

# Floating-point matrix multiplication accelerator

The tutorial goes through the process of:

1. creating an accelerator for matrix multiplication with **Vitis HLS**;
2. creating a platform that includes the accelerator with **Vivado**;
3. creating software to use the accelerator with **Vitis**.

This document is based on Xilinx application note XAPP1170, 2016.

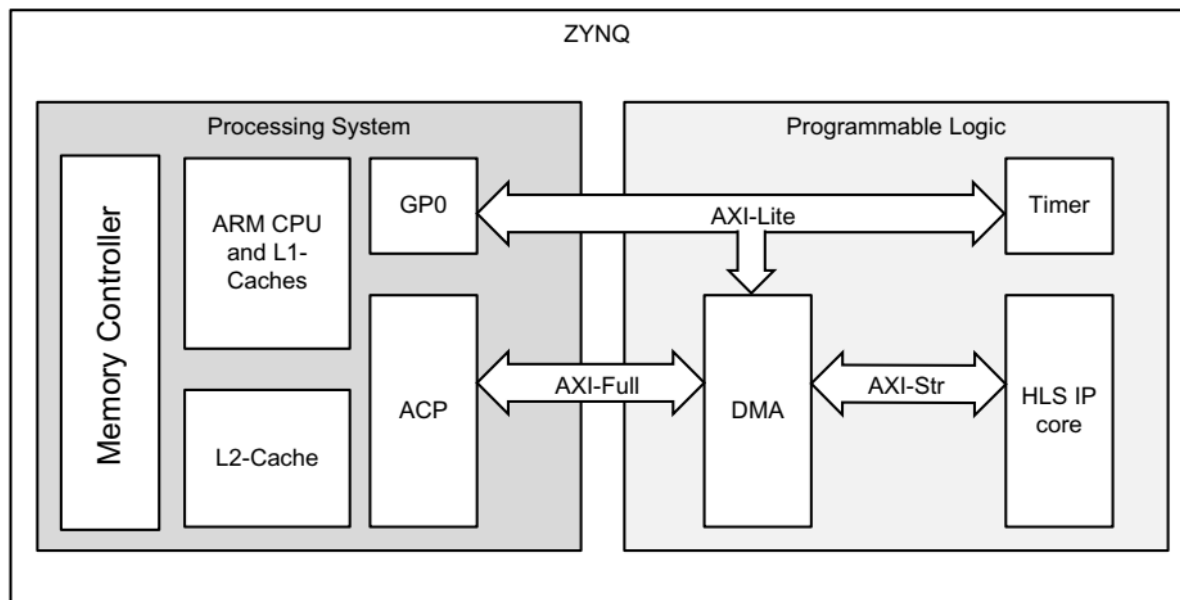
*A Zynq Accelerator for Floating Point Matrix Multiplication Designed with Vivado HLS*

*Authors: Daniele Bagni, A. Di Fresco, J. Noguera, F. M. Vallina*

Associated files:

- The file `mmult.h` contains several versions of the matrix multiplication code.
- The file `mmult_accel.cpp` contains the versions to be synthesized.
- The file `mmult_test.cpp` contains code to test synthesizable implementations.

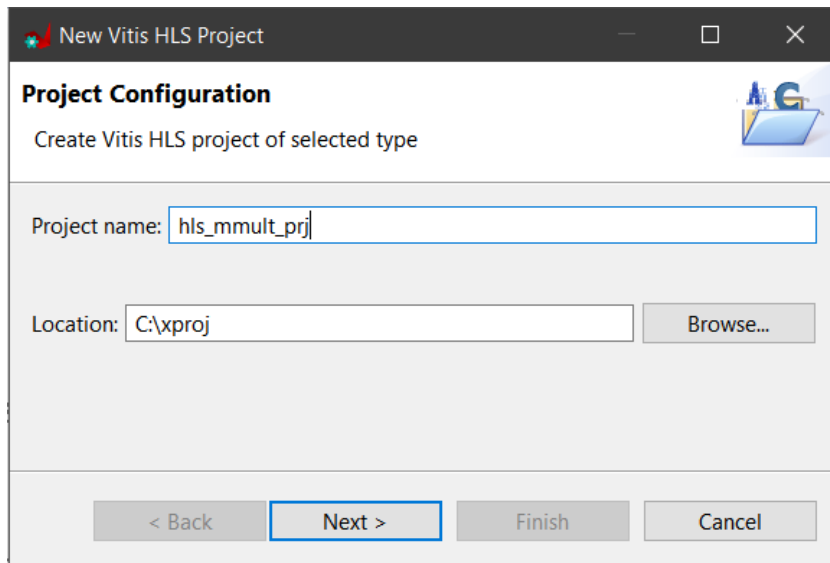
## Overall system organization



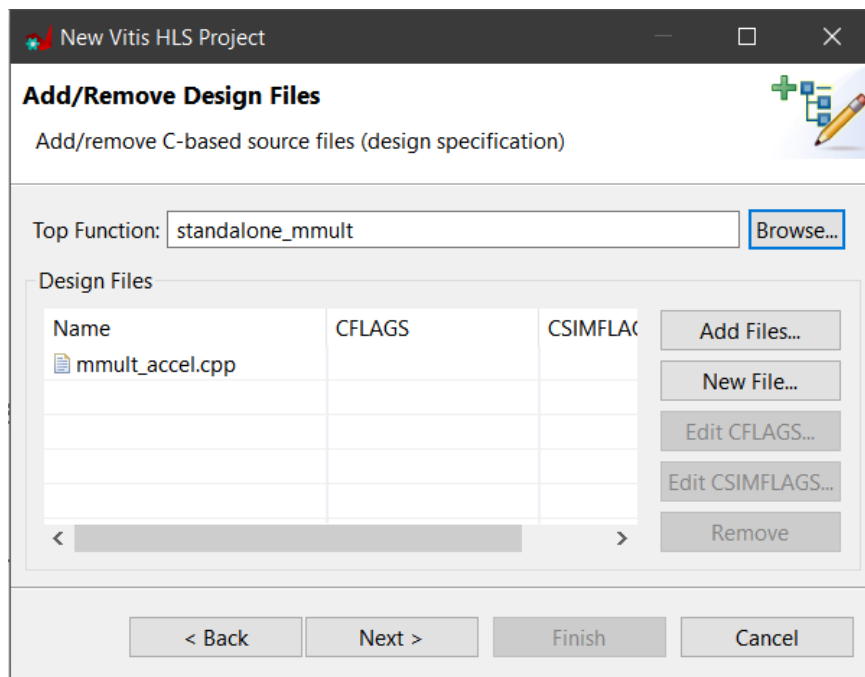
X15748-010316

## Part 1: High-Level Synthesis

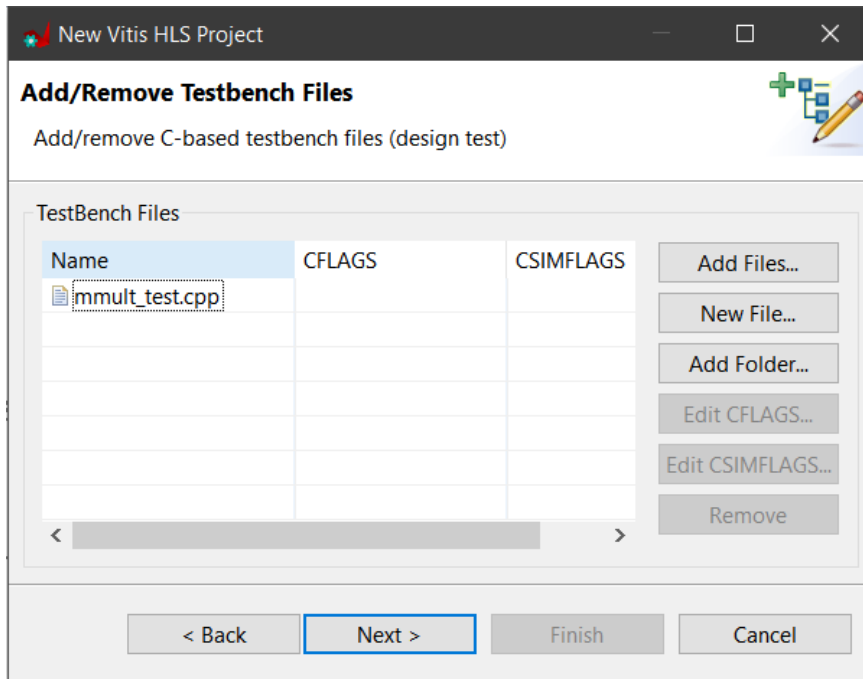
Create a new project named `hls_mmult_prj` in Vivado HLS.



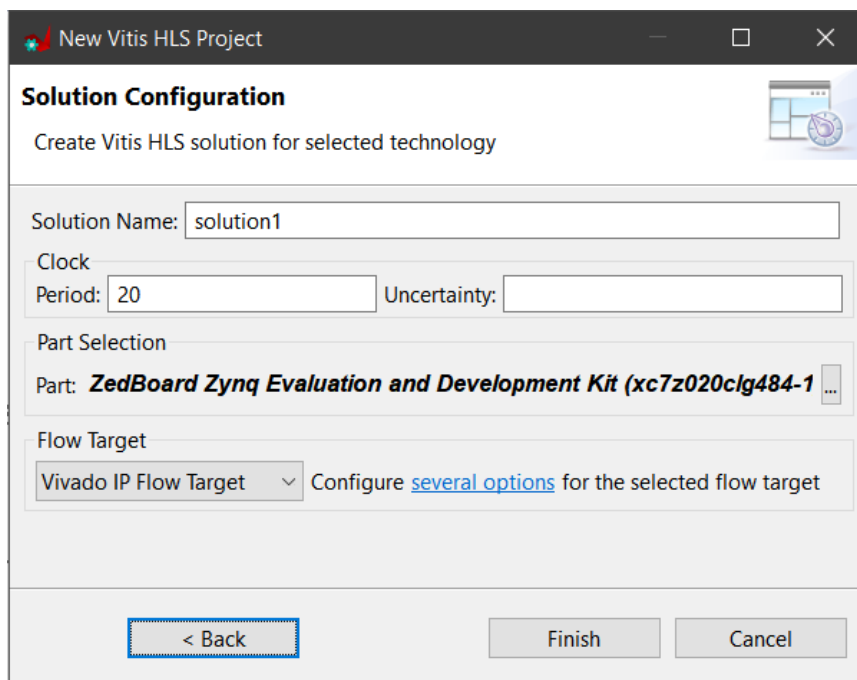
Add file `mmult_accel.cpp` and select function “`standalone_mmult`”.



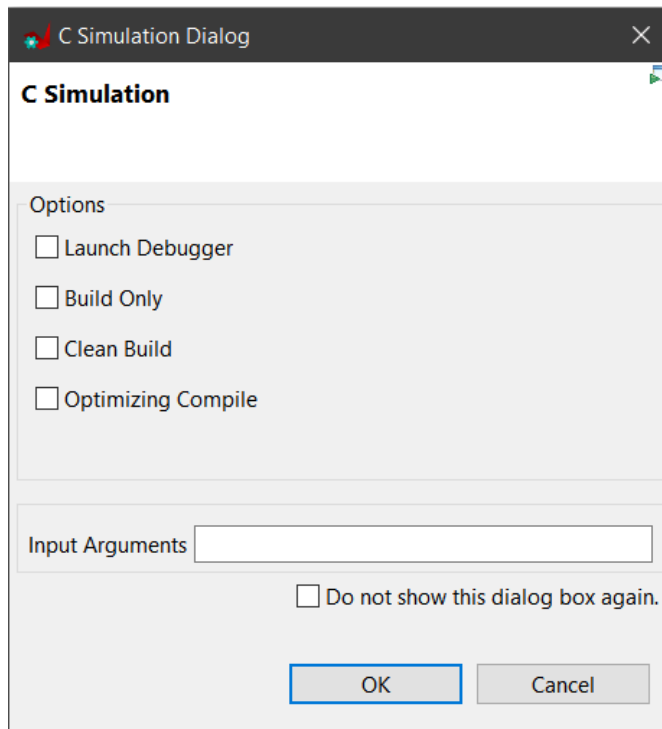
Add the testbench file `mmult_test.cpp` ...



Set clock period to 20 ns and choose ZedBoard ; Vivado IP Flow Target



Use Window → Show View → Flow Navigator to open the navigator.  
Run C Sim Simulation



You should get the messages ...

```
standalone_mmult_csim.log
1 INFO: [SIM 2] ***** CSIM start *****
2 INFO: [SIM 4] CSIM will launch GCC as the compiler.
3   Compiling ../../../../Users/jcf/Desktop/xapp1170/xapp1170_2015v4/empty/hls/mmult_test.cpp in debug
4   Compiling ../../../../Users/jcf/Desktop/xapp1170/xapp1170_2015v4/empty/hls/mmult_accel.cpp in debug
5   Generating csim.exe
6 NORMAL MODE
7
8 Matrixes identical ... Test successful!
9
10 INFO: [SIM 1] CSim done with 0 errors.
11 INFO: [SIM 3] ***** CSIM finish *****
12
```

Run C Synthesis.

The target implementation is (mmult.h):

```
// matrix multiplication of a A*B matrix
L1:for (int ia = 0; ia < DIM; ++ia)
  L2:for (int ib = 0; ib < DIM; ++ib)
  {
    // #pragma HLS PIPELINE II=1
    T sum = 0;
    L3:for (int id = 0; id < DIM; ++id)
      sum += a[ia][id] * b[id][ib];
    out[ia][ib] = sum;
  }

return;
```

and the results are

Timing Estimate

Target	Estimated	Uncertainty
20.00 ns	13.787 ns	5.40 ns

Performance & Resource Estimates ⓘ

☒ Modules
☒ Loops
☐ BR
☐ %

Modules & Loops	Issue Type	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BR
standalone_mmult	II Violation ⓘ	-	16504	3.300E5	-	16505	-	no	
L1_L2	II Violation ⓘ	-	16502	3.300E5	135	16	1024	yes	

HW Interfaces

And

Console

Errors

Warnings

Guidance ⓘ

Man Pages

☒ 14 Guidance-Infos
☒ 1 Guidance-Warnings
☒ 0 Guidance-Errors

Name	Web Help	Details
<div>All Categories</div> <div> <div>SCHEDULE</div> <div> <div>[HLS 200-885]</div> <div>[HLS 200-1470]</div> </div> </div> <div> <div>RUNTIME</div> <div>LOOP</div> <div> <div>THROUGHPUT</div> <div>[HLS 200-789]</div> </div> </div>	<div>LINK</div>	<div>Unable to schedule 'load' operation ('A_load_2', ../Users/jcf/Desktop/xapp1170/xs</div> <div>Pipelining result : Target II = 1, Final II = 16, Depth = 135, loop 'L1_L2'</div> <div>**** Estimated Fmax: 72.53 MHz</div>

Check also the messages that appear in the console:

```
INFO: [XFORM 203-510] Pipelining loop 'L2'
(../Users/jcf/Desktop/xapp1170/xapp1170_2015v4/empty/hls/mmult.h:57) in function
'standalone_mmult' automatically.
```

```

INFO: [XFORM 203-502] Unrolling all sub-loops inside loop 'L2'
(../Users/jcf/Desktop/xapp1170/xapp1170_2015v4/empty/hls/mmult.h:57) in function
'standalone_mmult' for pipelining.
INFO: [HLS 200-489] Unrolling loop 'L3'
(../Users/jcf/Desktop/xapp1170/xapp1170_2015v4/empty/hls/mmult.h:57) in function
'standalone_mmult' completely with a factor of 32.
[...]
INFO: [XFORM 203-541] Flattening a loop nest 'L1'
(../Users/jcf/Desktop/xapp1170/xapp1170_2015v4/empty/hls/mmult.h:53:14) in function
'standalone_mmult'.
[...]
INFO: [SCHED 204-61] Pipelining loop 'L1_L2'.
WARNING: [HLS 200-885] Unable to schedule 'load' operation ('A_load_2',
../Users/jcf/Desktop/xapp1170/xapp1170_2015v4/empty/hls/mmult.h:53) on array 'A'
due to limited memory ports. Please consider using a memory core with more ports or
partitioning the array 'A'.
INFO: [HLS 200-1470] Pipelining result : Target II = 1, Final II = 16, Depth = 135,
loop 'L1_L2'

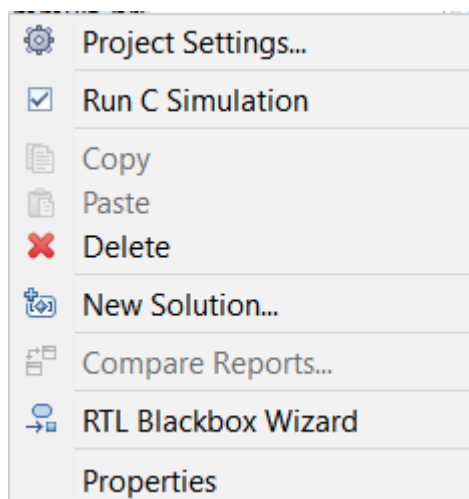
```

Notice that the synthesizer:

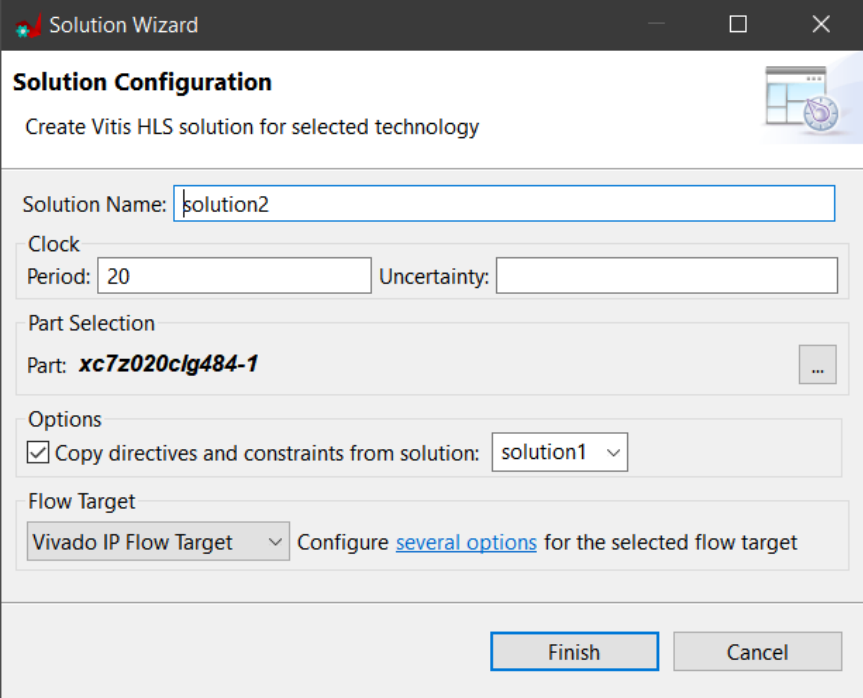
- automatically unrolled loop L3 completely;
- flattened loop nest L1 (L1\_L2 above);
- applied pipelining to L1\_L2 with II=1, but failed to meet that constraint.

We will solve the problem by partitioning the input arrays. The appropriate directives are already in the code as comments, but we will insert them in a new solution (solution2)

Right-click on the hls\_mmult\_prj project in the explorer and choose “New Solution...”



Be sure to copy directives and constraints from the previous solution.

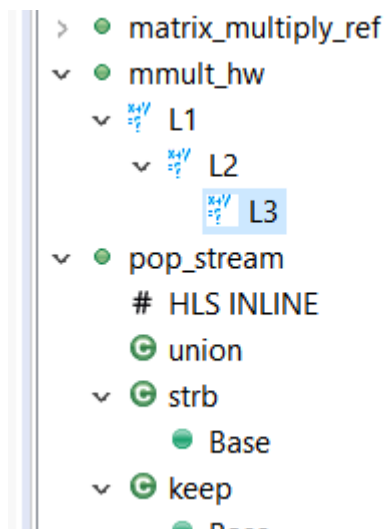


The image shows the 'Solution Wizard' dialog box, specifically the 'Solution Configuration' tab. The title bar says 'Solution Wizard'. Below the title bar, the tab is labeled 'Solution Configuration' with a sub-header 'Create Vitis HLS solution for selected technology'. The dialog contains several sections: 'Solution Name' with a text box containing 'solution2'; 'Clock' with 'Period' set to '20' and an empty 'Uncertainty' box; 'Part Selection' with 'Part' set to 'xc7z020c1g484-1'; 'Options' with a checked box for 'Copy directives and constraints from solution:' and a dropdown menu set to 'solution1'; and 'Flow Target' with a dropdown set to 'Vivado IP Flow Target' and a link to 'Configure several options for the selected flow target'. At the bottom are 'Finish' and 'Cancel' buttons.

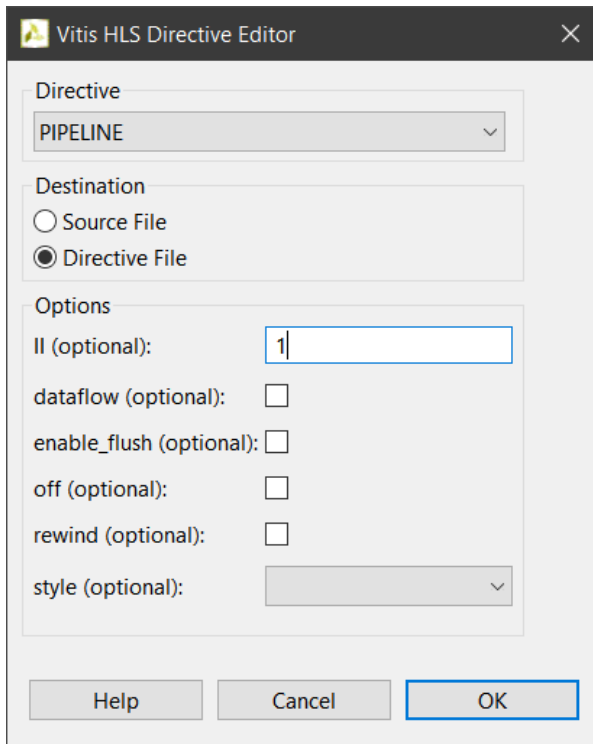
Notice that solution2 is now the active solution (modifications of constraints and directives are limited to the current solution).

Choose the “Directive” tabs in the top right window.

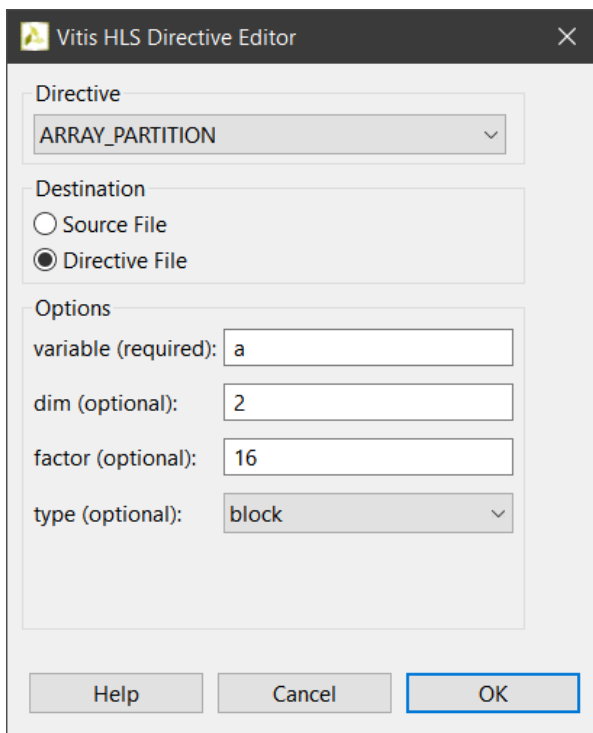
We are interested in the mmult\_hw function.



Select L2 and right-click to insert a directive:



Notice that folder “constraints” in solution2 now contains a “directives.tcl” file. The addition of this directive should lead to no change in this case, because the system already tries to pipeline the loop (as seen before). We will add two more directives to the function. Select `mmult_hw`, right-click and insert:



and



**Vitis HLS Directive Editor**

Directive: **ARRAY\_PARTITION**

Destination:   
☐ Source File   
☒ Directive File

Options:   
 variable (required): **b**   
 dim (optional): **1**   
 factor (optional): **16**   
 type (optional): **block**

Help Cancel OK

## Run C Synthesis

**Timing Estimate**

Target	Estimated	Uncertainty
20.00 ns	13.787 ns	5.40 ns

**Performance & Resource Estimates**

☒ Modules ☒ Loops ☐ Issues ☐ Warnings ☐ Errors ☐ Info ☐ Help

Modules & Loops	Issue Type	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT
standalone_mmult		-	1157	2.314E4	-	1158	-	no	0	160	16216	23627
mmult_hw_float_32_s		-	1156	2.312E4	-	1156	-	no	0	160	16213	23613
L1_L2		-	1154	2.308E4	132	1	1024	yes	-	-	-	-

Notice that the synthesis completes without constraint violations.

Create a new solution (solution3) to attempt to create an accelerator that runs at 100 MHz.

**Solution Wizard**

### Solution Configuration

Create Vitis HLS solution for selected technology

Solution Name:

Clock  
Period:  Uncertainty:


Part Selection  
Part: **xc7z020clg484-1** ...

Options  
☒ Copy directives and constraints from solution:  ▾















Flow Target  
 ▾ Configure [several options](#) for the selected flow target

**Finish** **Cancel**

Run C Synthesis. It completes successfully.

Timing Estimate									
									
Target	Estimated	Uncertainty							
10.00 ns	7.256 ns	2.70 ns							

Performance & Resource Estimates 									
    % <input checked="" type="checkbox"/> Modules <input checked="" type="checkbox"/> Loops    									
Modules & Loops	Issue Type	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	Est. Cycles
<div>   standalone_mmult </div>		-	1192	1.192E4	-	1193	-	no	
<div>   mmult_hw_float_32_s </div>		-	1191	1.191E4	-	1191	-	no	
<div>  L1_L2 </div>		-	1189	1.189E4	167	1	1024	yes	

Compare Latency and resources between solution2 and solution3.

We now have an accelerator that could, theoretically, receive new data on each clock cycle. However, the accelerator is not usable because of its interface.

How many bits has the interface?

HW Interfaces		
AP_MEMORY		
Interface	Bitwidth	
A_0_address0	6	
A_0_address1	6	
A_0_q0	32	
A_0_q1	32	
A_10_address0	6	
A_10_address1	6	
A_10_q0	32	
A_10_q1	32	
A_11_address0	6	
A_11_address1	6	
A_11_q0	32	
A_11_q1	32	
A_12_address0	6	
A_12_address1	6	
A_12_q0	32	
A_12_q1	32	
A_13_address0	6	
A_13_address1	6	

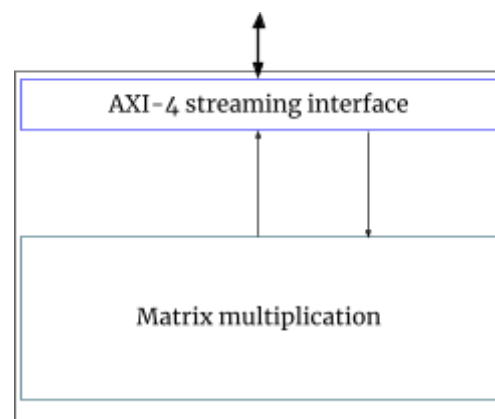
Too many ...

For achieving high performance with a feasible interface, some modifications are required.

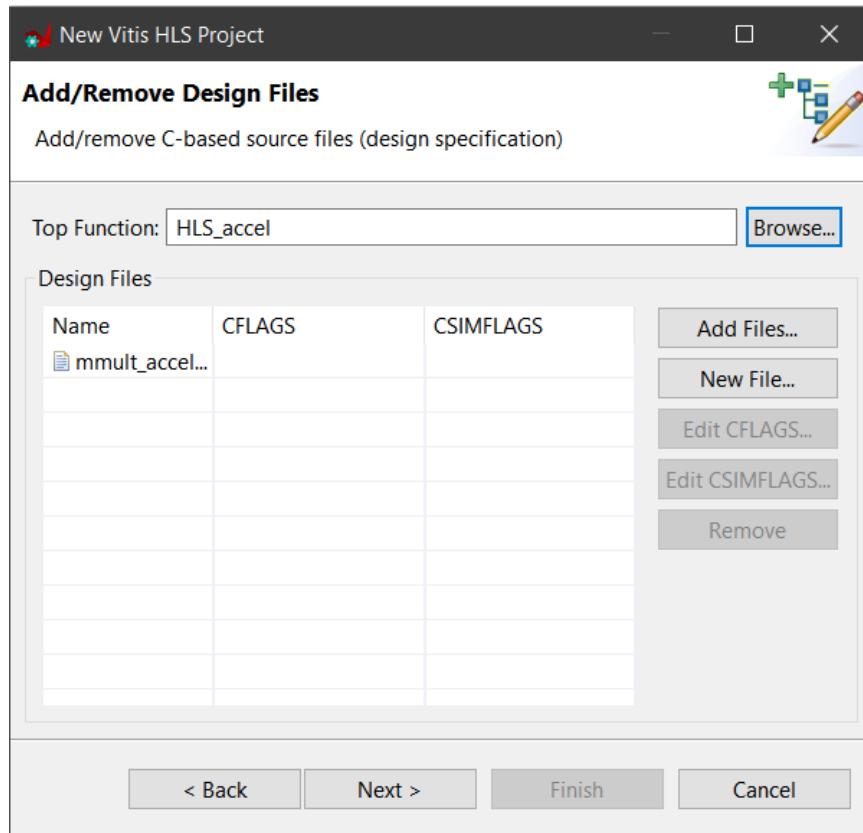
## Part 2: High-Level Synthesis - Streaming interface

This project will create an accelerator with a **streaming interface** that conforms to the AXI-4 bus protocol.

In this project, the accelerator consists of the mmult module “wrapped” inside an interface block.



Create a new project named `hls_wrapped_mmult_prj` in Vivado HLS. Use the same files as before, but use a different top level function, `HLS_accel()`. Add the flag “-DDB\_DEBUG” to all the C++ files.



DESIGN FILES		
Name	CFLAGS	CSIMFLAGS
mmult_test.cpp	-DDB_DEBUG	

Run C Simulation. You should get a message indicating success.

DEBUGGING AXI4 STREAMING DATA TYPES!

Matrixes identical ... Test successful!

INFO: [SIM 1] CSim done with 0 errors.

INFO: [SIM 3] \*\*\*\*\* CSIM finish \*\*\*\*\*

(Try to find out what the test code does.)

The top-level function is:

```
void HLS_accel (AXI_VAL INPUT_STREAM[2*MCR_SIZE], AXI_VAL OUTPUT_STREAM[MCR_SIZE])
{
```

```

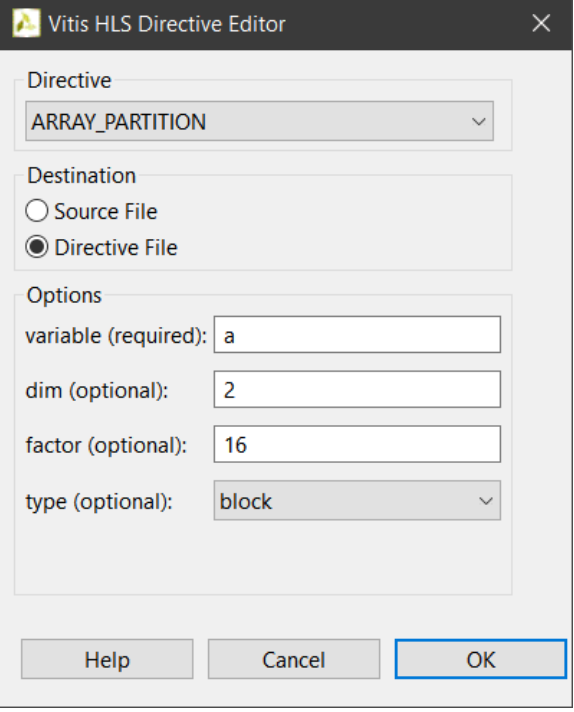
#pragma HLS INTERFACE s_axilite port=return      bundle=CONTROL_BUS
#pragma HLS INTERFACE axis      port=OUTPUT_STREAM
#pragma HLS INTERFACE axis      port=INPUT_STREAM
    wrapped_mmult_hw <float, 32, 32*32, 4, 5, 5>(INPUT_STREAM, OUTPUT_STREAM);
    return;
}

```

Read the “wrapped\_mmult\_hw()” and try to understand what is going on.  
Notice that:

- Some pipeline pragmas are already inserted in the code.
- The original hardware module function is called on line 155.

As previously, select mmult\_hw() in the directive window and insert the two directives:



Vitis HLS Directive Editor

Directive: ARRAY\_PARTITION

Destination: ☐ Source File ☒ Directive File

Options:

variable (required): a

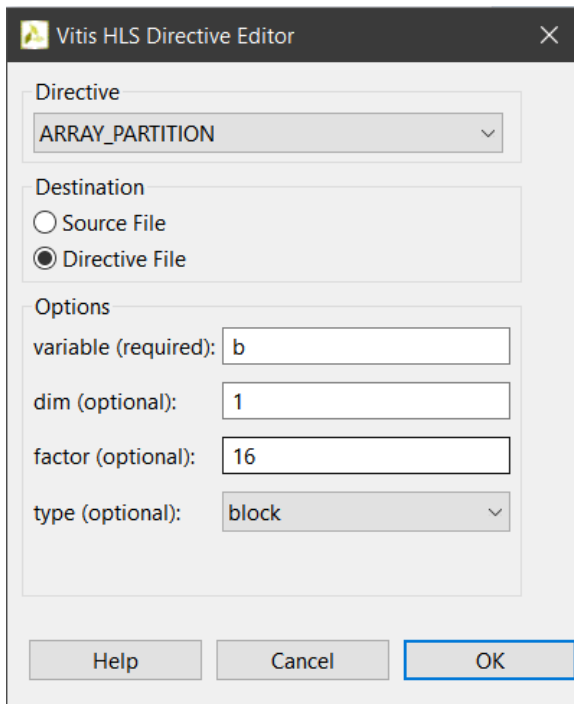
dim (optional): 2

factor (optional): 16

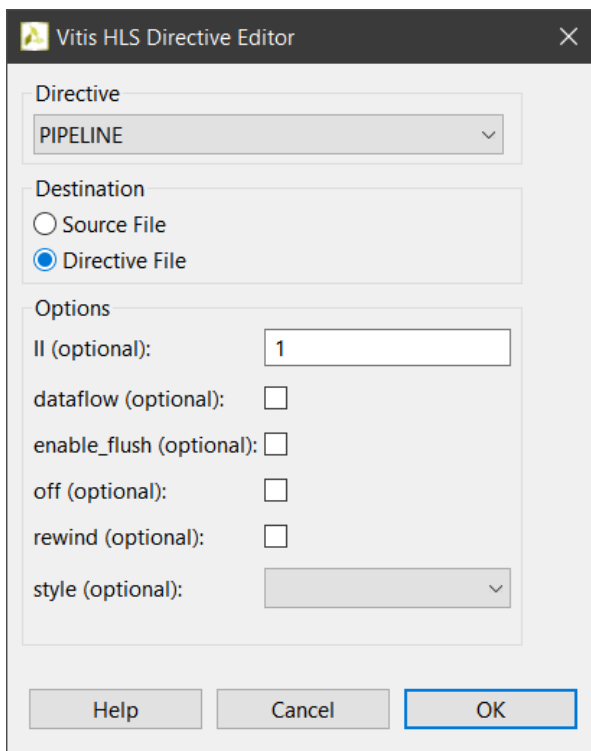
type (optional): block

Help Cancel OK

and



Select loop L2 in `mmult_hl()` and set the directive:

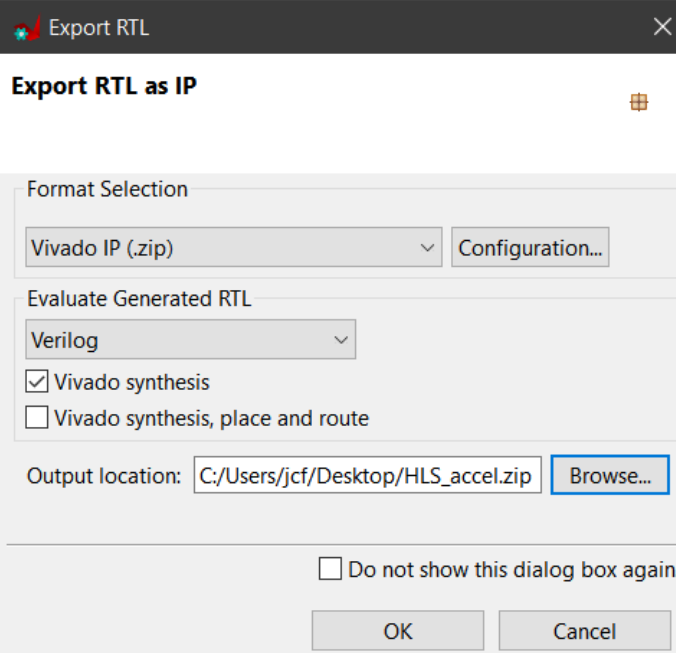


Run C Synthesis.

The synthesis step should run successfully. Inspect the results (including the information about the interface).

▼ SCHEDULE		
i [HLS 200-1470]		Pipelining result : Target II = 1, Final II = 1, Depth = 167, loop 'L1_L2'
i [HLS 200-1470]		Pipelining result : Target II = 1, Final II = 1, Depth = 1, loop 'VITIS_LOOP_136_1_VITIS_LOOP_137_2'
i [HLS 200-1470]		Pipelining result : Target II = 1, Final II = 1, Depth = 2, loop 'VITIS_LOOP_146_3_VITIS_LOOP_147_4'
i [HLS 200-1470]		Pipelining result : Target II = 1, Final II = 1, Depth = 4, loop 'VITIS_LOOP_158_5_VITIS_LOOP_159_6'

On the toolbar, select the green arrow and choose “Export RTL”. Take note of where you store the IP.



The dialog box titled "Export RTL as IP" contains the following elements:

- Format Selection:** A dropdown menu set to "Vivado IP (.zip)" and a "Configuration..." button.
- Evaluate Generated RTL:** A dropdown menu set to "Verilog". Below it are two checkboxes: "Vivado synthesis" (checked) and "Vivado synthesis, place and route" (unchecked).
- Output location:** A text field containing "C:/Users/jcf/Desktop/HLS\_accel.zip" and a "Browse..." button.
- Do not show this dialog box again:** An unchecked checkbox.
- Buttons:** "OK" and "Cancel" buttons at the bottom.

This will call Vivado to synthesize the block (takes some time) and include the result in the IP package.

#### Resource Usage

	Verilog
SLICE	0
LUT	10658
FF	12101
DSP	160
BRAM	66
SRL	436

#### Final Timing

	Verilog
CP required	10.000
CP achieved post-synthesis	8.276

Timing met

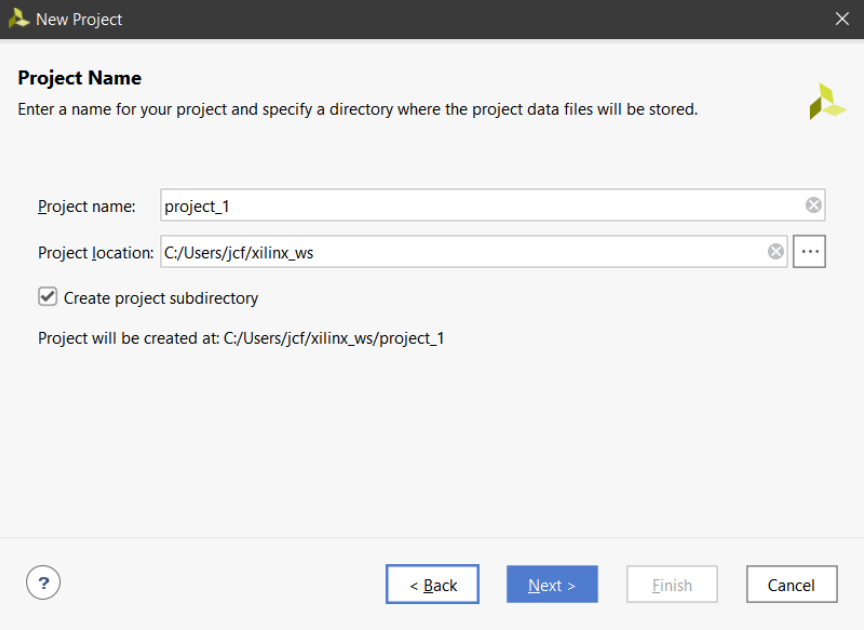
Export the report(.html) using the [Export Wizard](#)

## Part 3: Hardware platform design

We will use Vivado to create the hardware platform, which will include (see figure in the introduction):

1. A timer;
2. Our accelerator;
3. A Direct-Memory-Access controller.

Run Vivado 2020.02 and create a new project.



**New Project**

**Project Name**  
Enter a name for your project and specify a directory where the project data files will be stored.

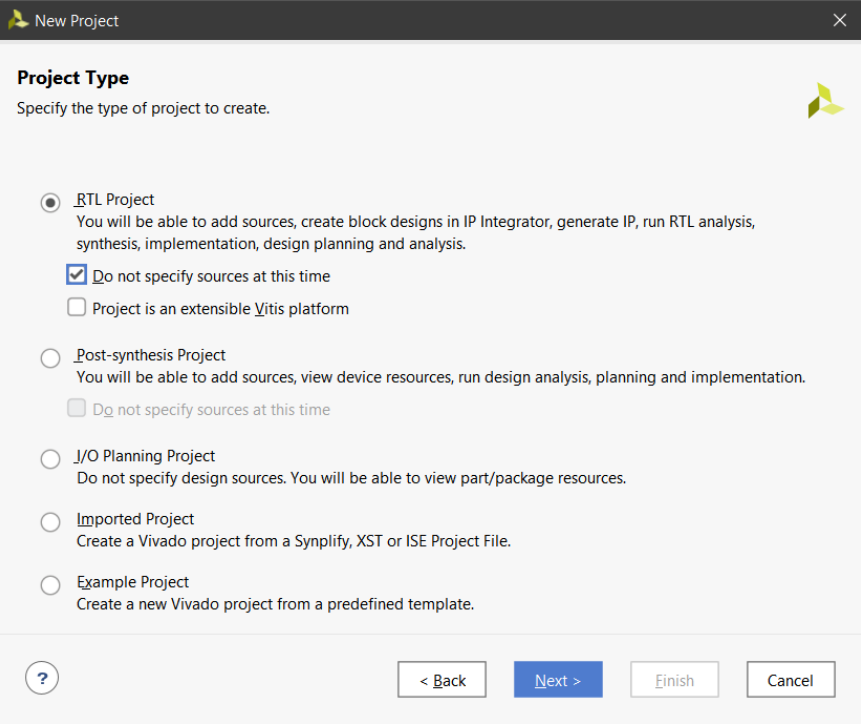
Project name:

Project location:

☒ Create project subdirectory

Project will be created at: C:/Users/jcf/xilinx\_ws/project\_1





**New Project**

**Project Type**  
Specify the type of project to create.

☒ **RTL Project**  
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.  
☒ Do not specify sources at this time  
☐ Project is an extensible Vitis platform

☐ **Post-synthesis Project**  
You will be able to add sources, view device resources, run design analysis, planning and implementation.  
☐ Do not specify sources at this time

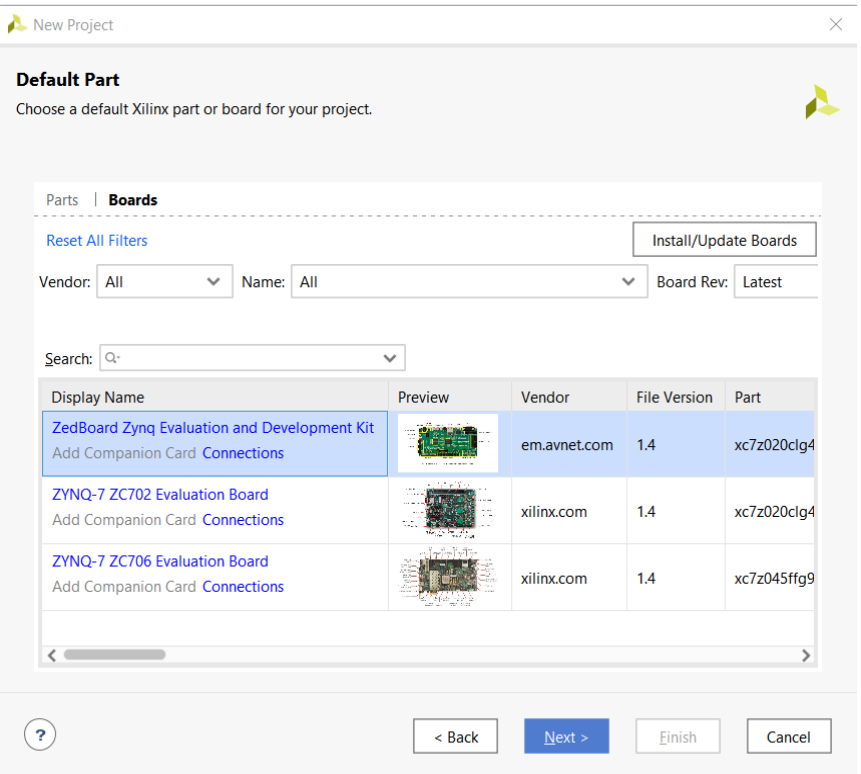
☐ **J/O Planning Project**  
Do not specify design sources. You will be able to view part/package resources.

☐ **Imported Project**  
Create a Vivado project from a Synplify, XST or ISE Project File.

☐ **Example Project**  
Create a new Vivado project from a predefined template.

? < Back Next > Finish Cancel

Select the ZedBoard:



**New Project**




**Default Part**  
Choose a default Xilinx part or board for your project.

Parts | **Boards**

Reset All Filters Install/Update Boards

Vendor: All Name: All Board Rev: Latest

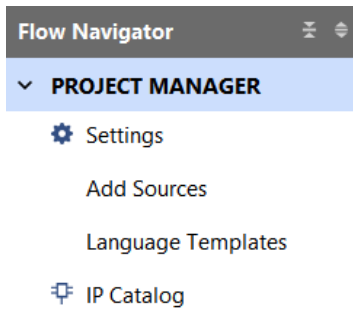
Search: Q-

Display Name	Preview	Vendor	File Version	Part
ZedBoard Zynq Evaluation and Development Kit Add Companion Card <a href="#">Connections</a>		em.avnet.com	1.4	xc7z020clg4
ZYNQ-7 ZC702 Evaluation Board Add Companion Card <a href="#">Connections</a>		xilinx.com	1.4	xc7z020clg4
ZYNQ-7 ZC706 Evaluation Board Add Companion Card <a href="#">Connections</a>		xilinx.com	1.4	xc7z045ffg9

? < Back Next > Finish Cancel

Click “Finish”. Vivado will create an empty project.

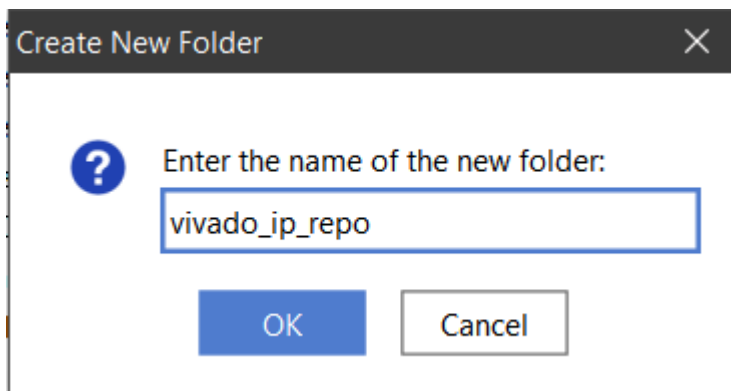
In the project Manager area of the Flow Navigator click “IP Catalog”:



In the IP Catalog tab, right-click and select “IP Settings...”.

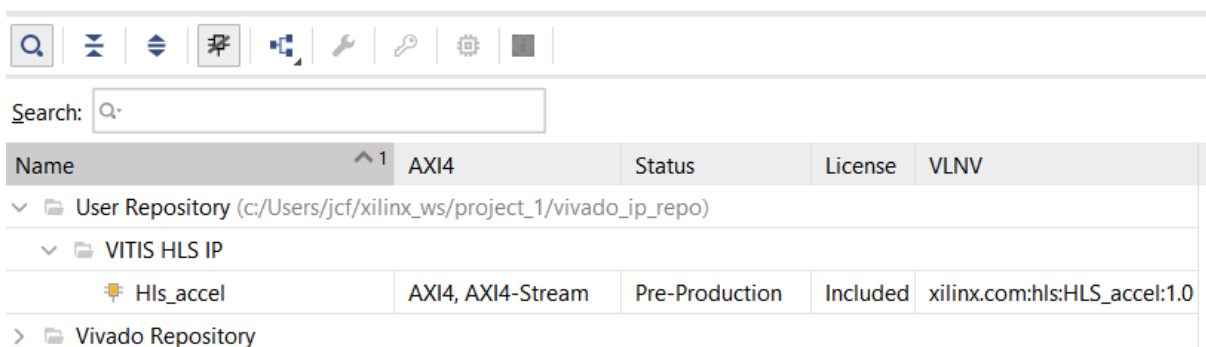
Add a new repository in a new folder. Select IP→ Repository on the left bar (project settings) and use the + button to add a repository.

For that, go to the project directory, click the “Create new folder” icon and enter the folder name

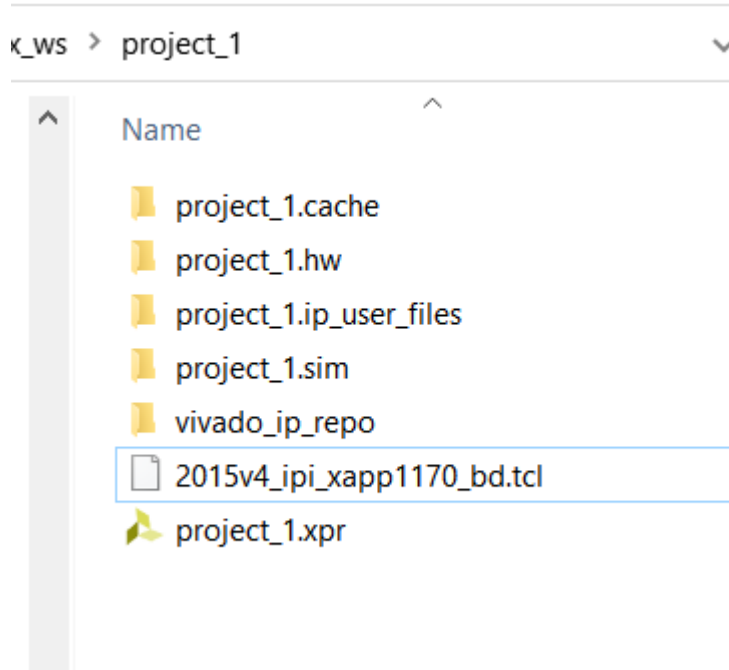


Close the window.

In the IP Catalog window, right-click on “User Repository” and select “Add IP to repository...”, browse to where you store the IP and select it. You should now have an IP in your catalog.

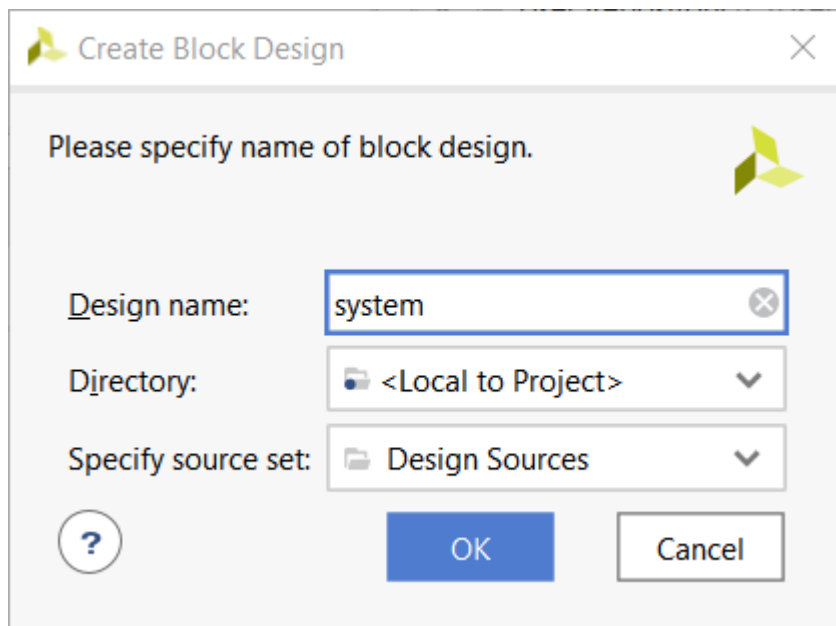


Using the Windows file manager, copy file “2015v4\_ipi\_xapp1170\_bd.tcl” to the project directory.

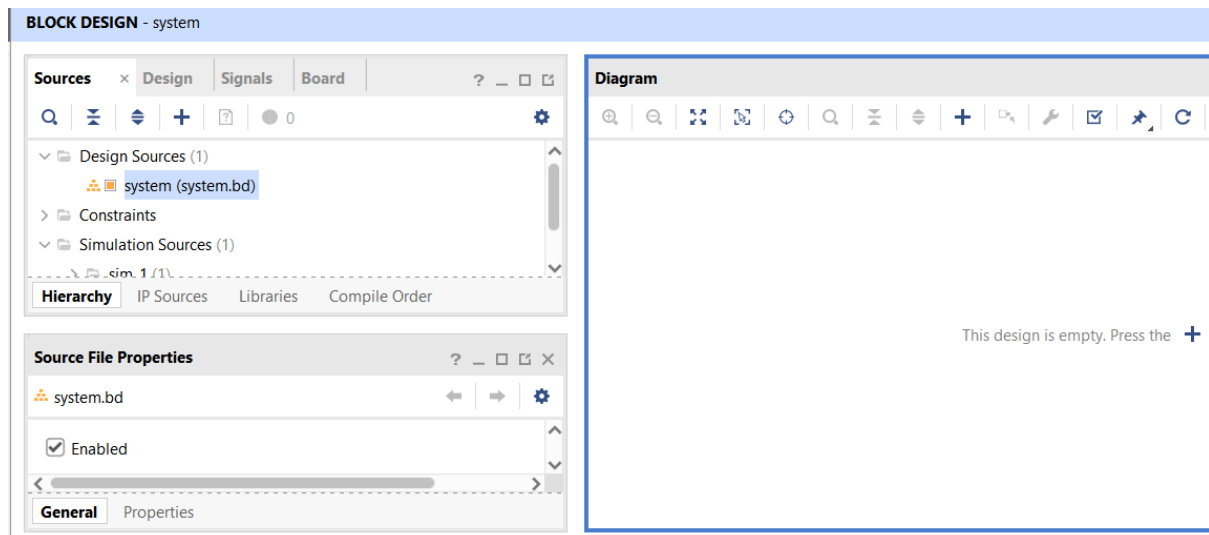


Go to the TCL console (a tab in the bottom of the Vivado window) and enter the command **source 2015v4\_ipi\_xapp1170\_bd.tcl** and press Enter.

Click Create Block Design in the Flow Navigator. Create a design called “system”.

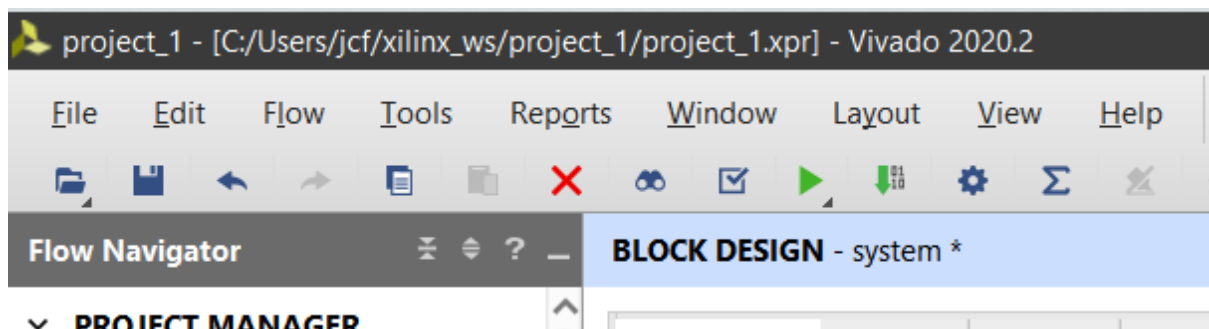


You should have and empty Diagram tab.



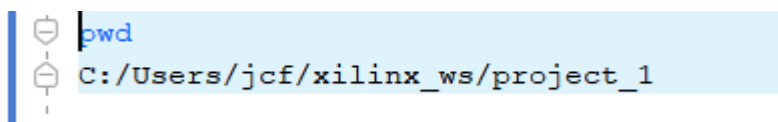
Go to the **Tcl Console** tab.

Use the command “cd” to change the current directory to the project directory (which is shown in the window bar)



`cd directory_name`

You can use the “pwd” command to determine the current directory (example below).



In the project directory, execute the Tcl command:

`source create_design.tcl`

Right-click in the Diagram tab, and select “Regenerate Layout”

Validate the design using the validation icon  
Everything should be OK.



