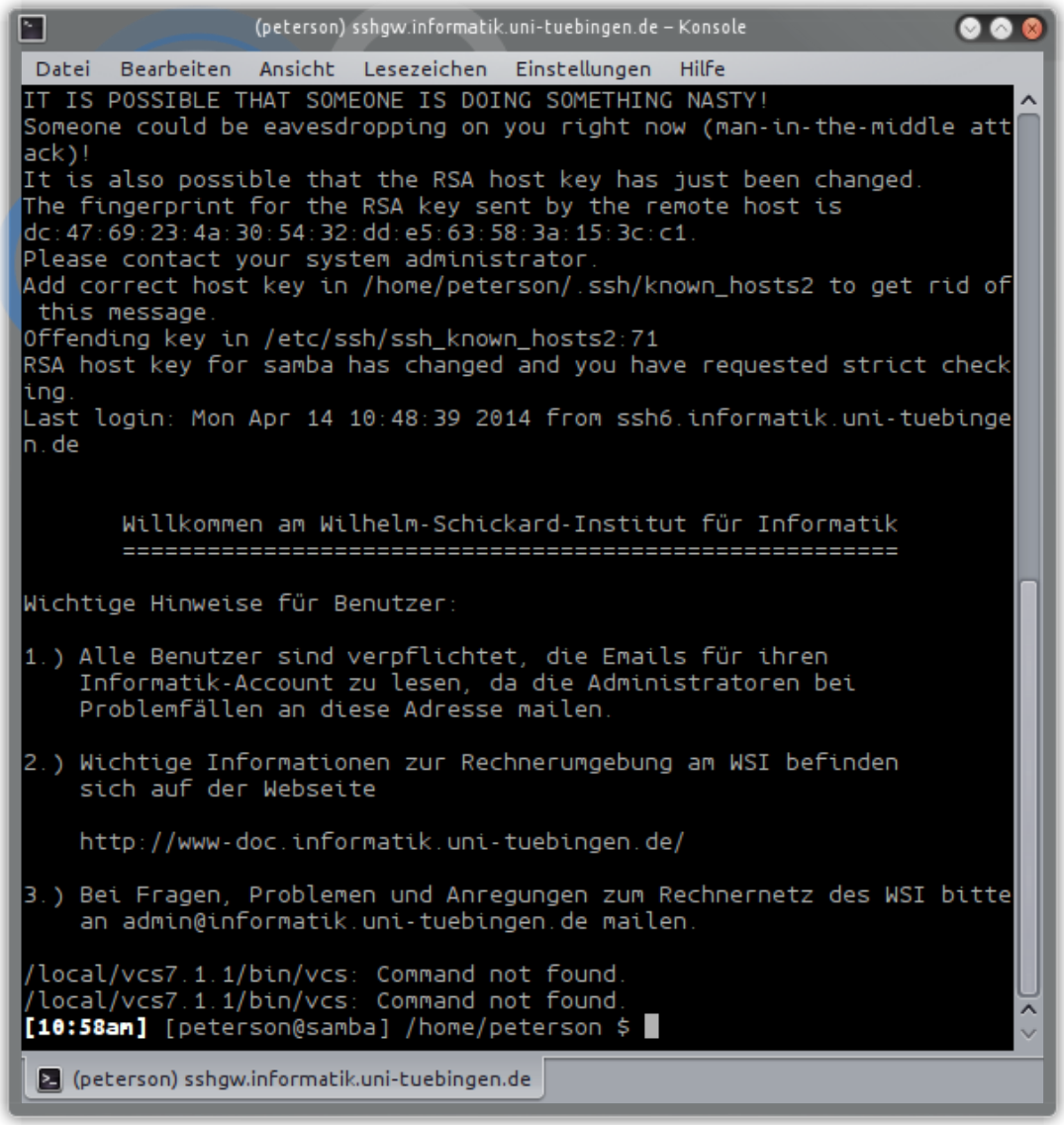


Figure 3: Screen after successful connection to a computer (here: samba)

Starting an SSH connection under Windows

This section covers how to start Xcelium from a Windows computer. You will need an X11 client for this – in the following steps we will use MobaXTerm Home Edition. **Note:** The screenshots still refer to the old version – please note only the text for the console entries, not the screenshots.

1. Installation and start of MobaXTerm

Download the Home Edition of MobaXTerm free of charge from the link above and run the installer. Then start MobaXTerm. You should see the following window:

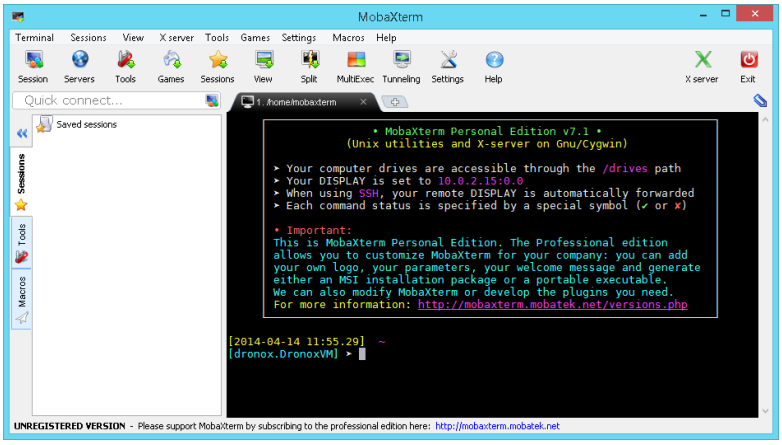


Figure 4: MobaXTerm after start

2. Configuration of the SSH connection

Now set up the SSH connection to the pool computer. To do this, click on `Session` at the top left. The following window opens:

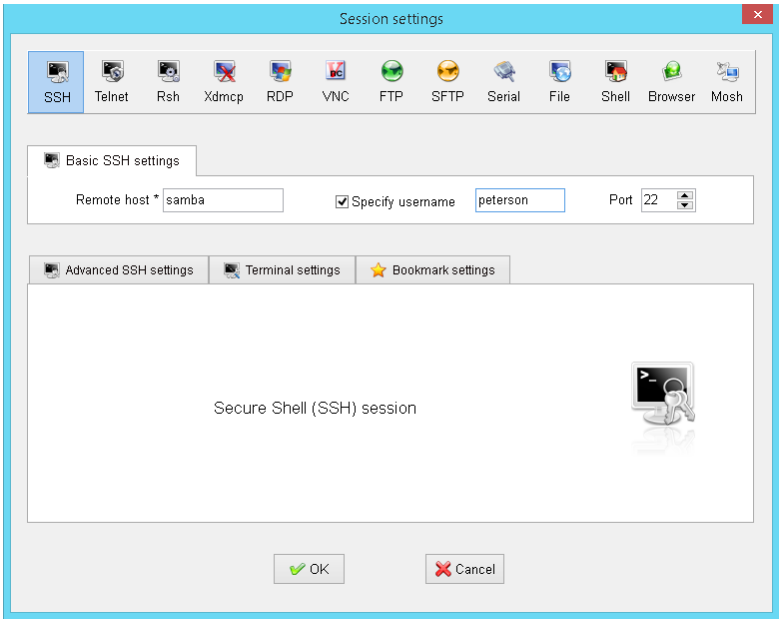


Figure 5: MobaXTerm: SSH-connection configuration (1)

Enter the name of the pool-computer from the above list of computer (e.g. samba) in `Remote host`. Also complete your WSI account at `Specify username`.
Then open the advanced settings by clicking on `Advanced SSH settings`. The following dialog opens:

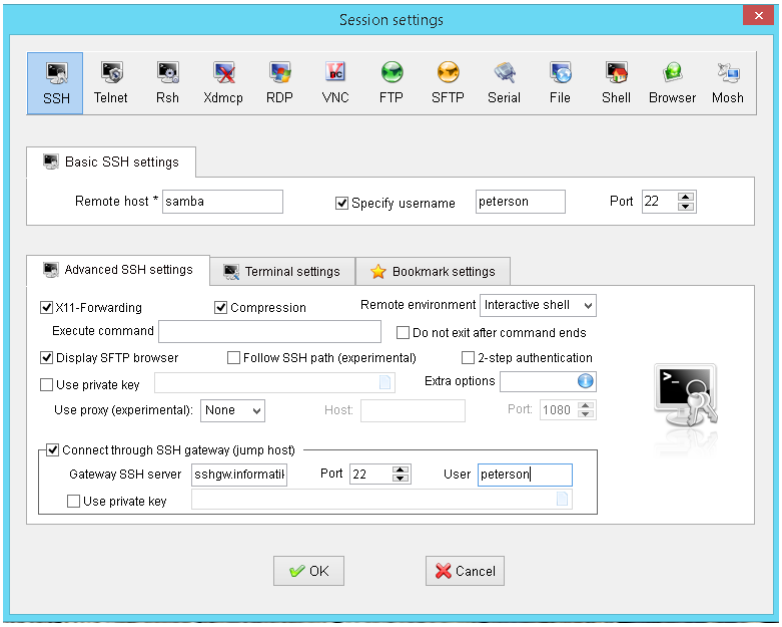


Figure 6: MobaXTerm: SSH-connection configuration (2)

Make sure that `X11-Forwarding` and `Compression` are checked. Also activate the option `Connect through SSH gateway (jump host)`. Enter the SSH gateway (sshgw.cs.uni-tuebingen.de) in `Gateway SSH server` and your WSI account in `User`.

Note: In newer MobaXTerm versions, the configuration options for the SSH gateway have been moved to a new tab (Network options).

3. Starting the SSH connection

Then confirm the SSH session by clicking on `OK`.
Enter your WSI password in the password dialog that opens.

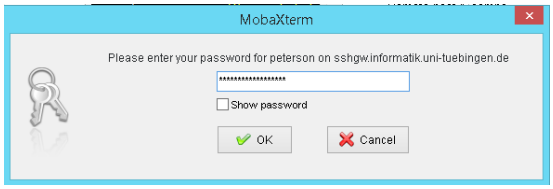


Figure 7: MobaXTerm: Start SSH-connection (1)

Confirm the dialogs until the following window opens and the computer samba asks you for the password again.

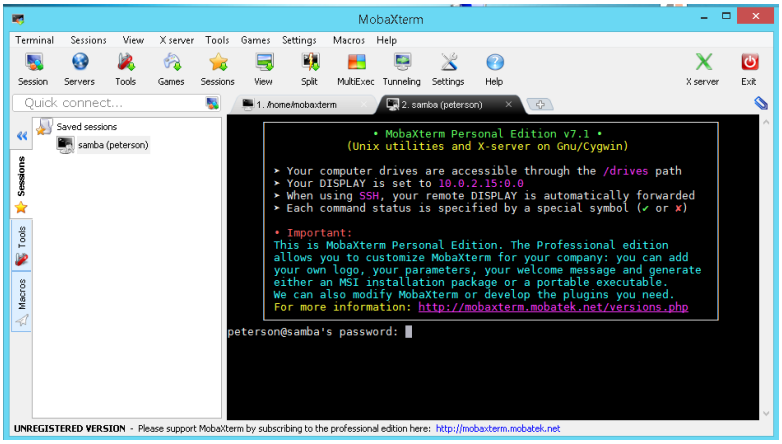


Figure 8: MobaXTerm: Start SSH-connection (2)

You should now be logged in.

Simulation with Xcelium

In the following, you find a brief step-by-step guide for an interactive and automated simulation of a halfadder.

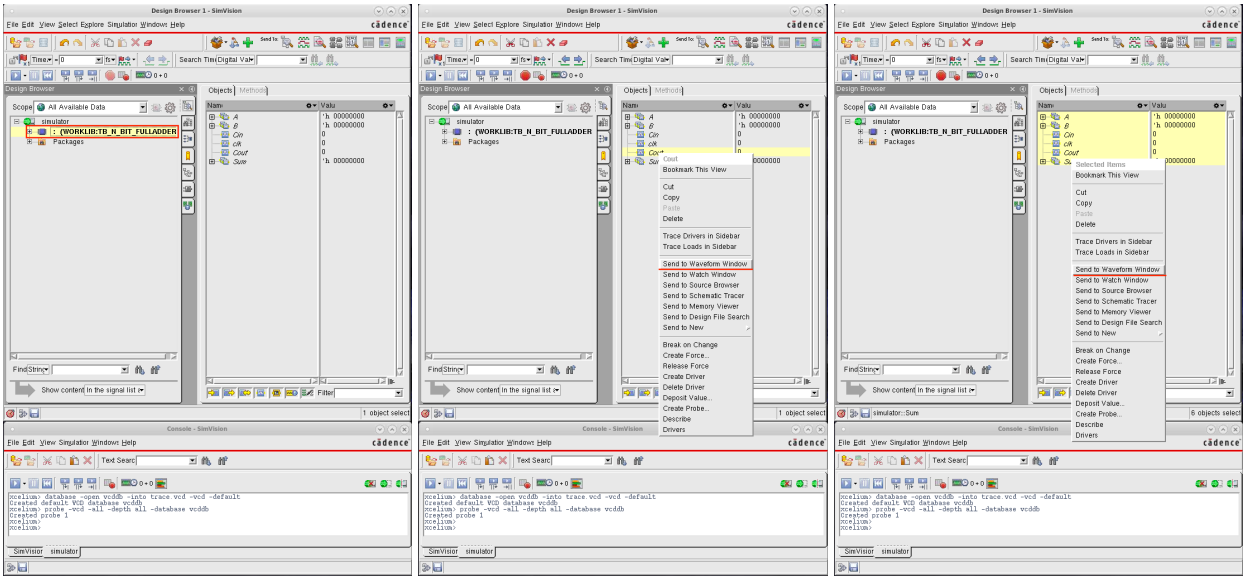
1. Connect to a pool computer.
2. In the terminal, navigate to the folder in which the Makefile and, in our case, the SystemVerilog files are located.
3. Adjust the Makefile in the appropriate places. The SystemVerilog source files and simulation and synthesis variables, such as the top module and name of the waveform trace file, are set here. *Note:* Exercise that have no testbench, where you toggle the inputs manually only require the `HDL_SRC` and `HDL_TB_TOP` variable on RTL.
4. Familiarize yourself with the Makefile.
5. Carry out the following steps for the automated simulation in the bash:

```
make clean  # Deletes all files created by the Makefile.
make sim    # Sets the necessary environment variables and executes the
            # Simulation with the help of a testbench/test driver.
```

6. Carry out the following steps for the interactive simulation:
   (a)

```
make clean     # Deletes all files created by the Makefile.
make sim_gui   # Sets the necessary environment variables
               # and starts the Xcelium simulation environment.
```

   (b) Add the signals to the waveform viewer. To do this, select the desired module in the Design Browser (see figure 9a). With a **right-click**, you can select individual or all signals under the *Objects* tab and add them to the waveform viewer using **"Send to Waveform Window"** (see figure 9b, 9c).



(a) Module selection (b) Signal selection (c) Signal selection

Figure 9: Waveform-Viewer

   (c) A new window "Waveform 1 - SimVision" should now have opened (see figure 10) and the corresponding signals should be listed.

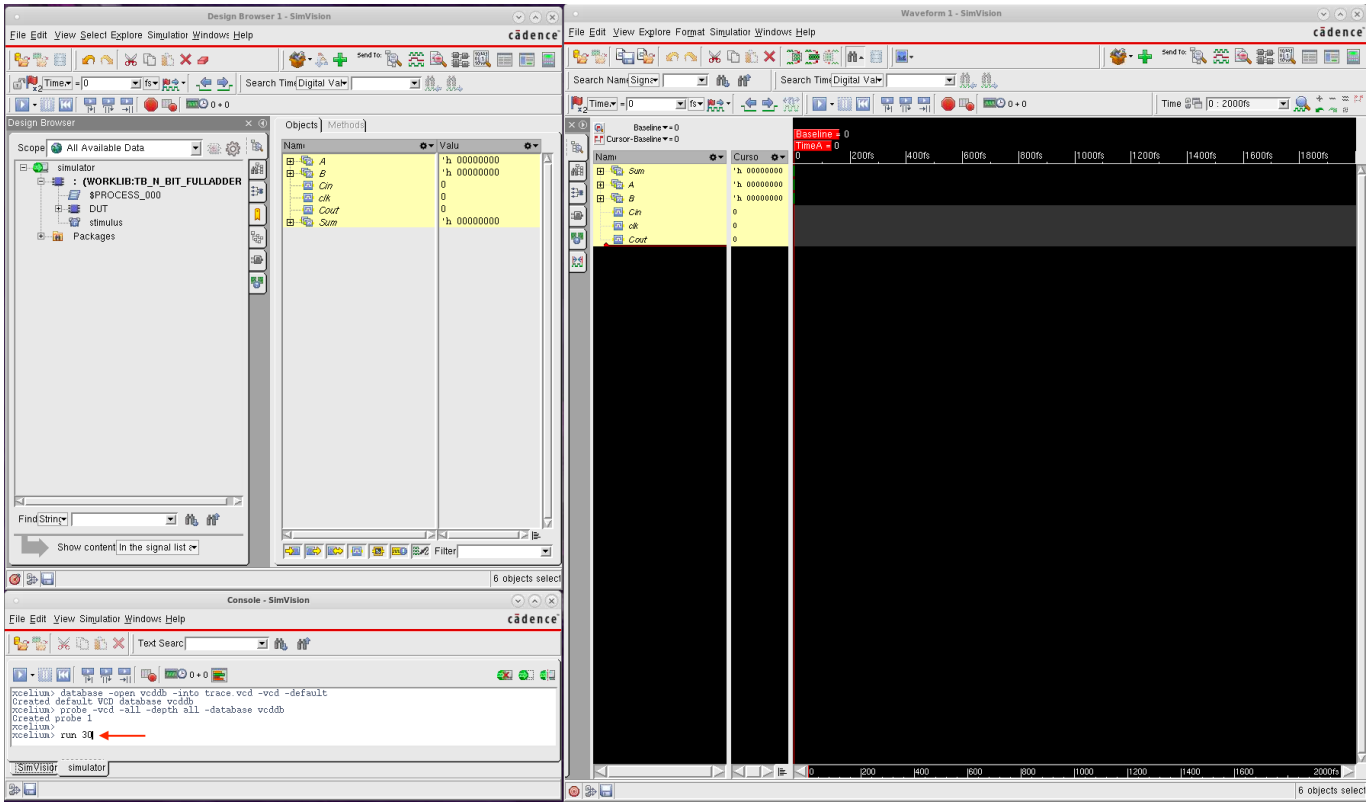


Figure 10: Waveform-Viewer

(d) Now enter **“run 30”** to run the testbench for 30 time units. You should see a progression of the signal in the waveform viewer. If no testbench is available (e.g. exercise sheet 1), the signal values must be set to a value manually using the command **“force”**. You can then use **“run 30”** to simulate again for the specified time. See figure 11.
Beispiele für den **"force"**-Befehl:

```
force Cin b"1"                    # Set Cin to 1
force A x"ABCDEFAA"               # Set A to Hex-value ABCDEFAA
force A b"10101011110011011110111110101010" # Set A to binary value
```

7. Both simulation methods generate a waveform trace file (.vcd - Value Change Dump). You can open this with "gtkwave". To do this see page 13.

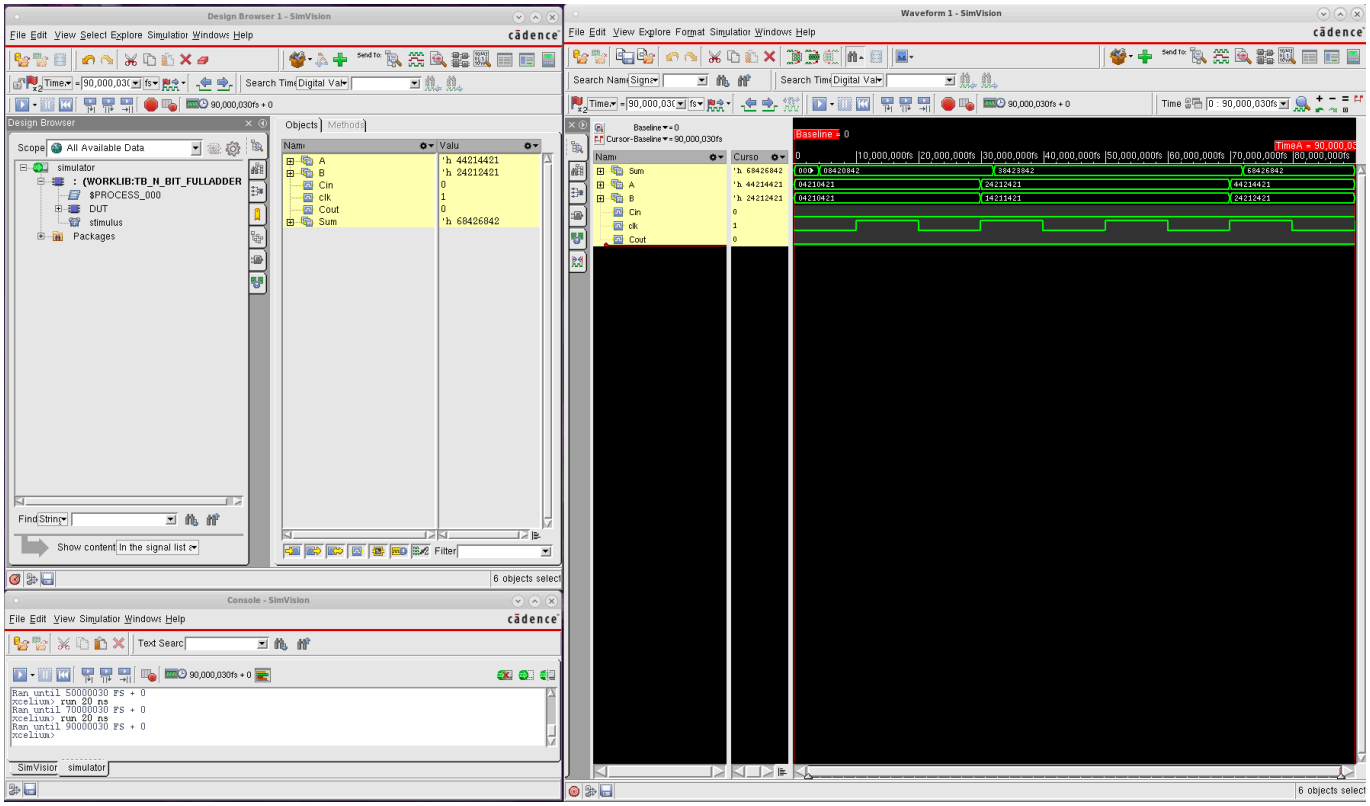


Figure 11: Waveform-Viewer

Generation of a value change dump (VCD file)

You can view the generated VCD file in GTKWave:

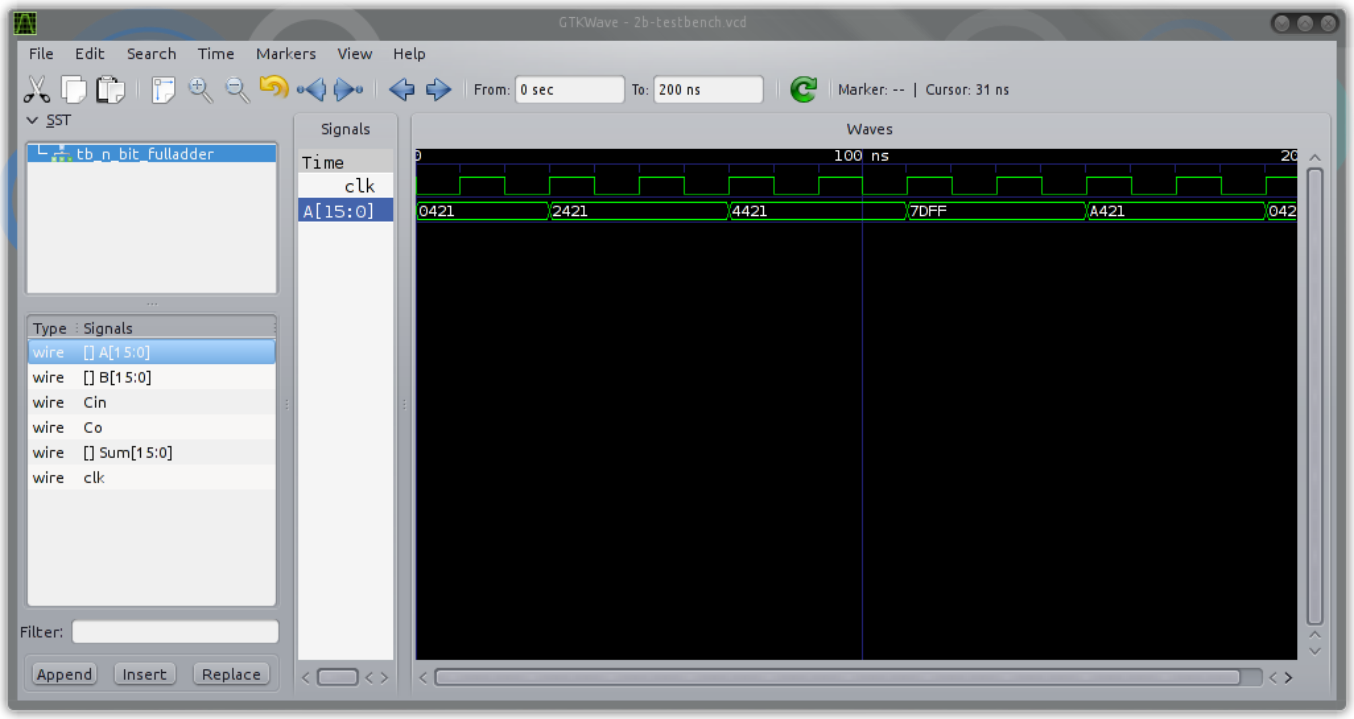


Figure 12: Example of GTKWave