

Введение в Vivado HLS Tool CLI Flow

2020.1

Abstract

This lab introduces how to perform basic actions in the Vivado® HLS command prompt.

Objectives

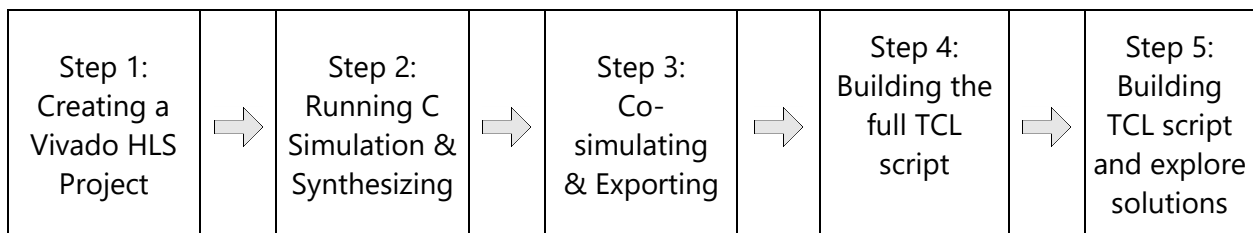
After completing this lab, you will be able to:

- Create a new project in the Vivado HLS tool CLI
- Simulate a C design by using a self-checking test bench
- Synthesize the design
- Simulate an RTL design by using a C test bench
- Implement the design

Introduction

This lab introduces the major features of the Vivado® High-Level Synthesis (HLS) tool Command Line Interface (CLI) flow. You will use the Vivado HLS tool in CLI mode to create a project. You will also simulate, synthesize, and implement the design provided.

General Flow



Before starting the lab

Step 0

Browse to the `C:\Xilinx_trn\HLS_2020\lab2_hls_cli_flow\source` directory using Windows Explorer and open **lab2.c** file, which is the source code for the lab, and **lab2_test.c** file, which is the source code for the test, with your preferred text editor. Explore the C code to understand algorithm and data dependencies.

```
1  #define iter 4
2  short int lab2 (char in_a[iter], char in_b[iter]) {
3      short int tmp;
4      tmp = 0;
5      for(int i = 0; i < iter; i++) {
6          tmp += in_a[i]*in_b[i];
7      }
8      return tmp;}
9
```

```
1  #include <stdio.h>
2
3  int main() {
4      char inArr_a[4] = {10,10,10,10};
5      char inArr_b[4] = {10,10,10,10};
6      int expRes = 400;
7      int gotRes;
8      int pass = 0;
9
10
11     for (int i = 0; i < 3; i++) // Call the function for 3 transactions
12     {
13         gotRes = lab2(inArr_a, inArr_a);
14
15         if (expRes != gotRes) // Test the output against expected results
16         {
17             fprintf(stdout, "  expected %d != got %d ERROR\n", expRes, gotRes);
18             pass = 1;
19         }
20         else
21         {
22             fprintf(stdout, "  expected %d == got %d PASS \n", expRes, gotRes);
23         }
24     }
25
26
27     if (pass == 0)
28     {
29         fprintf(stdout, "-----Pass!-----\n");
30         return 0;
31     }
32     else
33     {
34         fprintf(stderr, "-----Fail!-----\n");
35         return 1;
36     }
37 };
```

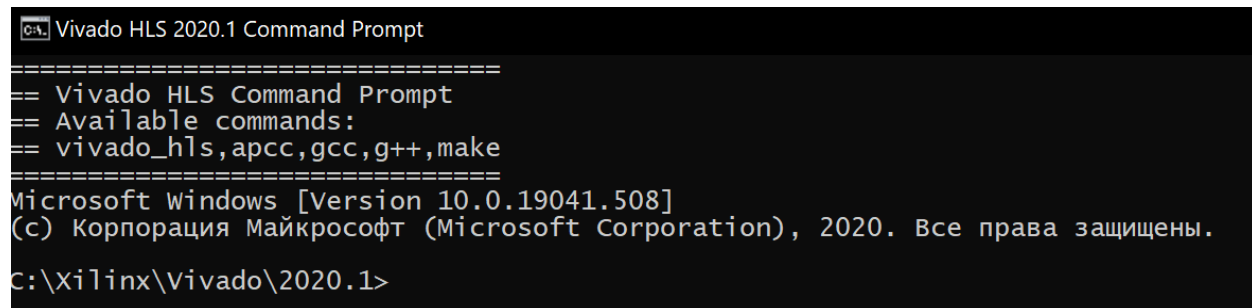
Creating a Vivado HLS Tool Project

Step 1

In this step, you will create a new Vivado HLS tool project, add source files, and provide solution settings for the default solution using the Vivado HLS tool command prompt. Later, you will open the created project in the Vivado HLS tool GUI to have a quick review of the settings were applied.

1-1. Launch the Vivado HLS tool command prompt and change the directory to the **C:\Xilinx_trn\HLS_2020\lab2_hls_cli_flow** working directory.

1-1-1. For Windows 10: **Start > Xilinx Design Tools > Vivado HLS 2020.1 Command Prompt.**

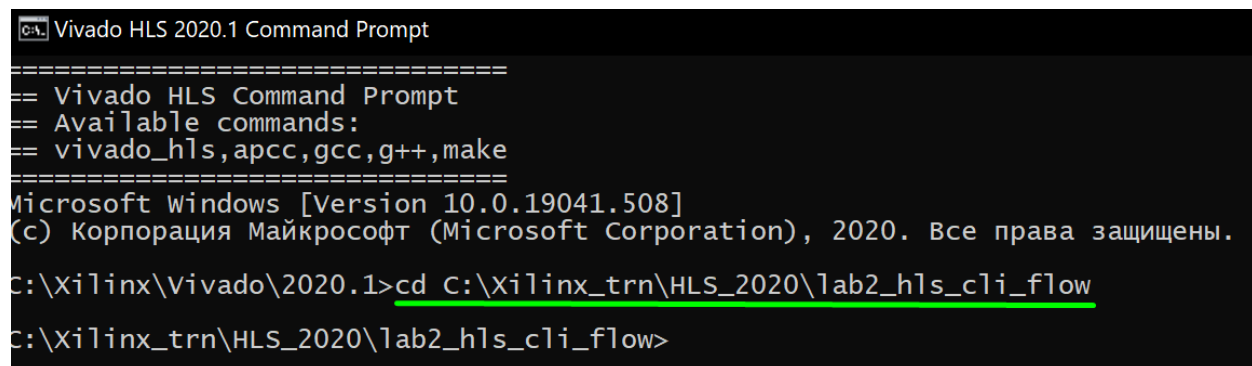


```
C:\ Vivado HLS 2020.1 Command Prompt
=====
== Vivado HLS Command Prompt
== Available commands:
== vivado_hls,apcc,gcc,g++,make
=====
Microsoft Windows [Version 10.0.19041.508]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.
C:\Xilinx\Vivado\2020.1>
```

Figure 1: Vivado HLS Command Prompt

1-1-2. Enter the following command to change the directory to the current working directory:

```
cd C:\Xilinx_trn\HLS_2020\lab2_hls_cli_flow
```



```
C:\ Vivado HLS 2020.1 Command Prompt
=====
== Vivado HLS Command Prompt
== Available commands:
== vivado_hls,apcc,gcc,g++,make
=====
Microsoft Windows [Version 10.0.19041.508]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.
C:\Xilinx\Vivado\2020.1>cd C:\Xilinx_trn\HLS_2020\lab2_hls_cli_flow
C:\Xilinx_trn\HLS_2020\lab2_hls_cli_flow>
```

Figure 2: Changing the Directory

1-2. Write the commands to create a new project, associate source files, and configure solution settings.

1-2-1. Browse to the **C:\Xilinx_trn\HLS_2020\lab2_hls_cli_flow** directory using Windows Explorer and open **lab2.tcl** with your preferred text editor.

1-2-2. Insert the following command at line no. 2 of **lab2.tcl** to create a new project named **lab2_prj**:

```
1  # Insert the command to create new project
2  open_project -reset lab2_prj
```

Question 1

What does the `-reset` switch do in the previous command?

1-2-3. Enter the following command to specify *lab2* as a top-level function to be synthesized in the design:

```
4  # Insert the command to specify the top-level function
5  set_top lab2
```

1-2-4. Enter the following command to add *lab2.c* as a design source file to the project:

```
7  # Insert the command to add design file
8  add_files ./source/lab2.c
```

Question 2

How does the above command differ from adding test bench files to the project?

1-2-5. Enter the following commands to associate *lab2_test.c* as test bench source files to the project:

```
10 # Insert the command to add testbench file
11 add_files -tb ./source/lab2_test.c
```

1-2-6. Enter the following command to create the solution, named *solution1*, to the project:

```
13 # Insert the command to create the solution named solution1
14 open_solution -reset "solution1"
```

1-2-7. Insert the command to associate the *xa7a12tcsq325-1Q* device to the solution:

```
16 # Insert the command to associate the device to the solution1
17 set_part {xa7a12tcsg325-1Q}
```

1-2-8. Insert the commands to associate the clock with a 6ns period to the solution1:

```
19 # Insert the command to associate clock period to the solution1
20 create_clock -period 6 -name clk
21 set_clock_uncertainty 0.1
```

After writing the above commands, the completed part of *lab2.tcl* file should look like the figure below.

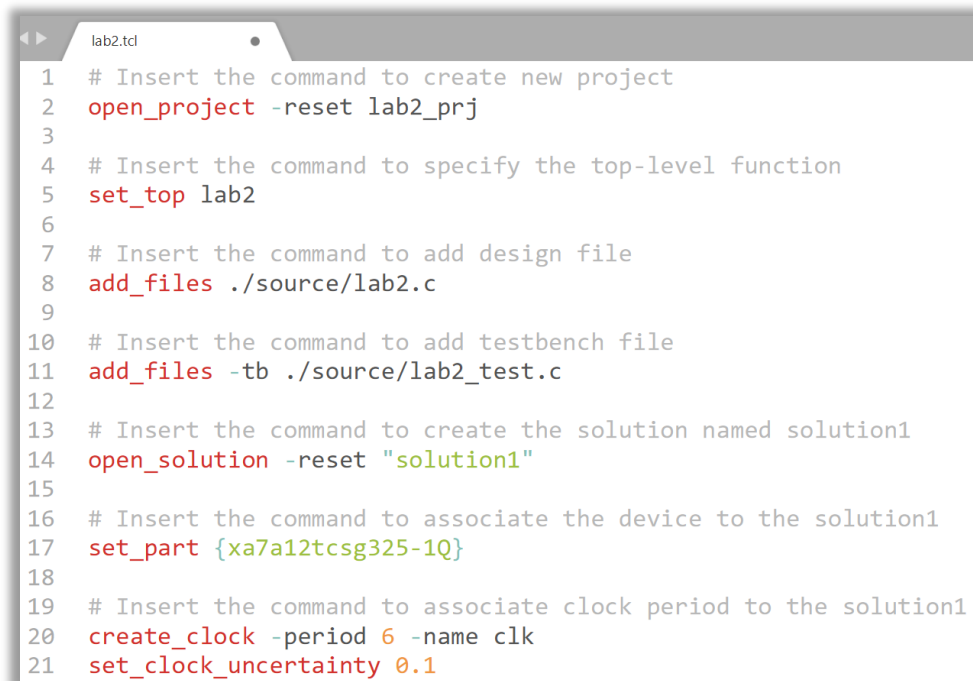
A screenshot of a text editor window titled 'lab2.tcl'. The editor contains 21 lines of Tcl script. Lines 1-3: Line 1 is a comment '# Insert the command to create new project', line 2 is 'open_project -reset lab2_prj', and line 3 is empty. Lines 4-6: Line 4 is a comment '# Insert the command to specify the top-level function', line 5 is 'set_top lab2', and line 6 is empty. Lines 7-9: Line 7 is a comment '# Insert the command to add design file', line 8 is 'add_files ./source/lab2.c', and line 9 is empty. Lines 10-12: Line 10 is a comment '# Insert the command to add testbench file', line 11 is 'add_files -tb ./source/lab2_test.c', and line 12 is empty. Lines 13-15: Line 13 is a comment '# Insert the command to create the solution named solution1', line 14 is 'open_solution -reset "solution1"', and line 15 is empty. Lines 16-18: Line 16 is a comment '# Insert the command to associate the device to the solution1', line 17 is 'set_part {xa7a12tcsg325-1Q}', and line 18 is empty. Lines 19-21: Line 19 is a comment '# Insert the command to associate clock period to the solution1', line 20 is 'create_clock -period 6 -name clk', and line 21 is 'set_clock_uncertainty 0.1'.

Figure 3: lab2.tcl with Project Information

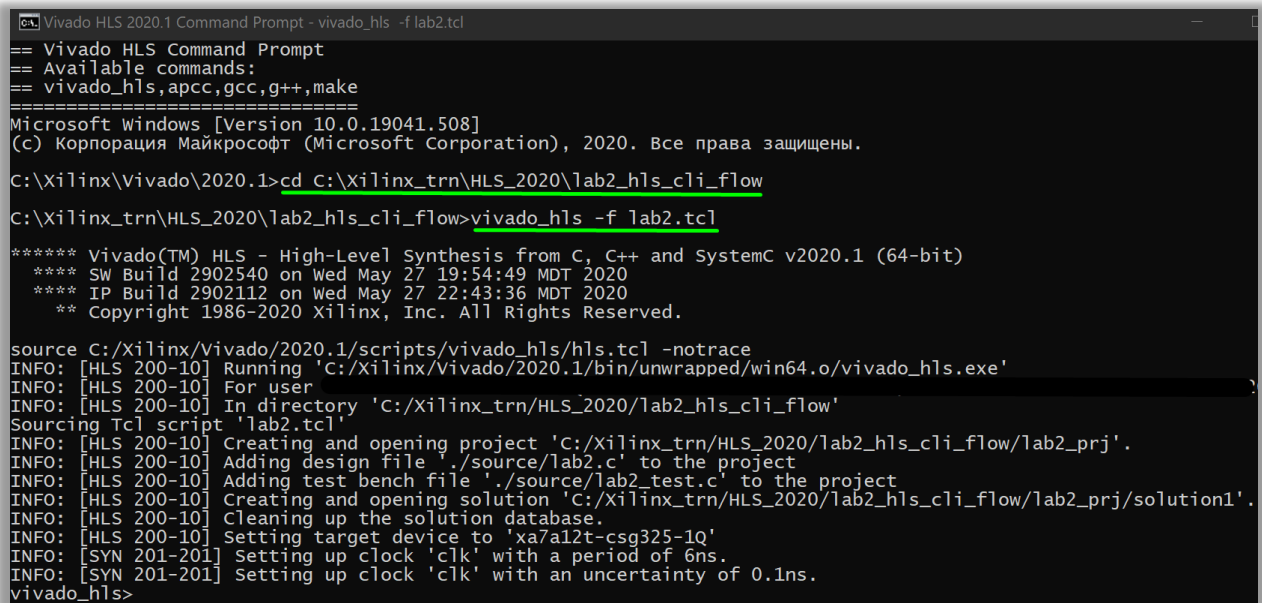
1-2-9. Save the **lab2.tcl** file.

1-2-10. Exit the text editor to close the Tcl file.

1-3. Run the *lab2.tcl* file to create the project and open the created project in the Vivado HLS tool GUI.

1-3-1. Enter the following command in the Vivado HLS tool command prompt to run the *lab2.tcl* file:

```
vivado_hls -f lab2.tcl
```



```
Microsoft Windows [Version 10.0.19041.508]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Xilinx\Vivado\2020.1>cd C:\Xilinx_trn\HLS_2020\lab2_hls_cli_flow
C:\Xilinx_trn\HLS_2020\lab2_hls_cli_flow>vivado_hls -f lab2.tcl

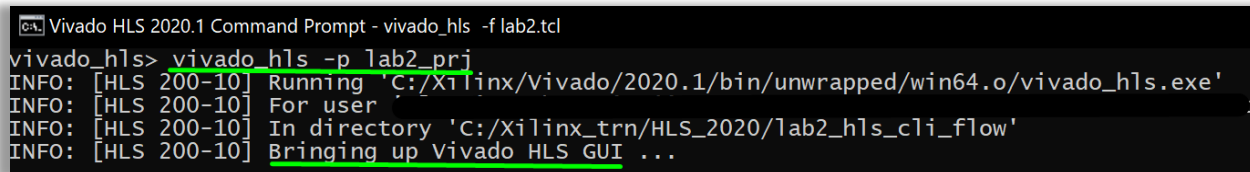
***** Vivado(TM) HLS - High-Level Synthesis from C, C++ and SystemC v2020.1 (64-bit)
***** SW Build 2902540 on Wed May 27 19:54:49 MDT 2020
***** IP Build 2902112 on Wed May 27 22:43:36 MDT 2020
***** Copyright 1986-2020 Xilinx, Inc. All Rights Reserved.

source C:/Xilinx/Vivado/2020.1/scripts/vivado_hls/hls.tcl -notrace
INFO: [HLS 200-10] Running 'C:/Xilinx/Vivado/2020.1/bin/unwrapped/win64.o/vivado_hls.exe'
INFO: [HLS 200-10] For user
INFO: [HLS 200-10] In directory 'C:/Xilinx_trn/HLS_2020/lab2_hls_cli_flow'
Sourcing Tcl script 'lab2.tcl'
INFO: [HLS 200-10] Creating and opening project 'C:/Xilinx_trn/HLS_2020/lab2_hls_cli_flow/lab2_prj'.
INFO: [HLS 200-10] Adding design file './source/lab2.c' to the project
INFO: [HLS 200-10] Adding test bench file './source/lab2_test.c' to the project
INFO: [HLS 200-10] Creating and opening solution 'C:/Xilinx_trn/HLS_2020/lab2_hls_cli_flow/lab2_prj/solution1'.
INFO: [HLS 200-10] Cleaning up the solution database.
INFO: [HLS 200-10] Setting target device to 'xa7a12t-csg325-1q'
INFO: [SYN 201-201] Setting up clock 'clk' with a period of 6ns.
INFO: [SYN 201-201] Setting up clock 'clk' with an uncertainty of 0.1ns.
vivado_hls>
```

Figure 4: Creating the Project

1-3-2. Enter the following command to launch the GUI with the recently created project:

```
vivado_hls -p lab2_prj
```



```
Microsoft Windows [Version 10.0.19041.508]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Xilinx\Vivado\2020.1>vivado_hls -p lab2_prj
vivado_hls> vivado_hls -p lab2_prj
INFO: [HLS 200-10] Running 'C:/Xilinx/Vivado/2020.1/bin/unwrapped/win64.o/vivado_hls.exe'
INFO: [HLS 200-10] For user
INFO: [HLS 200-10] In directory 'C:/Xilinx_trn/HLS_2020/lab2_hls_cli_flow'
INFO: [HLS 200-10] Bringing up Vivado HLS GUI ...
```

Figure 5: Launching the GUI from the CLI

You should see the created project in the Explorer view.

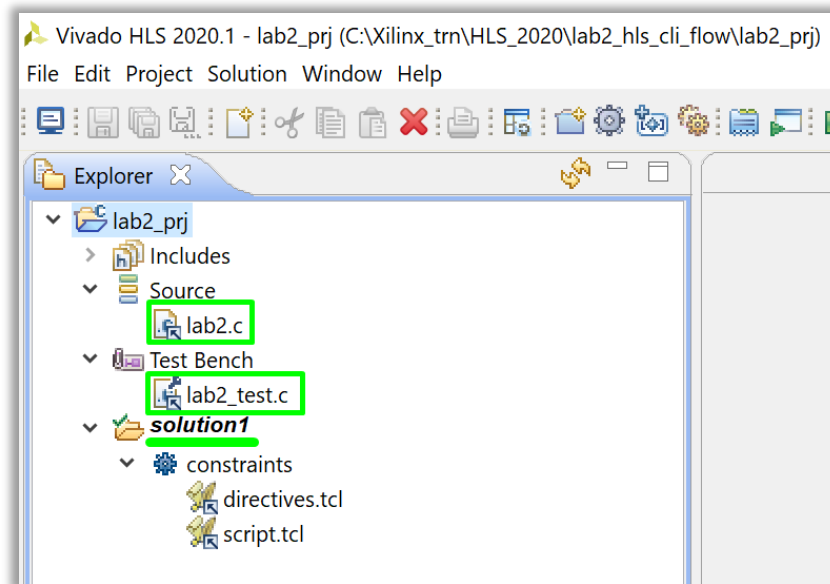


Figure 6: Vivado HLS GUI with Recently Created Project

1-3-3. Select **Project** > **Project Settings** in the Vivado HLS tool GUI.

The Project Settings dialog box opens.

1-3-3-1. Click **Synthesis** to ensure that the design source was added to the project as entered in the Tcl file.

1-3-3-2. Click **Simulation** to ensure that the test bench source was added to the project as entered in the Tcl file

1-3-4. Close the Project Settings dialog box once the source files are verified.

1-3-5. Select **Solution** > **Solution Settings** in the Vivado HLS tool GUI.

The Solution Settings dialog box for the active solution, i.e., *Solution1*, the only solution that exists in this current project, opens.

1-3-6. Select **Synthesis** to ensure that the clock frequency and part number are the ones described in the Tcl file.

1-3-7. Close the Solution Settings dialog box once the clock frequency and device settings are verified.

1-3-8. Select **File** > **Exit** in the Vivado HLS tool GUI to exit the GUI.

Running C Simulation and Synthesizing the Design

Step 2

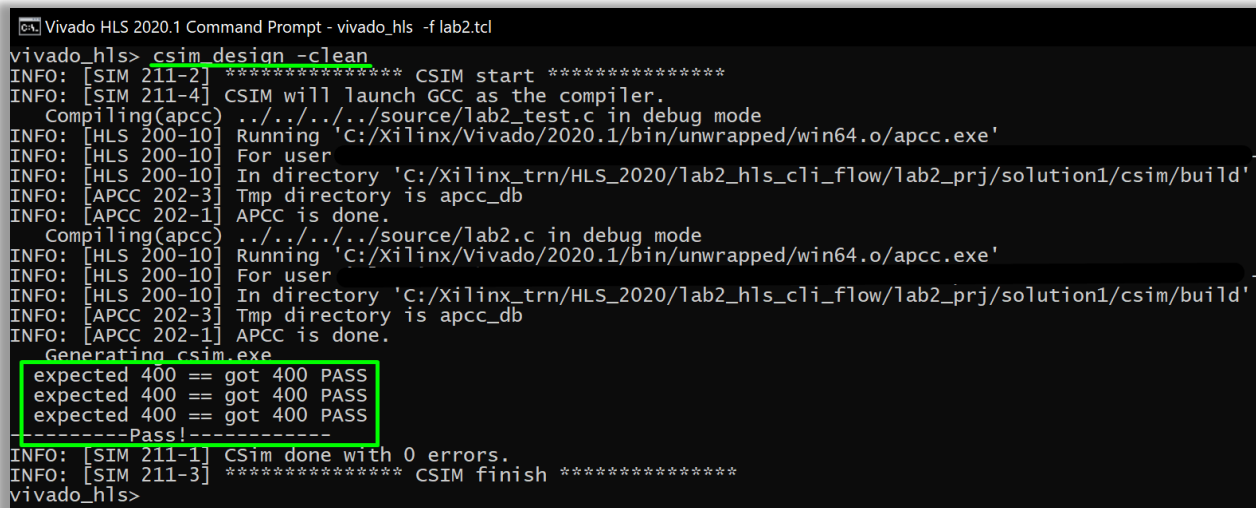
In this step, you will run C simulation and synthesize the design from the Vivado HLS tool command prompt and then open the Vivado HLS tool GUI to review the project status and Synthesis report.

2-1. Simulate the design.

- 2-1-1. Enter the following command in the Vivado HLS tool command prompt to run C simulation:

```
csim_design -clean
```

This command compiles and runs pre-synthesis C simulation of the design using the provided C test bench.



```
Vivado HLS 2020.1 Command Prompt - vivado_hls -f lab2.tcl
vivado_hls> csim_design -clean
INFO: [SIM 211-2] ***** CSIM start *****
INFO: [SIM 211-4] CSIM will launch GCC as the compiler.
Compiling(apcc) ../../../../source/lab2_test.c in debug mode
INFO: [HLS 200-10] Running 'C:/Xilinx/Vivado/2020.1/bin/unwrapped/win64.o/apcc.exe'
INFO: [HLS 200-10] For user
INFO: [HLS 200-10] In directory 'C:/Xilinx_trn/HLS_2020/lab2_hls_cli_flow/lab2_prj/solution1/csim/build'
INFO: [APCC 202-3] Tmp directory is apcc_db
INFO: [APCC 202-1] APCC is done.
Compiling(apcc) ../../../../source/lab2.c in debug mode
INFO: [HLS 200-10] Running 'C:/Xilinx/Vivado/2020.1/bin/unwrapped/win64.o/apcc.exe'
INFO: [HLS 200-10] For user
INFO: [HLS 200-10] In directory 'C:/Xilinx_trn/HLS_2020/lab2_hls_cli_flow/lab2_prj/solution1/csim/build'
INFO: [APCC 202-3] Tmp directory is apcc_db
INFO: [APCC 202-1] APCC is done.
Generating csim.exe
expected 400 == got 400 PASS
expected 400 == got 400 PASS
expected 400 == got 400 PASS
-----Pass!-----
INFO: [SIM 211-1] CSim done with 0 errors.
INFO: [SIM 211-3] ***** CSIM finish *****
vivado_hls>
```

Figure 7: Running C Simulation from the Command Prompt

2-2. Synthesize the design.

- 2-2-1. Enter following command in the Vivado HLS tool command prompt to synthesize the design:

```
csynth_design
```

This will elaborate and perform high-level synthesis on the source files added to the project. This also analyzes the sources files, validates the directives if they are present, and performs some initial code transformations.

```
Vivado HLS 2020.1 Command Prompt - vivado_hls -f lab2.tcl
vivado_hls> csynth_design
INFO: [SCHED 204-61] Option 'relax_ii_for_timing' is enabled, will increase II to preserve clock frequency constraints.
INFO: [HLS 200-10] Analyzing design file './source/lab2.c' ...
INFO: [HLS 200-111] Finished Linking Time (s): cpu = 00:00:03 ; elapsed = 00:15:47 . Memory (MB): peak = 999.867 ; gain = 927.609
INFO: [HLS 200-111] Finished Checking Pragmas Time (s): cpu = 00:00:03 ; elapsed = 00:15:47 . Memory (MB): peak = 999.867 ; gain = 927.609
INFO: [HLS 200-10] Starting code transformations ...
INFO: [HLS 200-111] Finished Standard Transforms Time (s): cpu = 00:00:03 ; elapsed = 00:15:48 . Memory (MB): peak = 999.867 ; gain = 927.609
INFO: [HLS 200-10] Checking synthesizability ...
INFO: [HLS 200-111] Finished Checking Synthesizability Time (s): cpu = 00:00:03 ; elapsed = 00:15:48 . Memory (MB): peak = 999.867 ; gain = 927.609
INFO: [HLS 200-111] Finished Pre-synthesis Time (s): cpu = 00:00:03 ; elapsed = 00:15:48 . Memory (MB): peak = 999.867 ; gain = 927.609
INFO: [HLS 200-111] Finished Architecture Synthesis Time (s): cpu = 00:00:03 ; elapsed = 00:15:48 . Memory (MB): peak = 999.867 ; gain = 927.609
INFO: [HLS 200-10] Starting hardware synthesis ...
INFO: [HLS 200-10] Synthesizing 'lab2' ...
INFO: [HLS 200-10] -----
INFO: [HLS 200-10] -- Implementing module 'lab2' -----
INFO: [HLS 200-10] -----
INFO: [SCHED 204-111] Starting scheduling ...
INFO: [SCHED 204-111] Finished scheduling.
INFO: [HLS 200-111] Elapsed time: 948.303 seconds; current allocated memory: 153.934 MB.
INFO: [BIND 205-100] Starting micro-architecture generation ...
INFO: [BIND 205-101] Performing variable lifetime analysis.
INFO: [BIND 205-101] Exploring resource sharing.
INFO: [BIND 205-101] Binding ...
INFO: [BIND 205-100] Finished micro-architecture generation.
INFO: [HLS 200-111] Elapsed time: 0.163 seconds; current allocated memory: 154.033 MB.
INFO: [HLS 200-10] -----
INFO: [HLS 200-10] -- Generating RTL for module 'lab2' -----
INFO: [HLS 200-10] -----
INFO: [RTGEN 206-500] Setting interface mode on port 'lab2/in_a' to 'ap_memory'.
INFO: [RTGEN 206-500] Setting interface mode on port 'lab2/in_b' to 'ap_memory'.
INFO: [RTGEN 206-500] Setting interface mode on function 'lab2' to 'ap_ctrl_hs'.
INFO: [SYN 201-210] Renamed object name 'lab2_mac_muladd_8s_8s_16ns_16_3_1' to 'lab2_mac_muladd_8bkb' due to the length limit 20
INFO: [RTGEN 206-100] Generating core module 'lab2_mac_muladd_8bkb': 1 instance(s).
INFO: [RTGEN 206-100] Finished creating RTL model for 'lab2'.
INFO: [HLS 200-111] Elapsed time: 0.2 seconds; current allocated memory: 154.234 MB.
INFO: [HLS 200-790] **** Loop Constraint Status: All loop constraints were satisfied.
INFO: [HLS 200-789] **** Estimated Fmax: 261.78 MHz
INFO: [HLS 200-111] Finished generating all RTL models Time (s): cpu = 00:00:04 ; elapsed = 00:15:50 . Memory (MB): peak = 999.867 ; gain = 927.609
INFO: [VHDL 208-304] Generating VHDL RTL for lab2.
INFO: [VLOG 209-307] Generating Verilog RTL for lab2.
vivado_hls>
```

Figure 8: Performing Synthesis from the Command Prompt

2-3. Verify the Synthesis report in the Vivado HLS tool GUI.

- 2-3-1. Enter the following command to launch the GUI:

```
vivado_hls -p lab2_prj
```

- 2-3-2. Select **Solution** > **Open Report** > **Synthesis** in the Vivado HLS tool GUI if the Synthesis report does not open automatically.

- 2-3-3. Analyze the report file.

Question 3

Write down the following details from the Synthesis report and Performance profile:

- Estimated clock period:
- Max latency (cycles):
- Max latency (time): Max latency (cycles) * Estimated clock period =
- Loop 1 trip count:

- Loop 1 Iteration Latency:
- Loop1 Latency:
- Iteration Interval:
- Number of BRAM_18K:
- Number of DSP48E used:
- Number of FFs used:
- Number of LUTs used:
- Number of URAM used:

Question 4

Explain why the lab2 Latency is one cycle more than Loop1 latency.

Explain why the lab2 Iteration Interval is one cycle more than lab2 latency.

2-3-4. Select **Interface** > **Summary** in the Outline pane (Synthesis tab).

The report also shows the top-level interface signals generated by the tools.

2-3-5. Select **File** > **Exit** in the Vivado HLS tool GUI to exit the GUI.

Co-simulating and Exporting the RTL

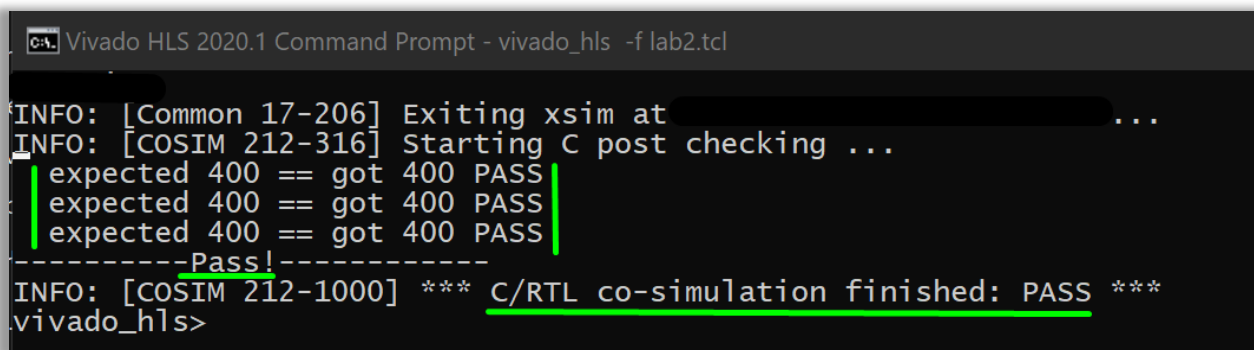
Step 3

In this step, you will perform C/RTL co-simulation on the generated RTL files by using the C test bench from the Vivado HLS tool command prompt. In addition, you will generate the IP from the Vivado HLS tool command prompt.

3-1. Perform C/RTL co-simulation.

- 3-1-1. Enter the following command in the Vivado HLS tool command prompt to execute post-synthesis co-simulation:

```
cosim_design -trace_level all -tool xsim
```



```
C:\> Vivado HLS 2020.1 Command Prompt - vivado_hls -f lab2.tcl

INFO: [Common 17-206] Exiting xsim at ...
INFO: [COSIM 212-316] Starting C post checking ...
expected 400 == got 400 PASS
expected 400 == got 400 PASS
expected 400 == got 400 PASS
-----Pass!-----
INFO: [COSIM 212-1000] *** C/RTL co-simulation finished: PASS ***
vivado_hls>
```

Figure 9: C/RTL Co-simulation

Notice that the message ***** C/RTL co-simulation finished: PASS ***** is displayed.

3-2. Export the design to IP.

- 3-2-1. Enter the following command to perform an RTL implementation for Verilog HDL:

```
export_design -flow impl -format ip_catalog
```

The `-flow` option will perform RTL synthesis or RTL synthesis and implementation on the generated IP based on the option given (syn|impl). Implementation is run to evaluate and provide confidence that the RTL will meet its estimated timing and area goals. These results are not included as part of the exported package.

```
#=== Post-Implementation Resource usage ===
SLICE:          3
LUT:            9
FF:            11
DSP:            1
BRAM:           0
SRL:            0
#=== Final timing ===
CP required:    6.000
CP achieved post-synthesis:  2.954
CP achieved post-implementation: 2.694
Timing met
```

Figure 10: Exporting Synthesized RTL as an IP

Note: Ignore the warnings.

3-3. Verify the results in the Vivado HLS tool GUI.

3-3-1. Enter the following command to launch the GUI:

```
vivado_hls -p lab2_prj
```

3-3-2. Select **Solution** > **Open Report** > **Cosimulation** in the Vivado HLS tool GUI to open the RTL Simulation report.

3-3-3. Select **Solution** > **Open Report** > **Export RTL** to open the Export RTL report.

Question 5

Compare Target Clock Period (CP), Estimated CP and CP achieved after implementation.

3-3-4. Close the Vivado HLS tool GUI.

3-3-5. Enter the following command to close the Vivado HLS tool command prompt:

```
exit
```

Building the full lab2.tcl file

Step 4

4-1. Complete the lab2.tcl file.

- 4-1-1. Browse to the `C:\Xilinx_trn\HLS_2020\lab2_hls_cli_flow` directory using Windows Explorer and open **lab2.tcl** with your preferred text editor.
- 4-1-2. Insert the following, highlighted, commands into *lab2.tcl* to complete the script:

```
1  # Insert the command to create new project
2  open_project -reset lab2_prj
3
4  # Insert the command to add design file
5  add_files ./source/lab2.c
6
7  # Insert the command to specify the top-level function
8  set_top lab2
9
10 # Insert the command to add testbench file
11 add_files -tb ./source/lab2_test.c
12
13 # Insert the command to create the solution named solution1
14 open_solution -reset "solution1"
15
16 # Insert the command to associate the device to the solution1
17 set_part {xa7a12tcsg325-1Q}
18
19 # Insert the command to associate clock period to the solution1
20 create_clock -period 6 -name clk
21 set_clock_uncertainty 0.1
22
23 # Insert the command to run C simulation
24 csim_design -clean
25
26 # Insert the command to Synthesize the design
27 csynth_design
28
29 # Insert the command to perform C/RTL Cosimulation
30 cosim_design -trace_level all -tool xsim
31
32 # Insert the command to export the design to IP
33 export_design -flow impl -format ip_catalog
34
```

Figure 11: Full lab2.tcl script

- 4-1-3. Save the **lab2.tcl** file.
- 4-1-4. Exit the text editor to close the Tcl file.

4-2. Run complete script.

- 4-2-1. Enter the following command in the Vivado HLS tool command prompt to run the *lab2.tcl* file:

```
vivado_hls -f lab2.tcl
```

Full procedure from creating the project up to exporting the IP will be implemented.

- 4-2-2. Enter the following command to close the Vivado HLS tool command prompt:

```
exit
```

Building the lab2_ex.tcl file and conduct an exploration**Step 5****5-1. Complete the lab2_ex.tcl file.**

- 5-1-1. Browse to the `C:\Xilinx_trn\HLS_2020\lab2_hls_cli_flow` directory using Windows Explorer and open **lab2_ex.tcl** with your preferred text editor.
- 5-1-2. Insert the commands into `lab2_ex.tcl` to complete the script for exploration procedure:

```
1  # The command to create new project
2  open_project -reset lab2_explore_prj
3
4  # The command to add design file
5  add_files ./source/lab2.c
6
7  # The command to specify the top-level function
8  set_top lab2
9
10 # The command to add testbench file
11 add_files -tb ./source/lab2_test.c
12
13 # The command to create the base solution named base
14 open_solution -reset "base"
15
16 # The command to associate the device to the solution1
17 set_part {xa7a12tcsg325-1Q}
18
19 # The command to associate clock period to the solution1
20 create_clock -period 6 -name clk
21 set_clock_uncertainty 0.1
22
23 # The comamnd to run the base C simulaiton
24 csim_design -clean
```

Figure 12: lab2_ex.tcl script (part1)

```

25
26 # Build the lists of solution's name and target delay
27 set all_solutions {ex_sol1 ex_sol2 ex_sol3 ex_sol4 ex_sol5 ex_sol6}
28 set all_period    {{6}      {7}      {8}      {9}      {10}    {11}  }
29
30 # The comand to run the loop for the lists
31 foreach the_solution $all_solutions the_period $all_period {
32
33 # The command to create the solution named from the list
34 open_solution -reset $the_solution
35
36 # The command to associate clock period to the solution from the list
37 create_clock -period $the_period -name clk
38 set_clock_uncertainty 0.1
39
40 # The comand to associate the device to the solution1
41 set_part {xa7a12tcsg325-1Q}
42
43 # The comand to Synthesize the design
44 csynth_design
45
46 # The comand to perform C/RTL Cosimulation
47 cosim_design -trace_level all -tool xsim
48
49 # The closing bracket for the loop
50 }

```

Figure 13: lab2_ex.tcl script (part2)

5-1-3. Save the **lab2_ex.tcl** file.

5-1-4. Exit the text editor to close the Tcl file.

5-2. Run complete script.

5-2-1. Enter the following command in the Vivado HLS tool command prompt to run the *lab2_ex.tcl* file:

```
vivado_hls -f lab2_ex.tcl
```

Full procedure from creating the project up to co-simulation for each delay in the list will be implemented.

5-2-2. Enter the following command to close the Vivado HLS tool command prompt:

```
exit
```

5-3. Verify and compare the Synthesis reports in the Vivado HLS tool GUI.

5-3-1. Enter the following command to launch the GUI:

```
vivado_hls -p lab2_explore_prj
```

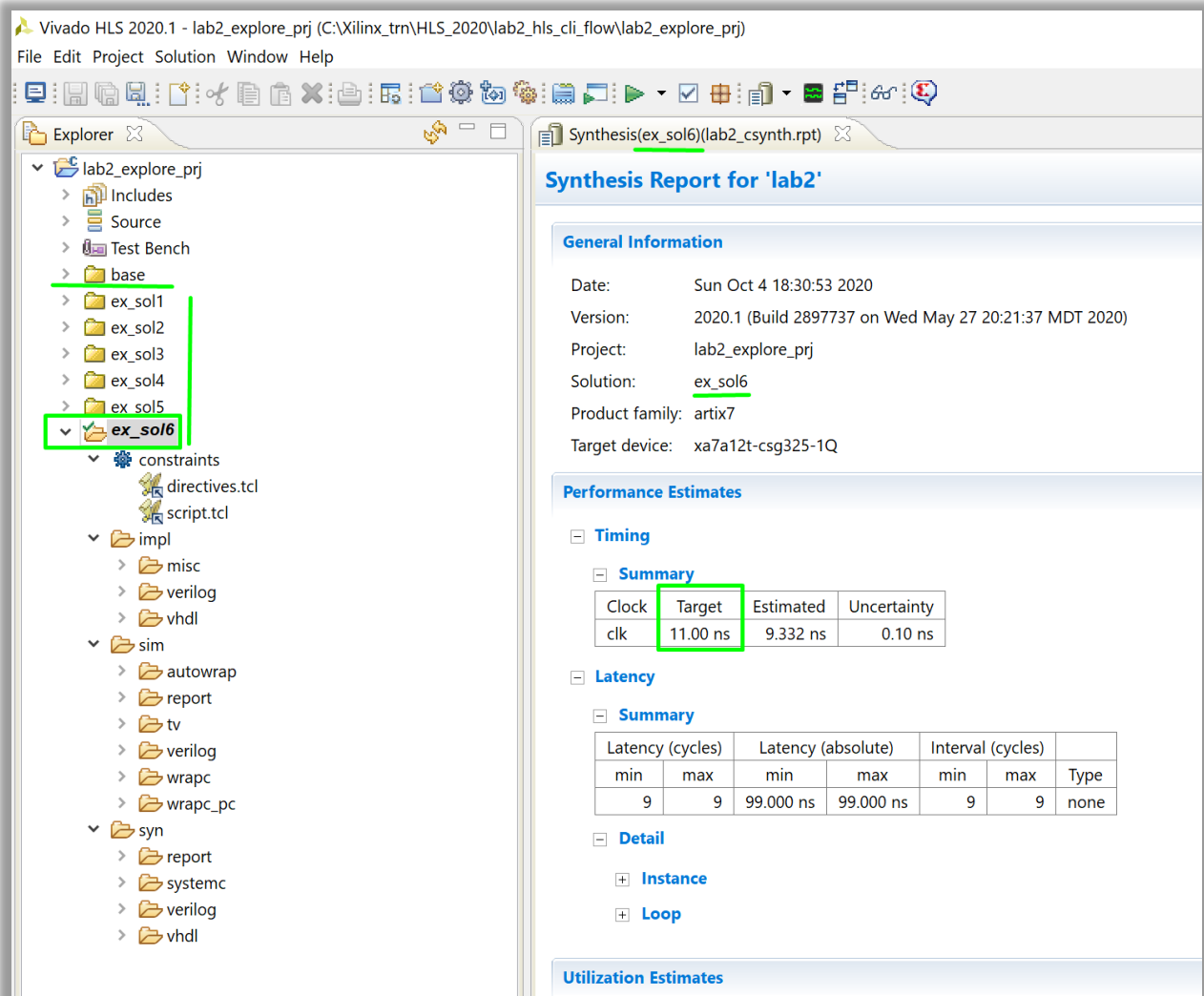



Figure 14: Vivado HLS tool GUI for lab2_explore_prj project

- 5-3-2. Verify that there are *base* solution and six solutions: *ex_sol[6:1]*.
- 5-3-3. Verify that *ex_sol6* is active and the target clock period is equal 11ns.
- 5-3-4. Select **Project** > **Compare Reports** in the Vivado HLS tool GUI.
- 5-3-5. In the window appeared select all solutions for comparing.

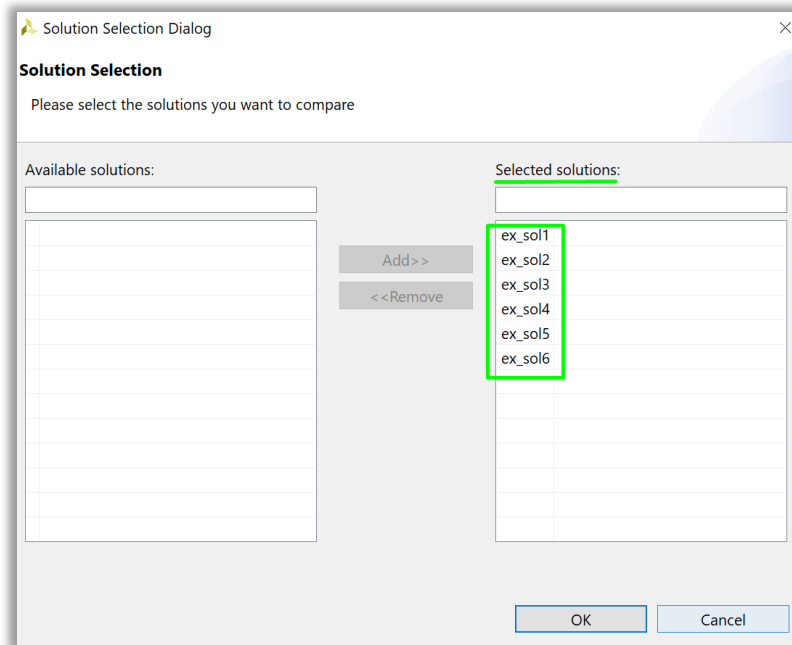


Figure 15: Solutions to compare - report

5-3-6. Use data from the *compare_report* tab to complete the spreadsheet as it done on the following figure (An example of the spreadsheet is in file *./source/lab2_ex.xlsx*).

Pay attention that Latency (ns) is calculated by multiplying Clock estimated (ns) on Latency (cycles).

		ex_sol1	ex_sol2	ex_sol3	ex_sol4	ex_sol5	ex_sol6
Clock	Target (ns)	6	7	8	9	10	11
	Estimated (ns)	3,82	6,44	7,18	7,18	9,33	9,33
Latency	(cycles)	17	13	13	13	9	9
	(ns)	65	84	93	93	84	84
Resources	BRAM_18K	0	0	0	0	0	0
	DSP48E	1	1	1	1	1	1
	FF	27	26	42	42	25	25
	LUT	72	66	66	66	60	60
	URAM	0	0	0	0	0	0

Figure 16: Solutions to compare - spreadsheet

5-3-7. Construct the diagram by using Latency (ns) and FF, LUT resources (DSP, BRAM18, URAM have constant values and can be omitted). (An example of the spreadsheet with the diagram is in file *./source/lab2_ex.xlsx*).

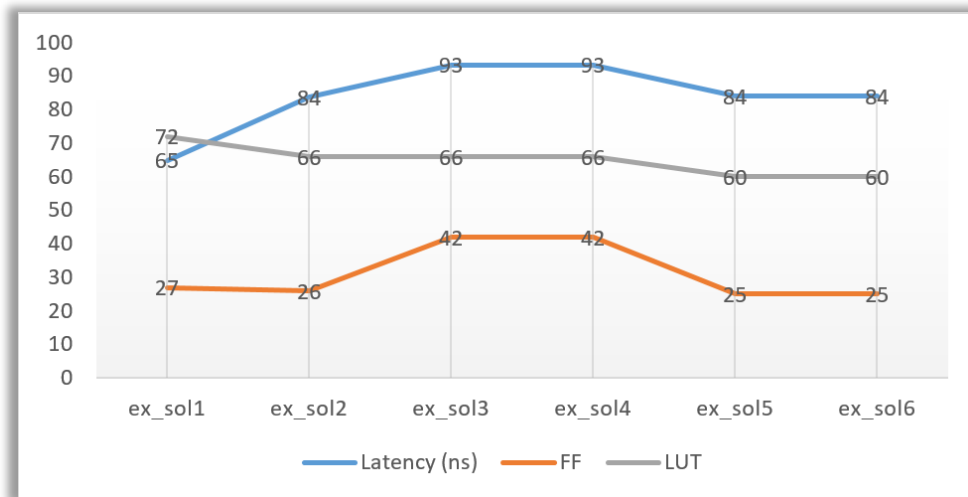


Figure 17: Solutions to compare - diagram

Question 6

Explain which solution, in your opinion, is the best (criteria are max performance and min resources) and why do you think so.

Summary

In this lab, you learned how to use the Vivado HLS tool command prompt to:

- Create the Vivado HLS tool project
- Create a solution with the desired settings
- Execute major actions in the Vivado HLS (simulate, synthesize, cosimulate and export the RTL of the design) design flow