Vitis HLS Design Flow

Field Application Engineer
Adaptive and Embedded Computing Group (AECG)

AMD
together we advance_

Revision History

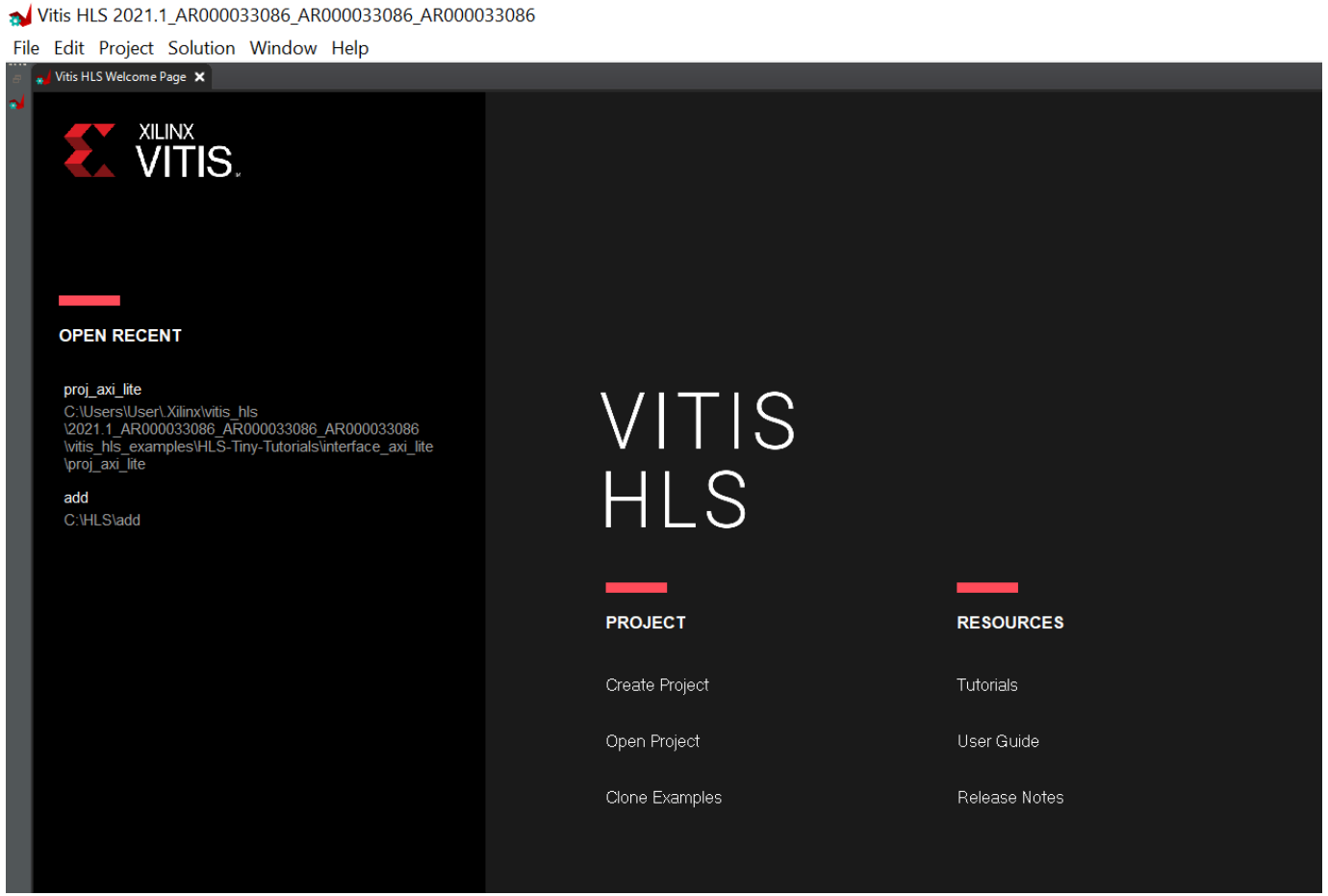
Date	Version	Description
11/08/23	1.0	Initial version for flow introduction.

Vitis HLS 2021.1 Part

Vitis HLS Design Flow

Open Vitis HLS 2021.1

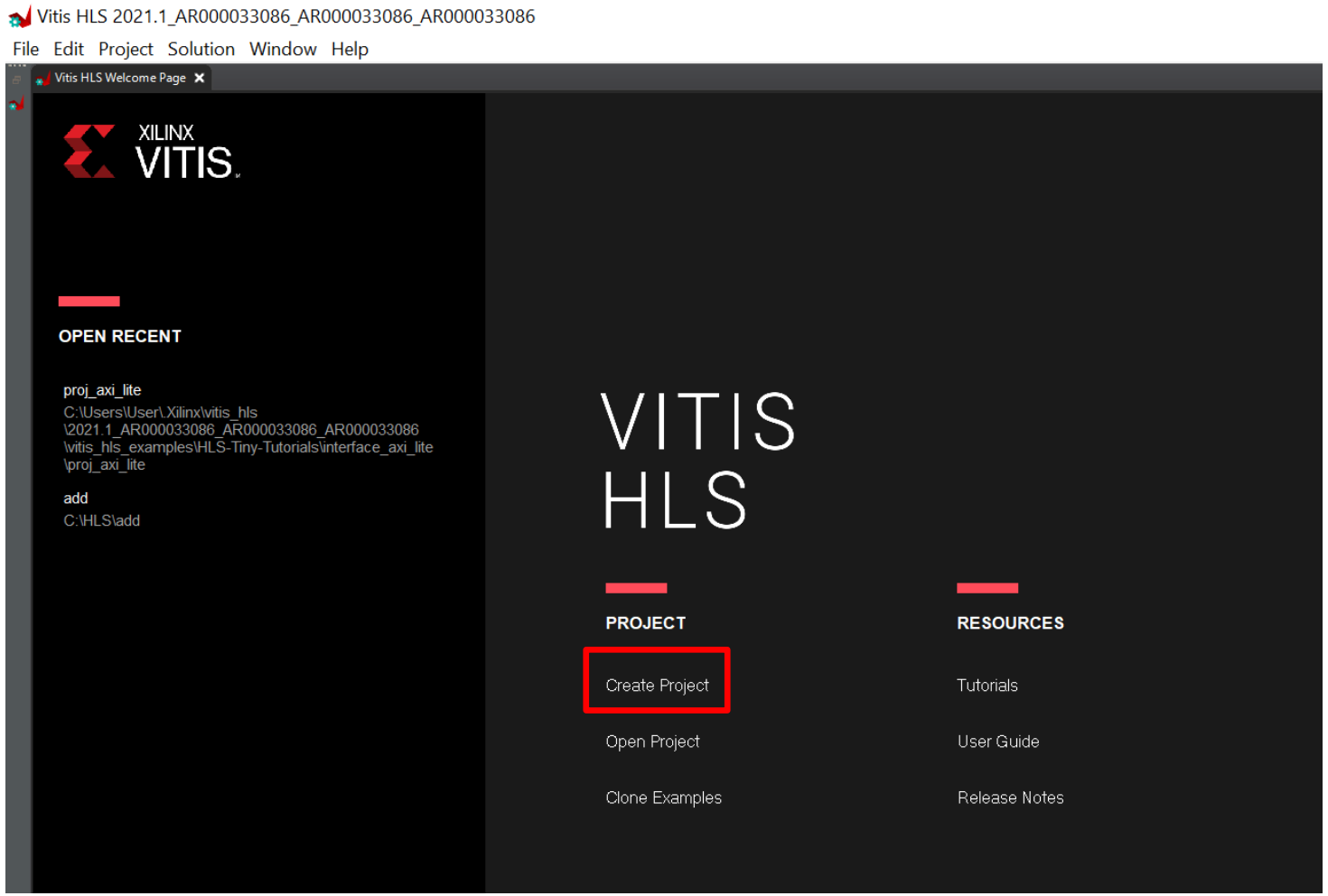
The GUI will be like the right image



Vitis HLS Design Flow

We can create a custom project or download example code from Xilinx Vitis HLS github

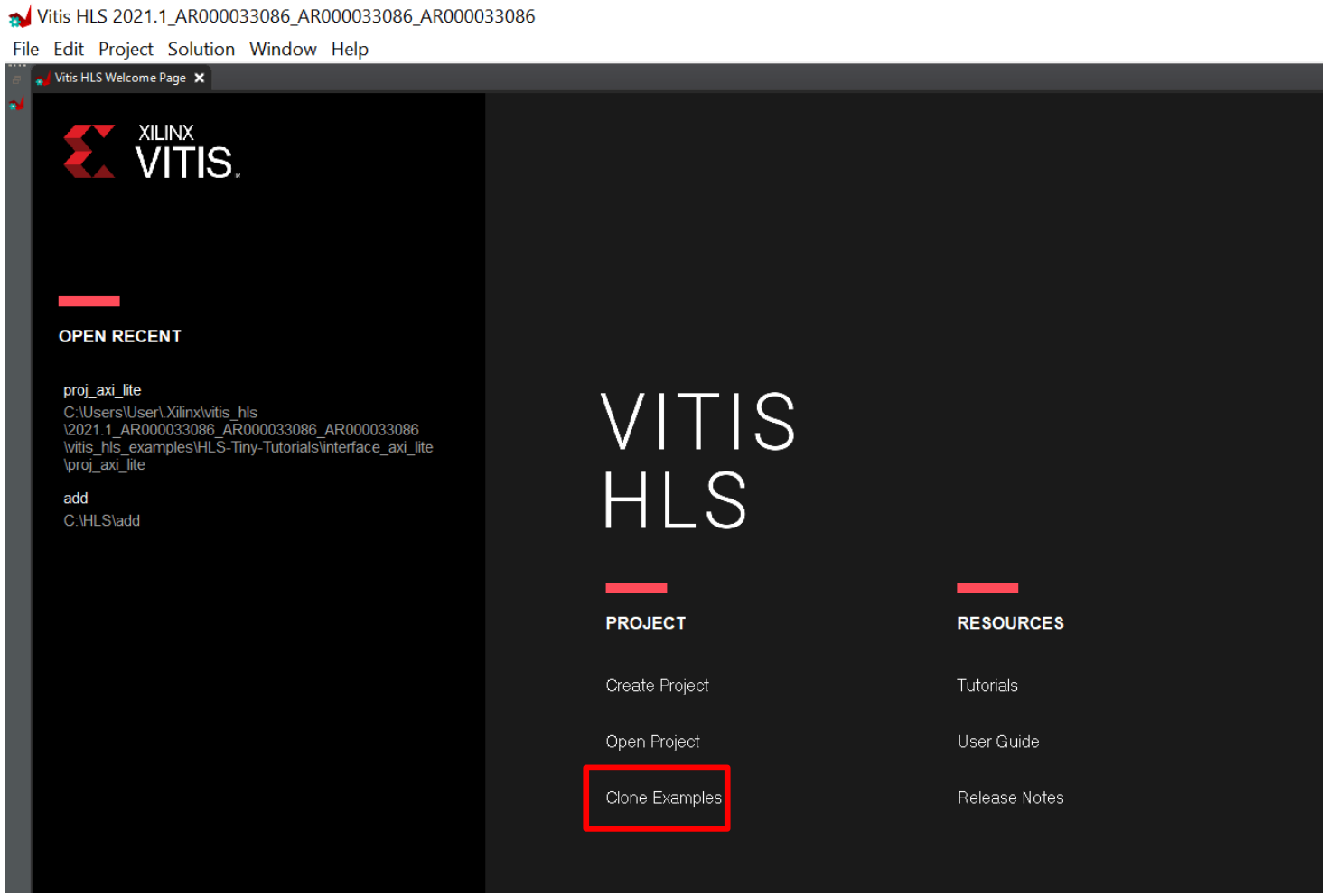
1. Create a custom project



Vitis HLS Design Flow

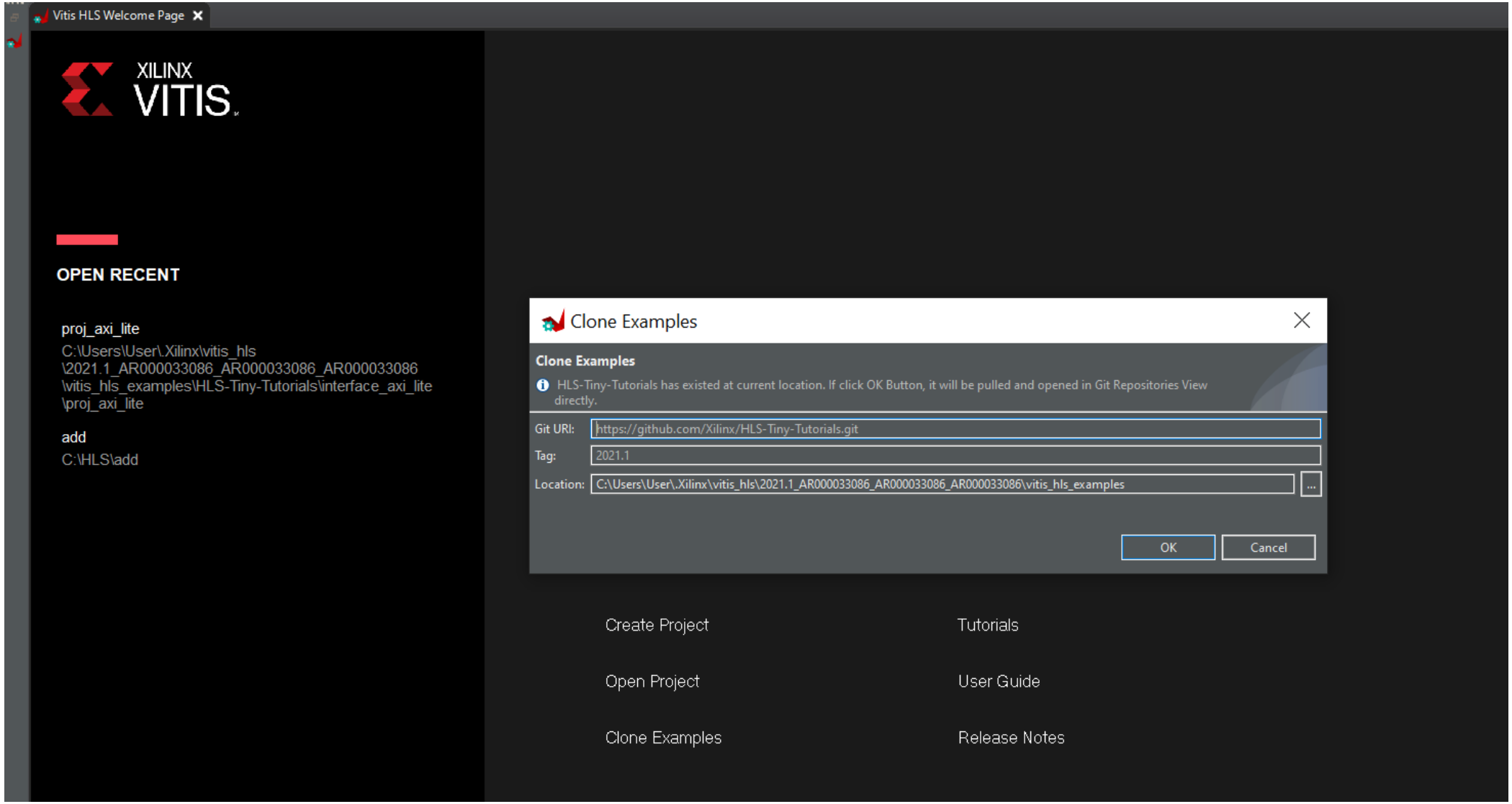
We can create a custom project or download example code from Xilinx Vitis HLS github

2. Download from Xilinx Vitis HLS github



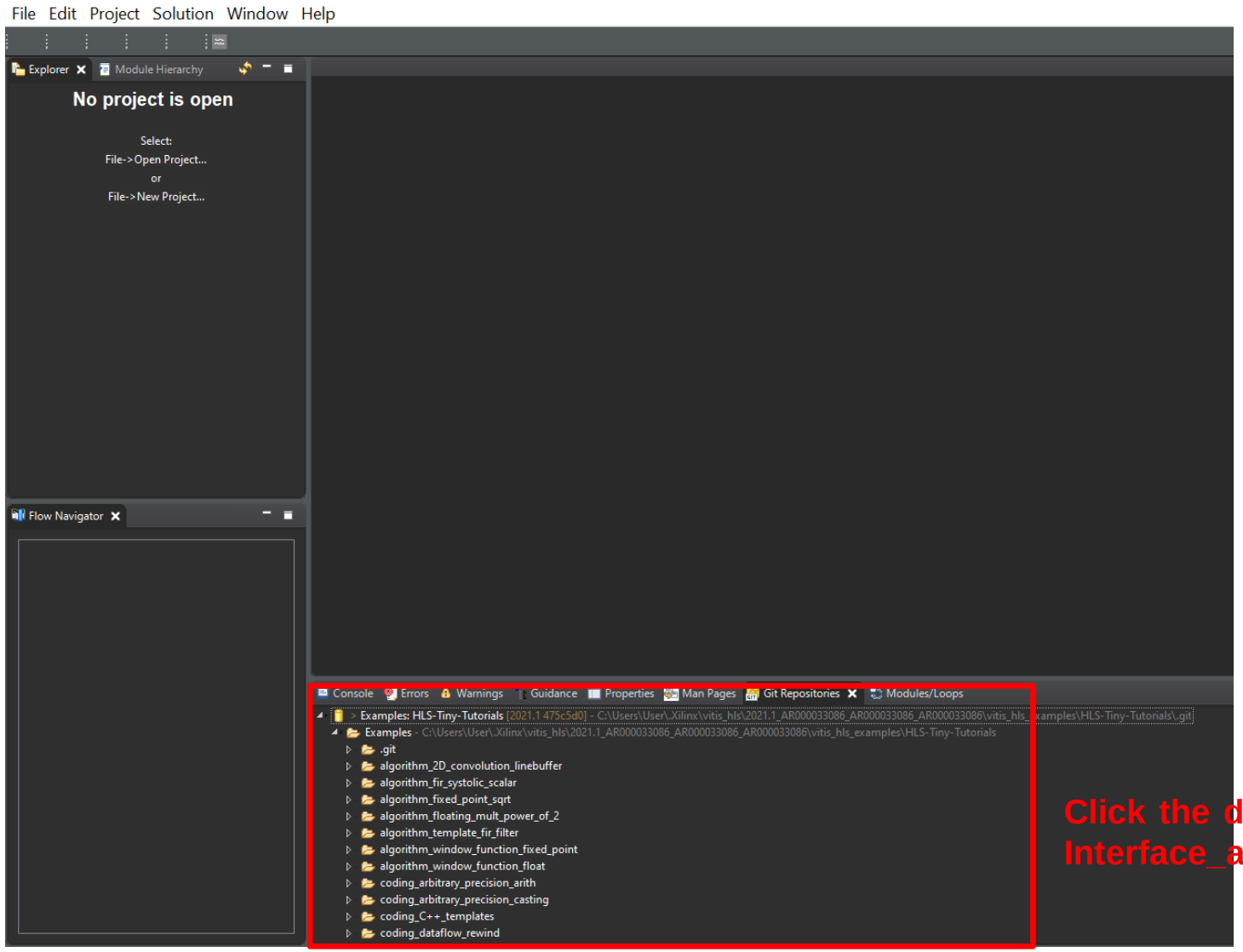
Vitis HLS Design Flow

We will start from example code from Xilinx Vitis HLS github first, keep option default.



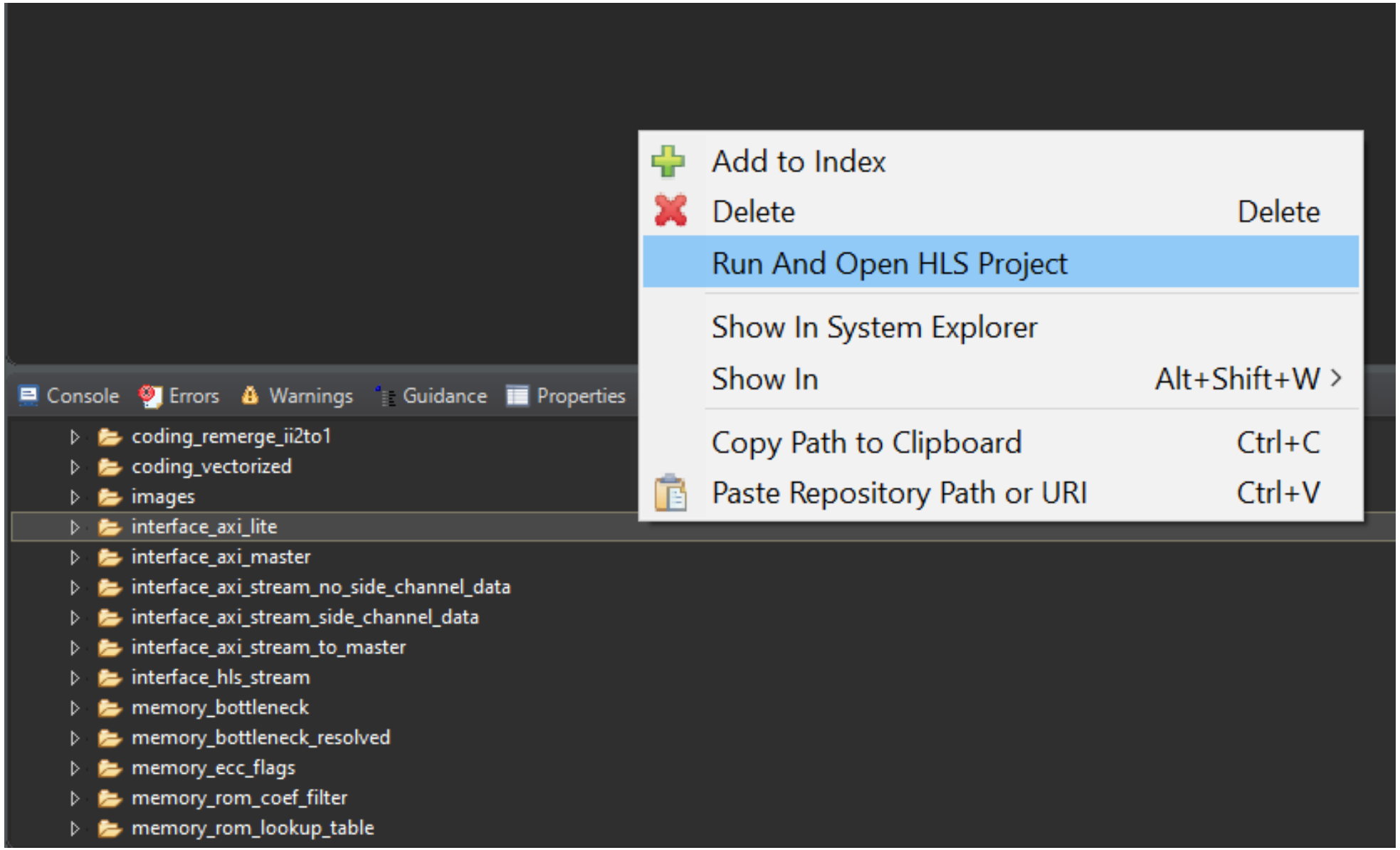
Vitis HLS Design Flow

Click OK will download the whole example code directory automatically.

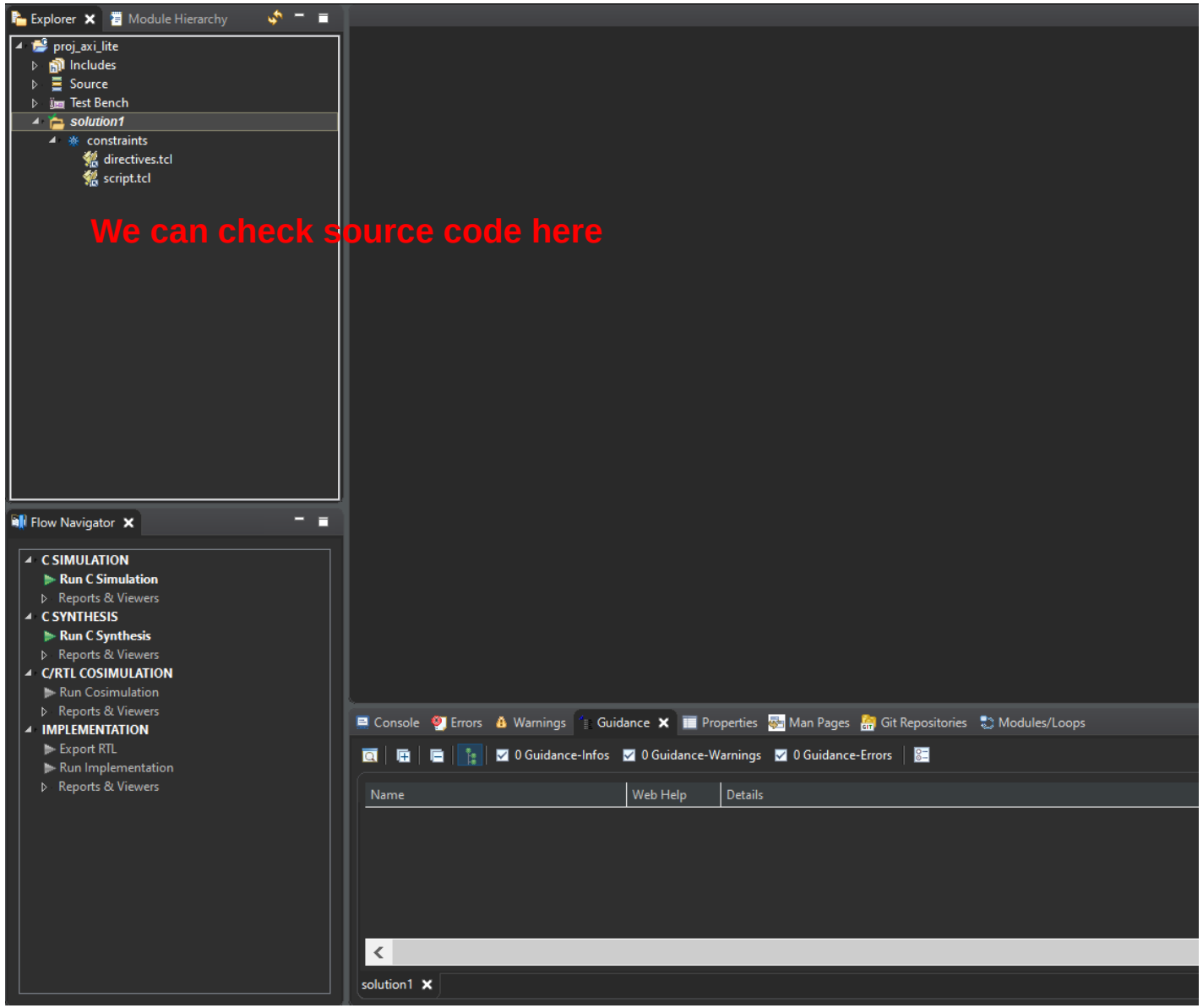


Click the directory you want, we choose Interface_axi_lite this time

Vitis HLS Design Flow



Vitis HLS Design Flow



Vitis HLS Design Flow

First we can check example.cpp

Explorer | Module Hierarchy

- proj_axi_lite
  - Includes
  - Source
    - example.cpp 7c2535c removed register optior
  - Test Bench
    - example_test.cpp 697ea1f Changed year
  - *solution1*
    - constraints
      - directives.tcl
      - script.tcl

example.cpp

```cpp
/*
 * Copyright 2020 Xilinx, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

#include <stdio.h>


int example(int *a, int *b, int *c)
{
#pragma HLS INTERFACE s_axilite port=a bundle=BUS_A
#pragma HLS INTERFACE s_axilite port=b bundle=BUS_A
#pragma HLS INTERFACE s_axilite port=c bundle=BUS_A
#pragma HLS INTERFACE s_axilite port=return bundle=BUS_A

    *c += *a + *b;

    return *c;
}
```

Vitis HLS Design Flow

```
#include <stdio.h>


int example(int *a, int *b, int *c)
{
#pragma HLS INTERFACE s_axilite port=a bundle=BUS_A
#pragma HLS INTERFACE s_axilite port=b bundle=BUS_A
#pragma HLS INTERFACE s_axilite port=c bundle=BUS_A
#pragma HLS INTERFACE s_axilite port=return bundle=BUS_A

    *c += *a + *b;

    return *c;
}
```

Define AXI-Lite Port

#pragma HLS interface <mode> port=<name> (register) bundle=<string>

Vitis HLS Design Flow

#pragma HLS interface <mode> port=<name> (register) bundle=<string>

- *<mode>*:

Specifies the interface protocol mode for the function arguments. In this case we chose s_axilite.

- port=*<name>*:

Specifies the name of the function argument which the INTERFACE pragma applies to.

- (register):

Is an optional keyword to register (i.e. store) the signal and any associated protocol signals. It causes the signals to persist until at least the last cycle of the function execution. This option is not applicable to axilite.

Vitis HLS Design Flow

#pragma HLS interface <mode> port=<name> (register) bundle=<string>

- bundle=*<string>:*

This keyword allows you to manually group the port signals into one data bus. If the function return is specified as an AXI4-Lite interface (i.e. line 25 in the example code), all of the data ports are automatically bundled into a single bus. Vitis HLS would use the default bundle name ***control***, if we did not explicitly provide a bundle name here. This is a common practice when another device, such as a CPU, is used to configure and control when the block starts and stops operation.

- s_axilite port=return:

Setting a function argument of type s_axilite and port name ‘return’ will create an interrupt signal in the IP block. You can program the interrupt through the AXI4-Lite interface, and the C driver files.

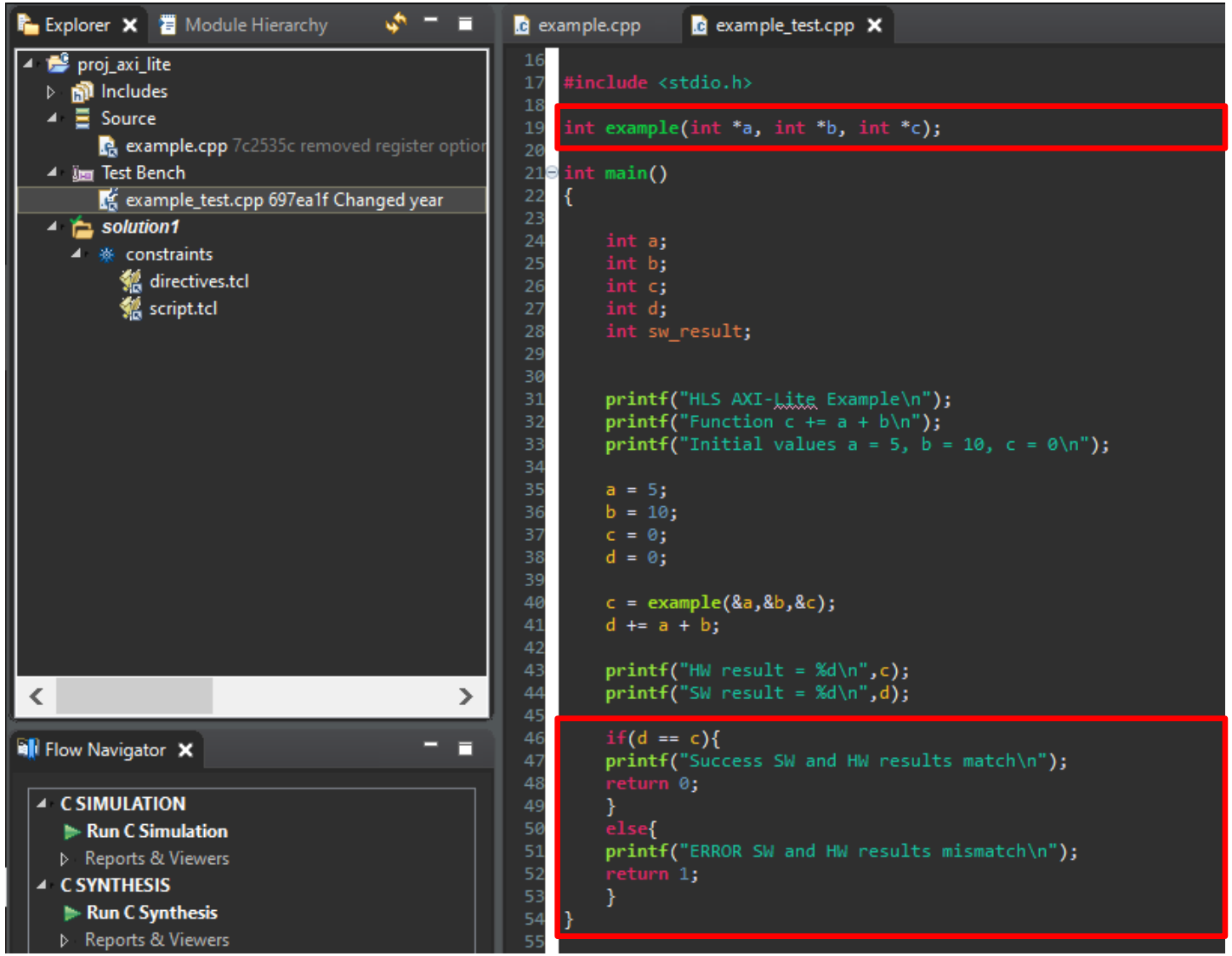
Vitis HLS Design Flow

```
#include <stdio.h>


int example(int *a, int *b, int *c)
{
#pragma HLS INTERFACE s_axilite port=a bundle=BUS_A
#pragma HLS INTERFACE s_axilite port=b bundle=BUS_A
#pragma HLS INTERFACE s_axilite port=c bundle=BUS_A
#pragma HLS INTERFACE s_axilite port=return bundle=BUS_A

    *c += *a + *b;

    return *c;
}
```

Simple Function or Algorithm

Vitis HLS Design Flow

Second we can check example_test.cpp, is like RTL testbench



call function which will be tested

check function behavior

Vitis HLS Design Flow

C simulation ---> run example.cpp and example_test.cpp

File Edit Project Solution Window Help

Explorer

proj_axi_lite
- Includes
- Source
  - example.cpp
- Test Bench
  - example_test.cpp
- solution1
  - constraints
    - directives.tcl
    - script.tcl

C Simulation
C Synthesis
Co-Simulation
Export RTL
Implementation

example.cpp | example_test.cpp

```
#include <stdio.h>

int example(int *a, int *b, int *c);

int main()
{

    int a;
    int b;
    int c;
    int d;
    int sw_result;


    printf("HLS AXI-Lite Example\n");
    printf("Function c += a + b\n");
    printf("Initial values a = 5, b = 1

    a = 5;
    b = 10;
    c = 0;
    d = 0;
```

C Simulation Dialog

C Simulation

Options
- Launch Debugger
- Build Only
- Clean Build
- Optimizing Compile
- Enable Pre-synthesis Control Flow Viewer

Input Arguments

Do not show this dialog again

OK | Cancel

Vitis HLS Design Flow

C synthesis ---> Convert C code to RTL code

example.cpp | example_test.cpp | example_csim.log

```
INFO: [SIM 2] *************** CSIM start ***************
INFO: [SIM 4] CSIM will launch GCC as the compiler.
   Compiling ../../../../example_test.cpp in debug mode
   Compiling ../../../../example.cpp in debug mode
   Generating csim.exe
HLS AXI-Lite Example
Function c += a + b
Initial values a = 5, b = 10, c = 0
HW result = 15
SW result = 15
Success SW and HW results match
INFO: [SIM 1] CSim done with 0 errors.
INFO: [SIM 3] *************** CSIM finish ***************
```

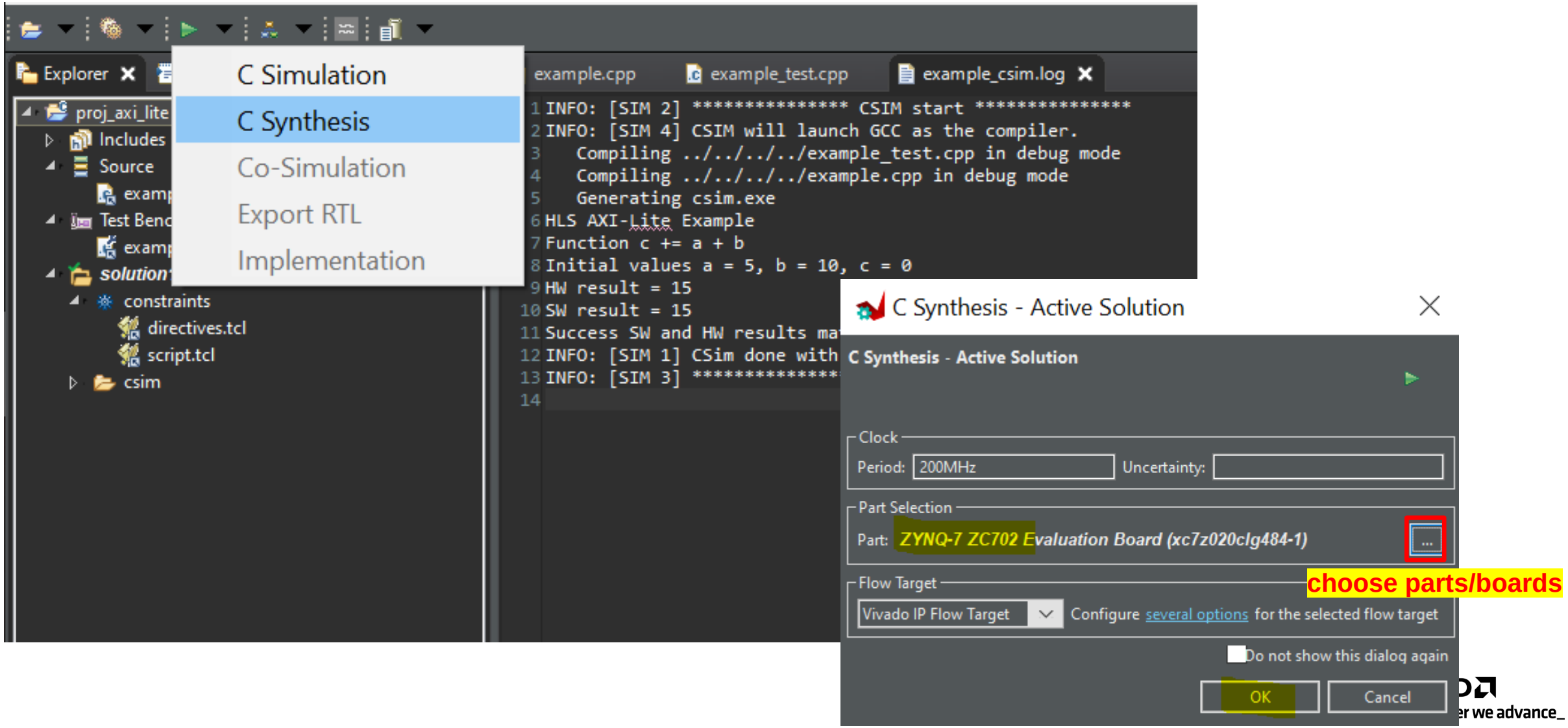
Console | Errors | Warnings | Guidance | Properties | Man Pages | Git Repositories | Modules/Loops

Vitis HLS Console

```
   Compiling ../../../../example_test.cpp in debug mode
   Compiling ../../../../example.cpp in debug mode
   Generating csim.exe
HLS AXI-Lite Example
Function c += a + b
Initial values a = 5, b = 10, c = 0
HW result = 15
SW result = 15
Success SW and HW results match
INFO: [SIM 211-1] CSim done with 0 errors.
INFO: [SIM 211-3] ************** CSIM finish **************
INFO: [HLS 200-111] Finished Command csim_design CPU user time: 0 seconds. CPU system time: 1 seconds. Elapsed time: 2.286 seconds; current allocated memory: 288.811 MB.
Finished C simulation.
```

Vitis HLS Design Flow

C synthesis ---> Convert C code to RTL code



Vitis HLS Design Flow

C synthesis ---> the result

example.cpp | example_test.cpp | example_csim.log | Synthesis Summary(solution1)

Synthesis Summary Report of 'example'

General Information

Date: Wed Nov 8 17:10:34 2023
Version: 2021.1_AR000033086_AR000033086_AR000033086 (Build 3247384 on Thu Jun 10 19:36:33 MDT 2021)
Project: proj_axi_lite

Solution: solution1 (Vivado IP Flow Target)
Product family: zynq
Target device: xc7z020-clg484-1

Timing Estimate

Target	Estimated	Uncertainty
5.00 ns	4.371 ns	1.35 ns

Performance & Resource Estimates

Modules | Loops

Modules && Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
example	Timing Violation			-0.72	2	10.000	-	3	-	no	0	0	351	444	0

HW Interfaces

S_AXILITE

Interface	Data Width	Address Width	Offset	Register
s_axi_BUS_A	32	6	24	0

TOP LEVEL CONTROL

Interface	Type	Ports
ap_clk	clock	ap_clk
ap_rst_n	reset	ap_rst_n
interrupt	interrupt	interrupt
ap_ctrl	ap_ctrl_hs	

SW I/O Information

Top Function Arguments

Argument	Direction	Datatype

Vitis HLS Design Flow

C synthesis ---> the result

example.cpp | example_test.cpp | example_csim.log | Synthesis Summary(solution1)

Synthesis Summary Report of 'example'

General Information

Date: Wed Nov 8 17:10:34 2023
Version: 2021.1_AR000033086_AR000033086_AR000033086 (Build 3247384 on Thu Jun 10 19:36:33 MDT 2021)
Project: proj_axi_lite

Solution: solution1 (Vivado IP Flow Target)
Product family: zynq
Target device: xc7z020-clg484-1

Timing Estimate

Target	Estimated	Uncertainty
5.00 ns	4.371 ns	1.35 ns

Performance & Resource Estimates

Modules Loops

Modules && Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
example	Timing Violation			-0.72	2	10.000	-	3	-	no	0	0	351	444	0

HW Interfaces

S_AXILITE

Interface	Data Width	Address Width	Offset	Register
s_axi_BUS_A	32	6	24	0

TOP LEVEL CONTROL

Interface	Type	Ports
ap_clk	clock	ap_clk
ap_rst_n	reset	ap_rst_n
interrupt	interrupt	interrupt
ap_ctrl	ap_ctrl_hs	

SW I/O Information

Top Function Arguments

Argument	Direction	Datatype

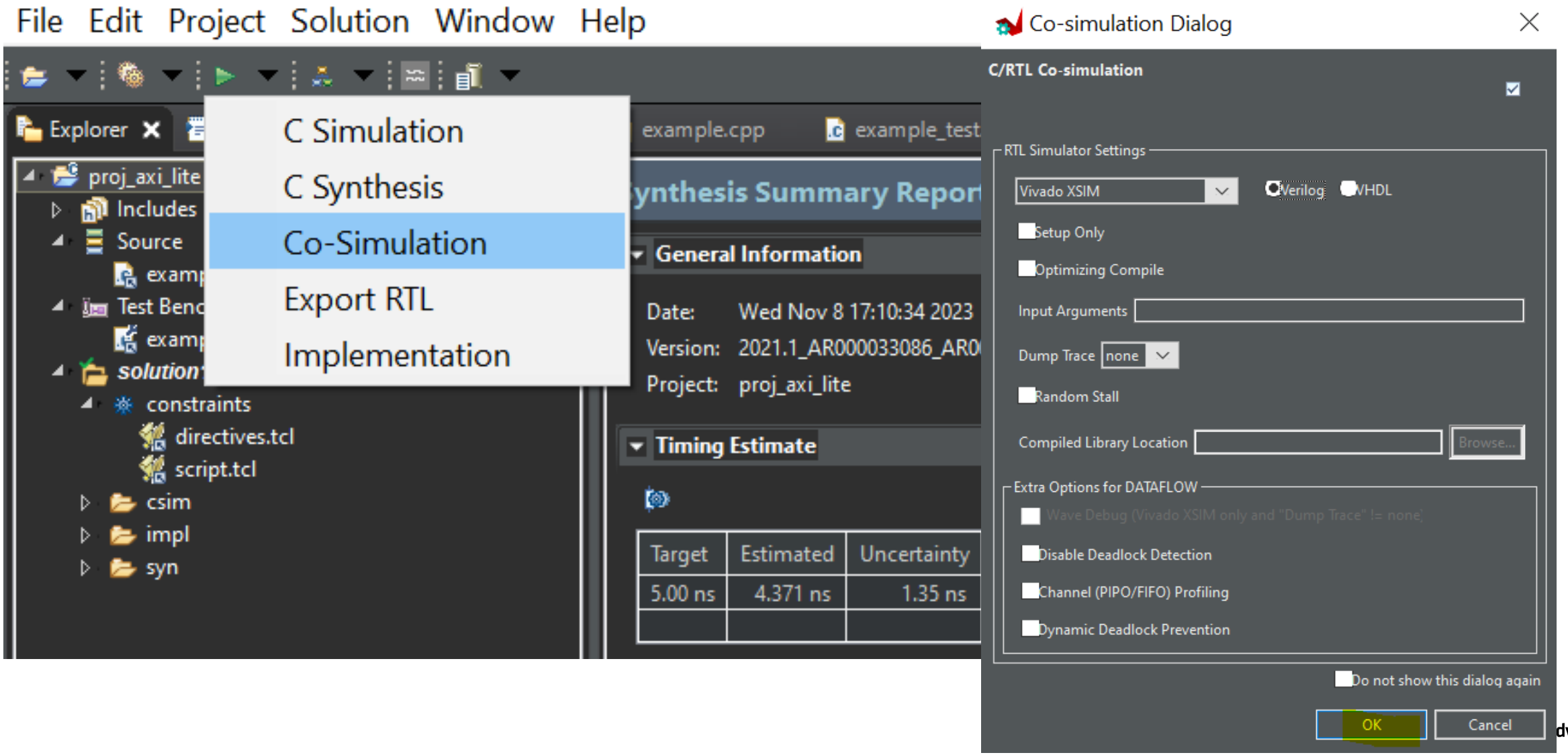
Console | Errors | Warnings | Guidance | Properties | Man Pages | Git Reposito

Vitis HLS Console

```
INFO: [HLS 200-111] Finished Creating RTL model: CPU user time: 0 seconds. CPU sy
INFO: [HLS 200-111] Finished Generating all RTL models: CPU user time: 0 seconds.
INFO: [HLS 200-111] Finished Updating report files: CPU user time: 1 seconds. CPU
INFO: [VHDL 208-304] Generating VHDL RTL for example.
INFO: [VLOG 209-307] Generating Verilog RTL for example.
INFO: [HLS 200-789] **** Estimated Fmax: 228.78 MHz
INFO: [HLS 200-111] Finished Command csynth_design CPU user time: 2 seconds. CPU
INFO: [HLS 200-112] Total CPU user time: 6 seconds. Total CPU system time: 2 seco
Finished C synthesis.
```

Vitis HLS Design Flow

C Co-Simulation ---> Co-simulation C with RTL code to verify the same behavior they have



Vitis HLS Design Flow

C Co-Simulation ---> Co-simulation C with RTL code to verify the same behavior they have

测试激励文件会为综合验证顶层函数输出， 如果输出正确， 则会向测试激励文件的main() 函数返回 0 值。 Vitis HLS 针对 C 语言仿真和 C/RTL 协同仿真使用相同的返回值， 以判定结果是否正确。如果C 语言测试激励文件返回非 0 值， Vitis HLS 就会报告仿真失败

Vitis HLS Design Flow

C Co-Simulation ---> the result

example.cpp | example_test.cpp | example_csim.log | Synthesis Summary(solution1) | Co-simulation Report(solution1)

Cosimulation Report for 'example'

General Information

Date: Wed Nov 8 17:20:33 TST 2023
Version: 2021.1_AR000033086_AR000033086_AR000033086 (Build 3247384 on Thu Jun 10 19:36:33 MDT 2021)
Project: proj_axi_lite
Status: Pass

Solution: solution1 (Vivado IP Flow Target)
Product family: zynq
Target device: xc7z020-clg484-1

Cosim Options

Tool: Vivado XSIM

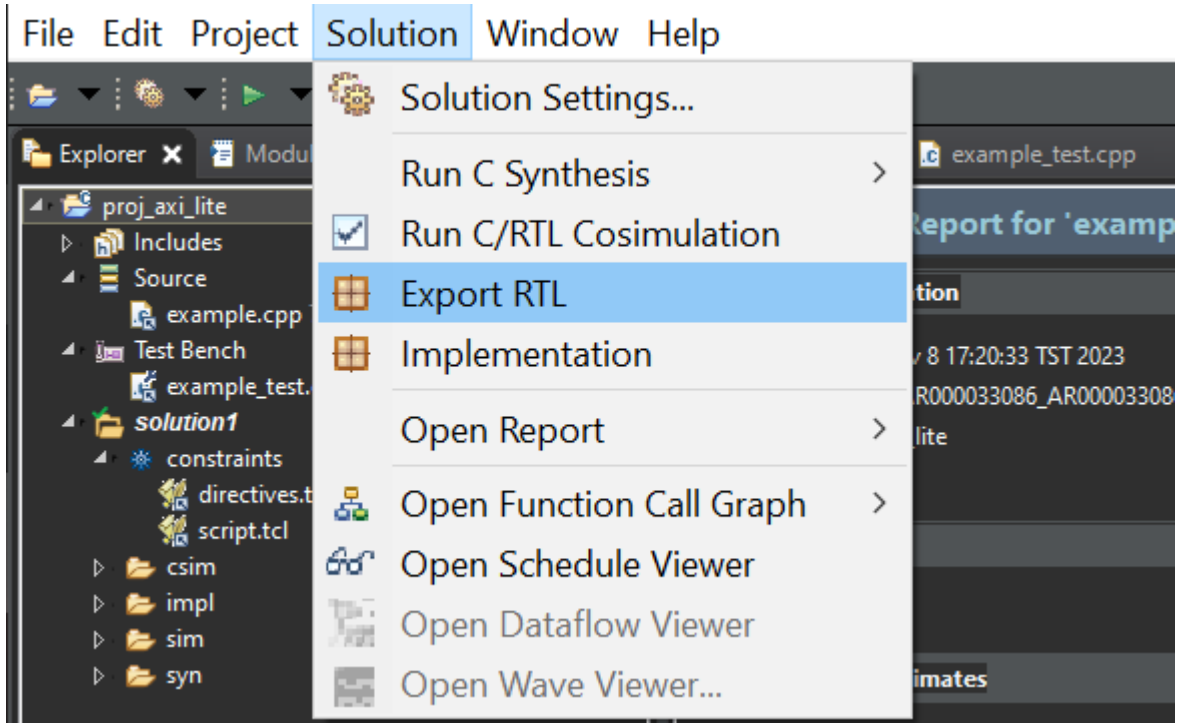
RTL: Verilog

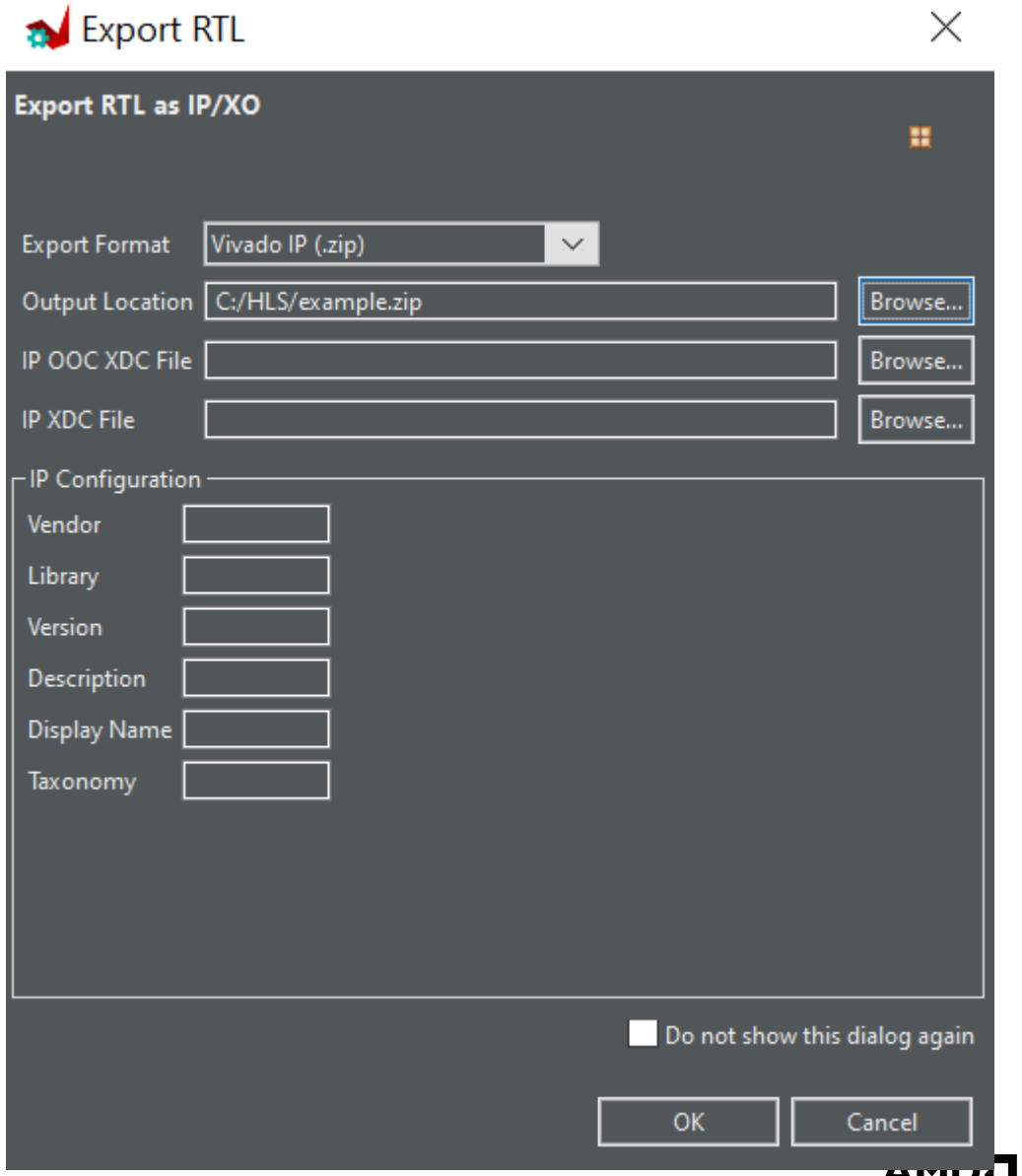
Performance Estimates

Modules	Avg II	Max II	Min II	Avg Latency	Max Latency	Min Latency
example				2	2	2

Vitis HLS Design Flow

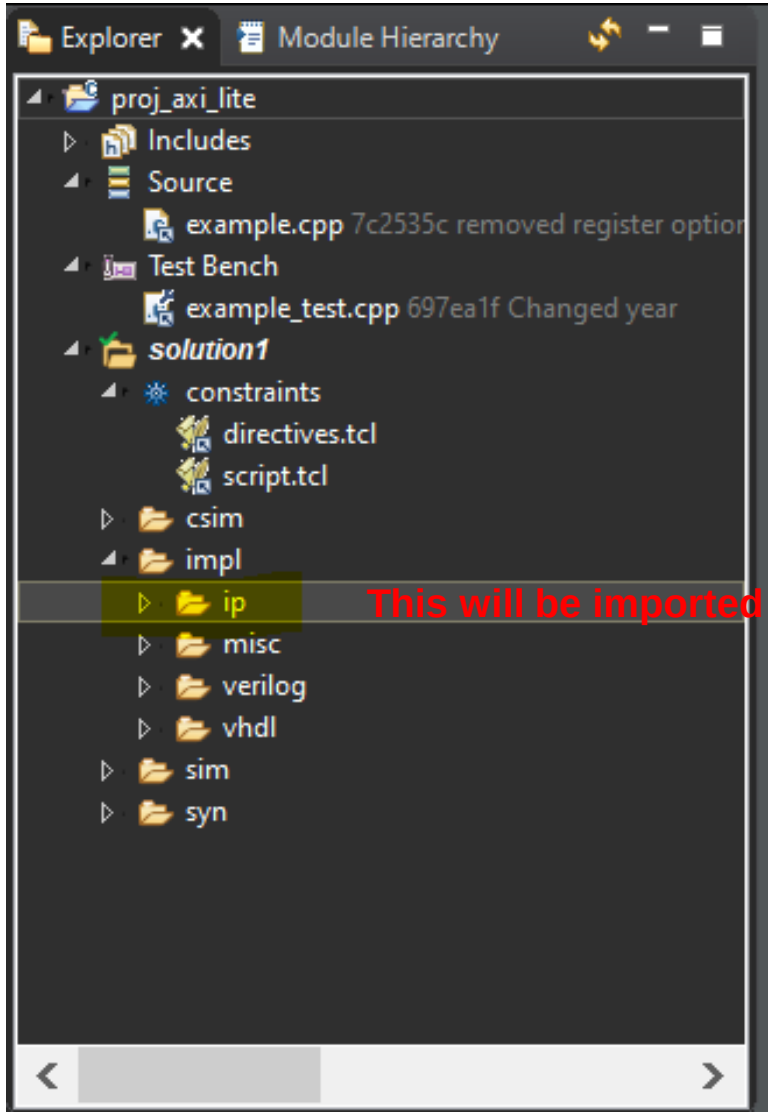
Export HLS IP





Vitis HLS Design Flow

Export HLS IP

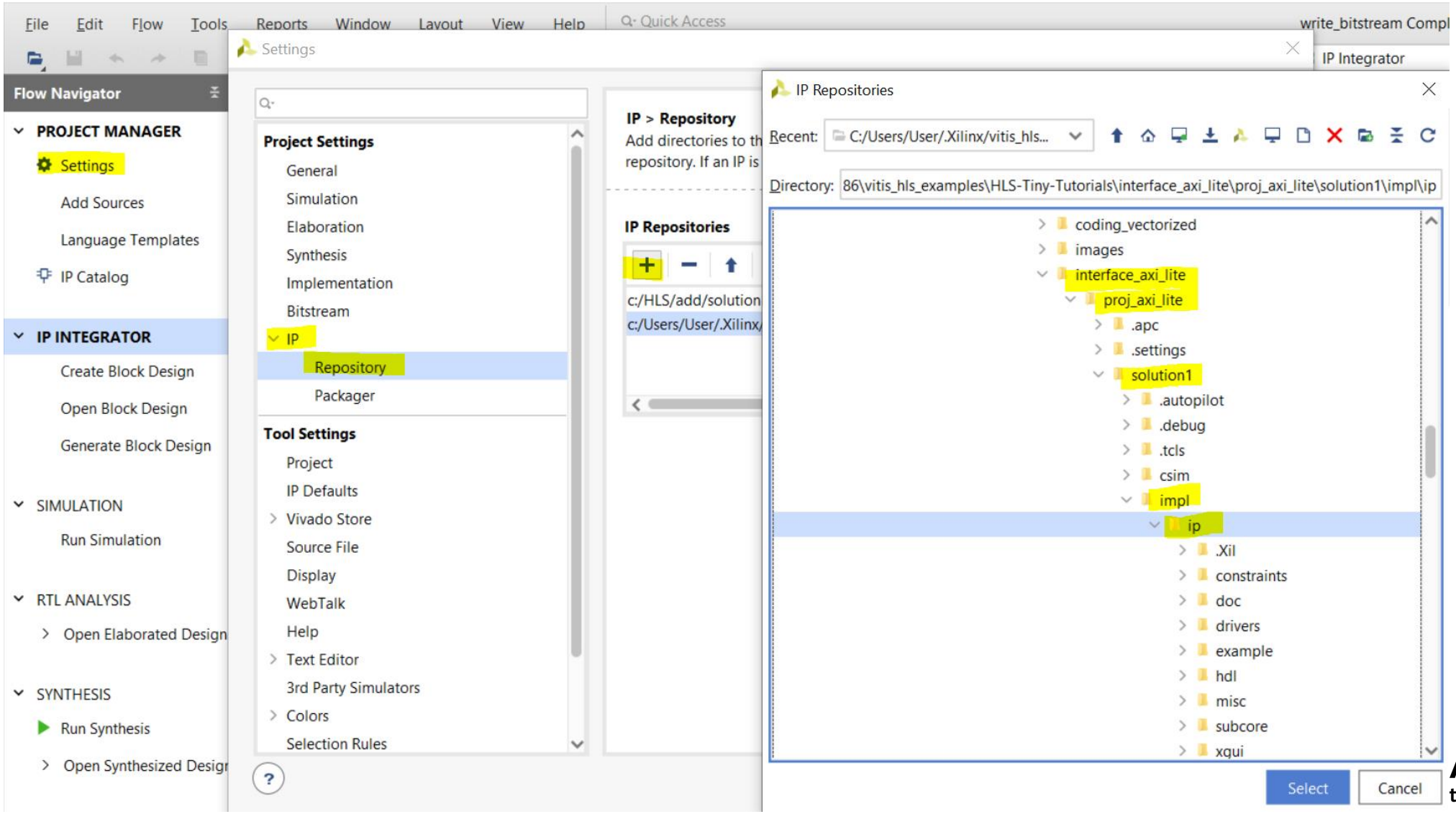


This will be imported to Vivado IP catalog

Vivado 2021.1 Part

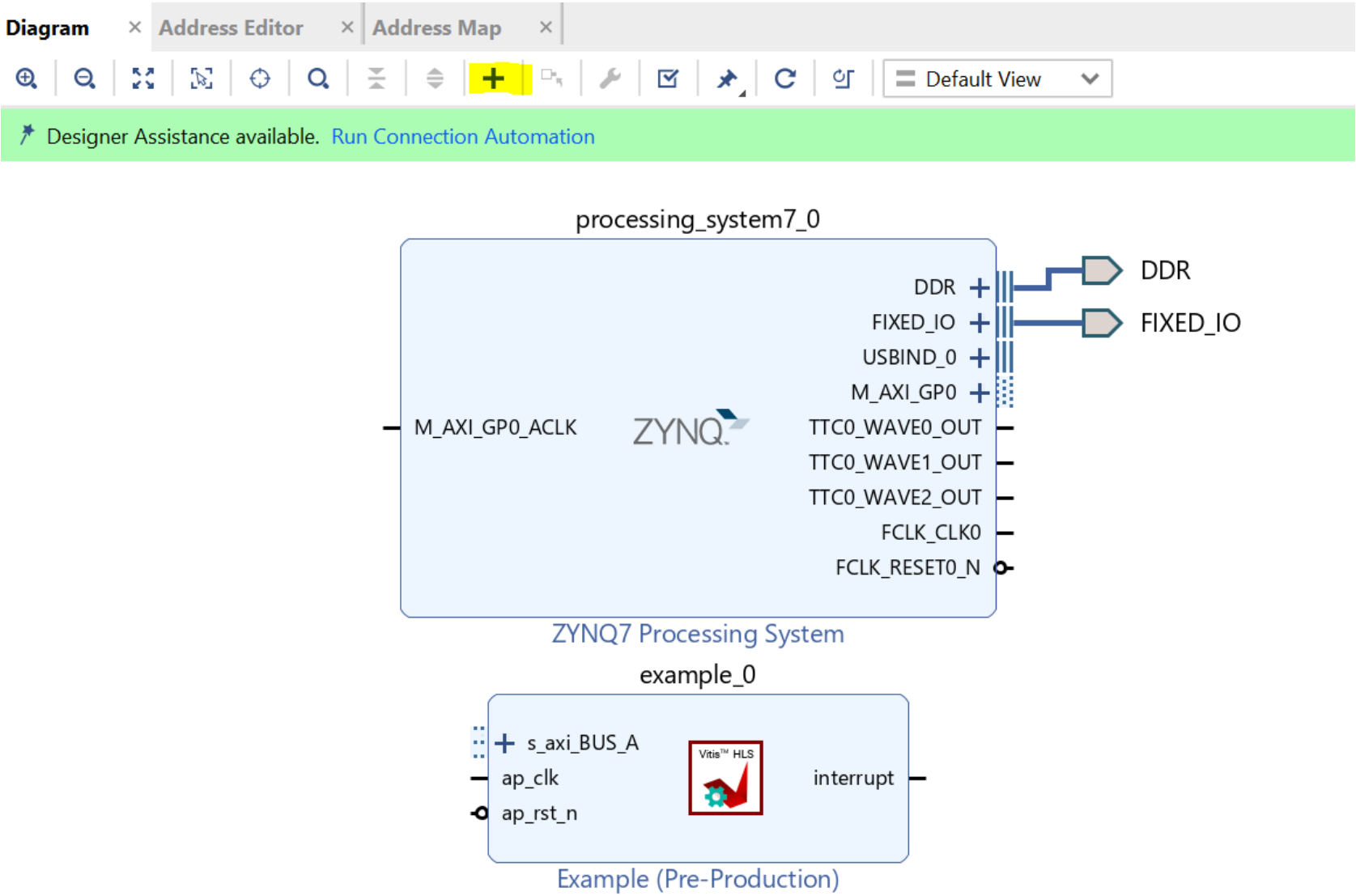
Vitis HLS Design Flow

Create project with ZC702 and add IP from Vitis HLS directory



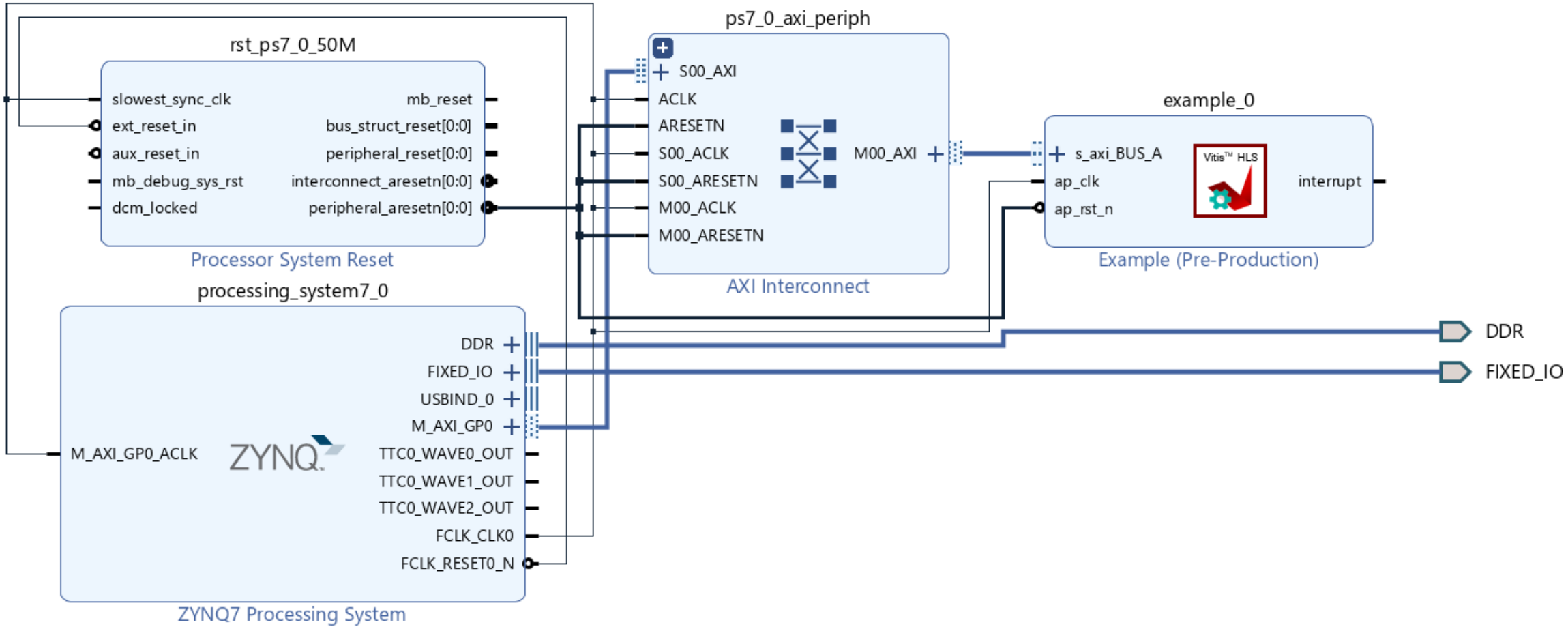
Vitis HLS Design Flow

Create Block Design and put two IP including HLS on it



Vitis HLS Design Flow

Run Connection Automation



Vitis HLS Design Flow

Generate HDL wrapper and bitstream
Generate .xsa finally

Vitis 2021.1 Part

Vitis HLS Design Flow

Create Platform from .xsa
Create HelloWord.c application and modify code to the following

Part 1

```
#include "platform.h"
#include "xbasic_types.h"
#include "xparameters.h" // Contains definitions for all peripherals
#include "xexample.h" // Contains hls example (axilite) IP macros and functions


// Define global values for HLS example IP
XExample do_hls_example;
XExample_Config *do_hls_example_cfg;
```

Vitis HLS Design Flow

Create Platform from .xsa
Create HelloWord.c application and modify code to the following

Part 2

```
// Initialize the HLS example IP
void init_HLS_example(){

        int status;
        // Create HLS example IP pointer
        do_hls_example_cfg = XExample_LookupConfig(
        XPAR_XEXAMPLE_0_DEVICE_ID);

        if (!do_hls_example_cfg) {
            xil_printf(
                    "Error loading configuration for do_hls_example_cfg \n\r");
        }


        status = XExample_CfgInitialize(&do_hls_example,
                do_hls_example_cfg);
        if (status != XST_SUCCESS) {
            xil_printf("Error initializing for do_hls_example \n\r");
        }

        XExample_Initialize(&do_hls_example,
        XPAR_XEXAMPLE_0_DEVICE_ID);
}
```

Vitis HLS Design Flow

Create Platform from .xsa
Create HelloWord.c application and modify code to the following

Part 3

```
// Function that adds using HLS example IP
// The functions used here are defined in xexample.h
void example_hls(int a, int b) {
    unsigned int c;
    c = 0; // result output from HLS IP

    // Write inputs
    XExample_Set_a(&do_hls_example, a);
    XExample_Set_b(&do_hls_example, b);
    xil_printf("Write a: %d \n\r", a);
    xil_printf("Write b: %d \n\r", b);

    // Start HLS IP
    XExample_Start(&do_hls_example);
    xil_printf("Started HLS Example IP \n\r");

    // Wait until it is finished
    while (!XExample_IsDone(&do_hls_example))
        ;

    // Get hls_multiplier returned value
    c = XExample_Get_return(&do_hls_example);

    xil_printf("HLS IP Return Value: %d\n\r", c);
    xil_printf("End of test\n\n\r");
}
```

Vitis HLS Design Flow

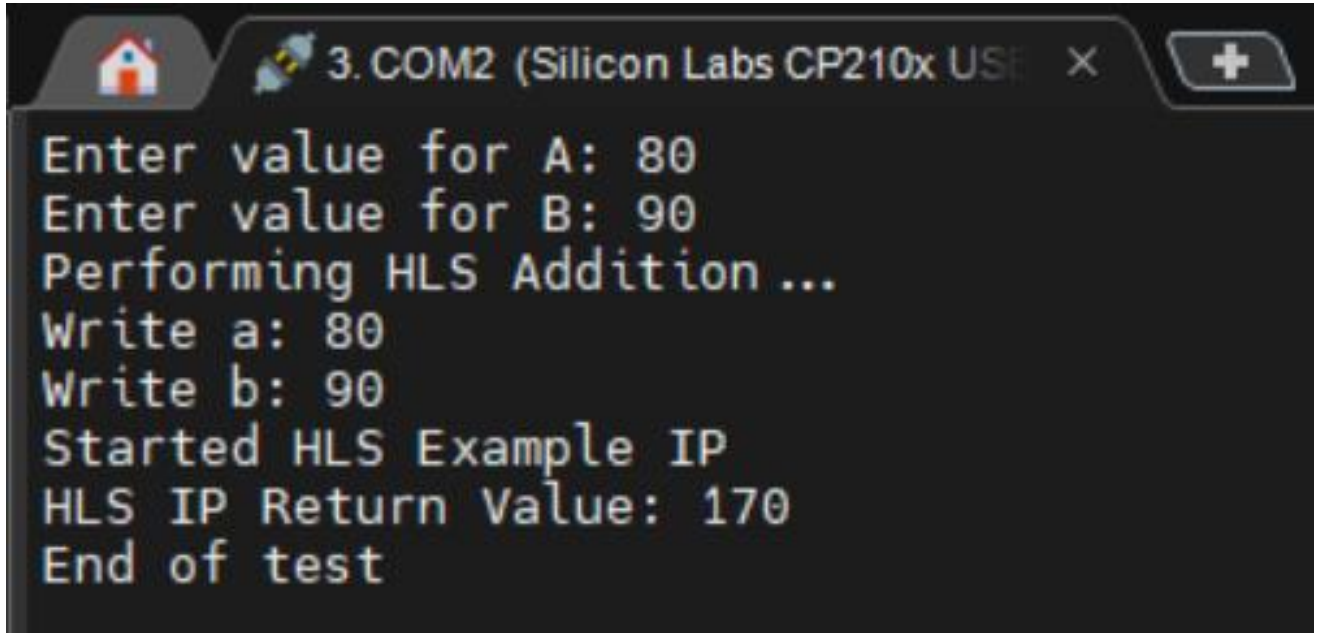
Create Platform from .xsa
Create HelloWord.c application and modify code to the following

Part 4

```
int main() {
    // setup
    init_platform();
    init_HLS_example();
    int a = 0;
    int b = 0;

    while (1) {
        // Enter the HLS IP inputs - a and b (Defined in Vitis HLS)
        xil_printf("Enter value for A: ");
        scanf("%d", &a);
        xil_printf("%d\n\r", a);
        xil_printf("Enter value for B: ");
        scanf("%d", &b);
        xil_printf("%d\n\r", b);

        xil_printf("Performing HLS Addition... \n\r");

        // perform addition in HLS IP
        example_hls(a, b);
    }

    cleanup_platform();
    return 0;
}
```

Vitis HLS Design Flow

Build and Run --- Result



Reference

English:

1. AXI Basics 6 - Introduction to AXI4-Lite in Vitis HLS (xilinx.com)
2. AXI Basics 7 - Connecting to the PS using AXI4-Lite and Vitis HLS (xilinx.com)

Chinese:

1. AXI 基础第 6 讲 - Vitis HLS 中的 AXI4-Lite 简介（第 1 部分）(xilinx.com)
2. AXI 基础第 7 讲 - 使用 AXI4-Lite 将 Vitis HLS 创建的 IP 连接到 PS (xilinx.com)
3. Vitis HLS 加法器(整数)设计_hls print-CSDN博客

AMD