# Image Dehazing On ZYNQ

**What Is an Image?**

Before beginning image processing, it is essential to comprehend what an image is. Based on the number of pixels, an image's dimensions (height and breadth) serve as a representation. For instance, if an image is 500 × 400 (width x height), then 200000 pixels make up the entire image.

This pixel is a point on the image that takes on a specific shade, opacity, or color. It is usually represented in one of the following:

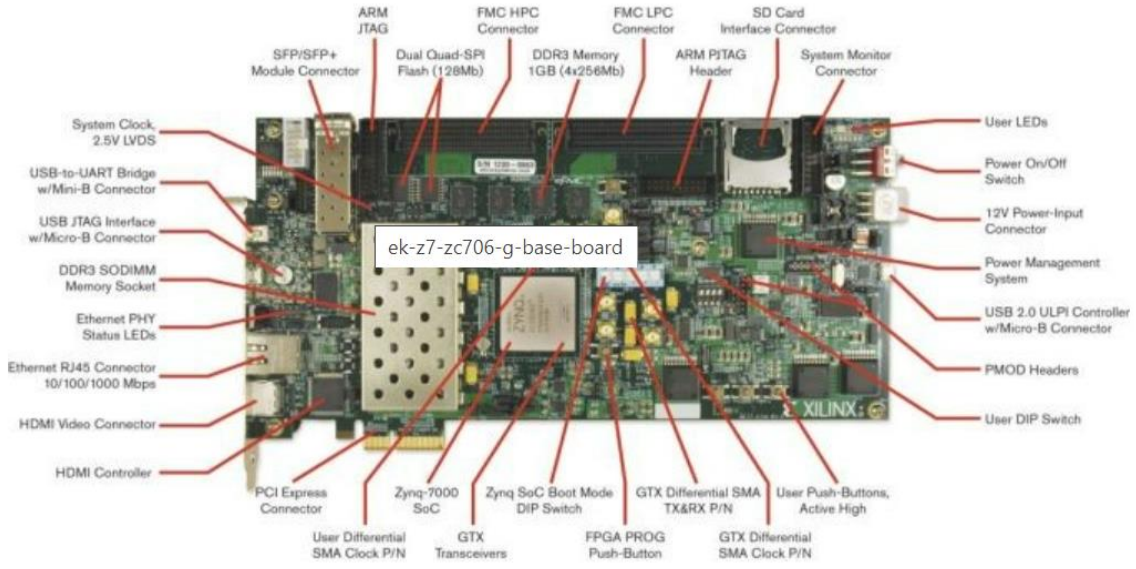
- Grayscale - A pixel is an integer with a value between 0 to 255 (0 is completely black, and 255 is entirely white).
- RGB - A pixel comprises three integers between 0 to 255 (the integers represent the intensity of red, green, and blue).
- RGBA is an extension of RGB with an added alpha field, representing the image's opacity.

The process of converting an image into a digital format and carrying out specific procedures to extract some useful information from it is known as image processing. The image processing system typically interprets all images as 2D signals when implementing specific specified signal processing techniques.

Image processing's primary goal is to convert a physical image into a digital format and apply various operations to create specific models or extract information from the image.

## Zynq7000 SoC ZC706 Evaluation Kit:-

The Zynq™ 7000 SoC ZC706 Evaluation Kit includes all the basic components of hardware, design tools, IP, and pre-verified reference designs including a targeted design, enabling a complete embedded processing platform and transceiver based designs including PCIe. The included pre-verified reference designs and industry-standard FPGA Mezzanine Connectors (FMC) allow scaling and customization with daughter cards. Please check the PetaLinux Software Development Kit page for information on the AMD processing systems.




## FEATURES

- Optimized for quickly prototyping embedded applications using Zynq 7000 SoCs
- Hardware, design tools, IP, and pre-verified reference designs
- Demonstrates a embedded design, targeting video pipeline

- Advanced memory interface with
- 1GB DDR3 Component Memory
- 1GB DDR3 SODIM Memory
- Enabling serial connectivity with PCIe Gen2x4, SFP+ and SMA Pairs, USB OTG, UART, IIC
- Supports embedded processing with Dual ARM Cortex-A9 core processors
- Develop networking applications with 10-100-1000 Mbps Ethernet (RGMII)
- Implement Video display applications with HDMI out
- Expand I/O with the FPGA Mezzanine Card (FMC) interface

**General Design Flow in Zynq:**

**Xilinx Folder:** D:\Xilinx

**Image for dehazing: ..\..\Downloads\haze.png**

I. Vivado

- Open Vivado and select Zedboard
- Create a new Vivado Project
- Create an empty block design workspace inside the new project
- Add required IP blocks using the IP integrator tool and build Hardware Design
- Validate and save block design
- Create HDL system wrapper
- Run design Synthesis and Implementation
- Generate Bit File
- Export Hardware Design, including the generated bitstream file, to the SDK tool

Launch SDK

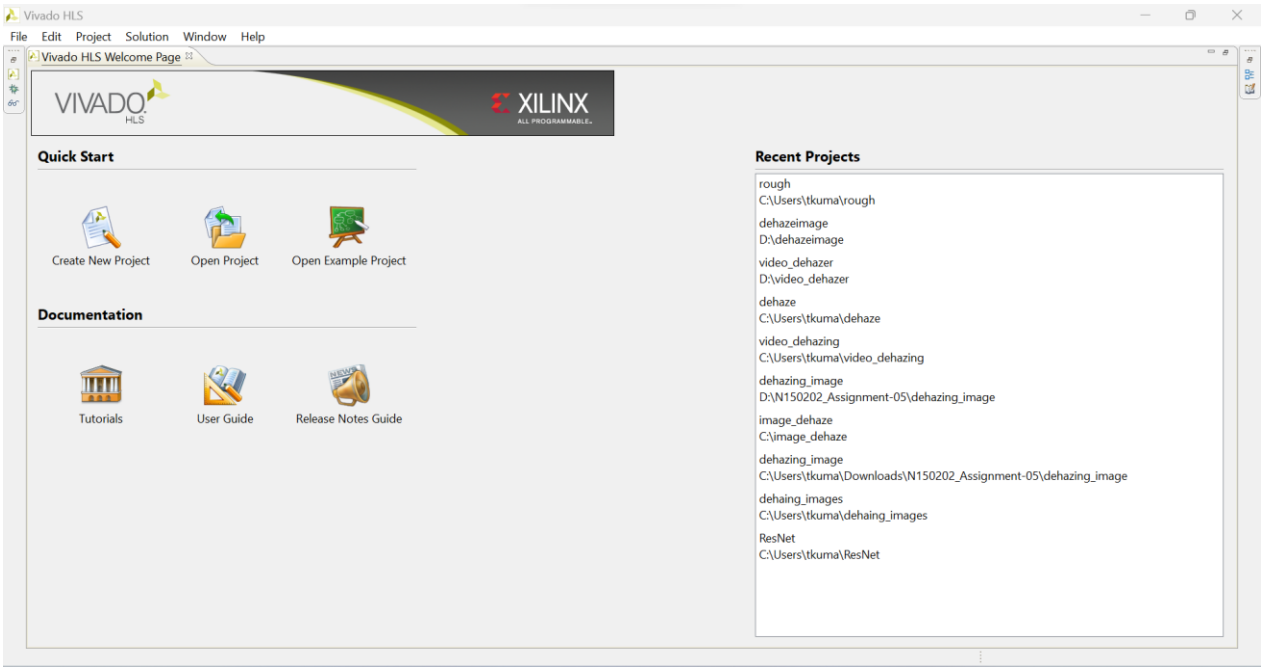
Now the Hardware design is exported to the SDK tool. The Vivado to SDK hand-off is done internally through Vivado. We will use SDK to create a Software application that will use the customized board interface data and FPGA hardware configuration by importing the hardware design information from Vivado.

II. SDK

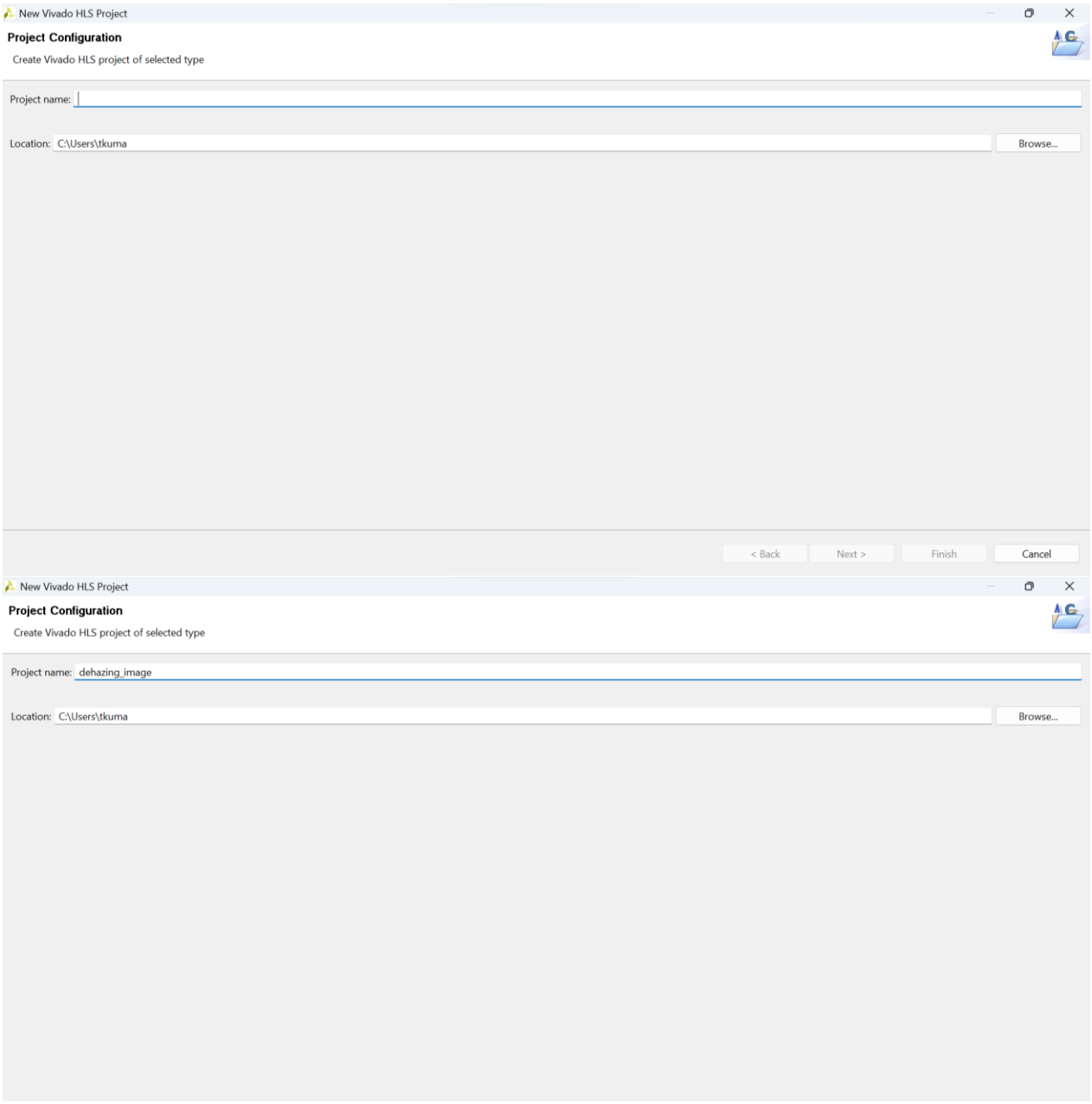
- Create a new application project and select the default Hello World template
- Program FPGA and run the application

▼ **BOARD USED:** ZYNQ 7000 ZC706

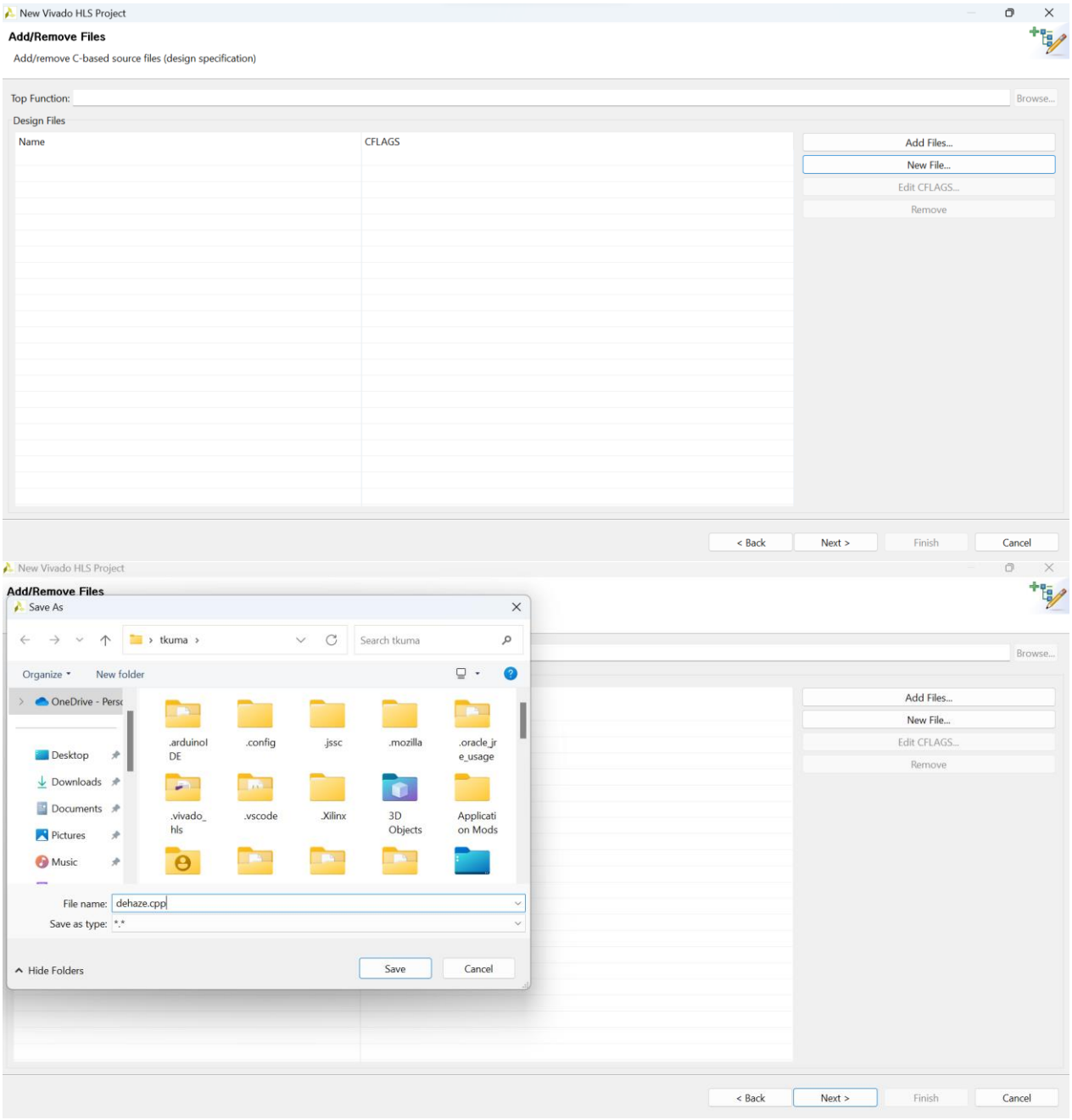
1. Step-1: Open the **Vivado HLS** on your pc and click on create new project.

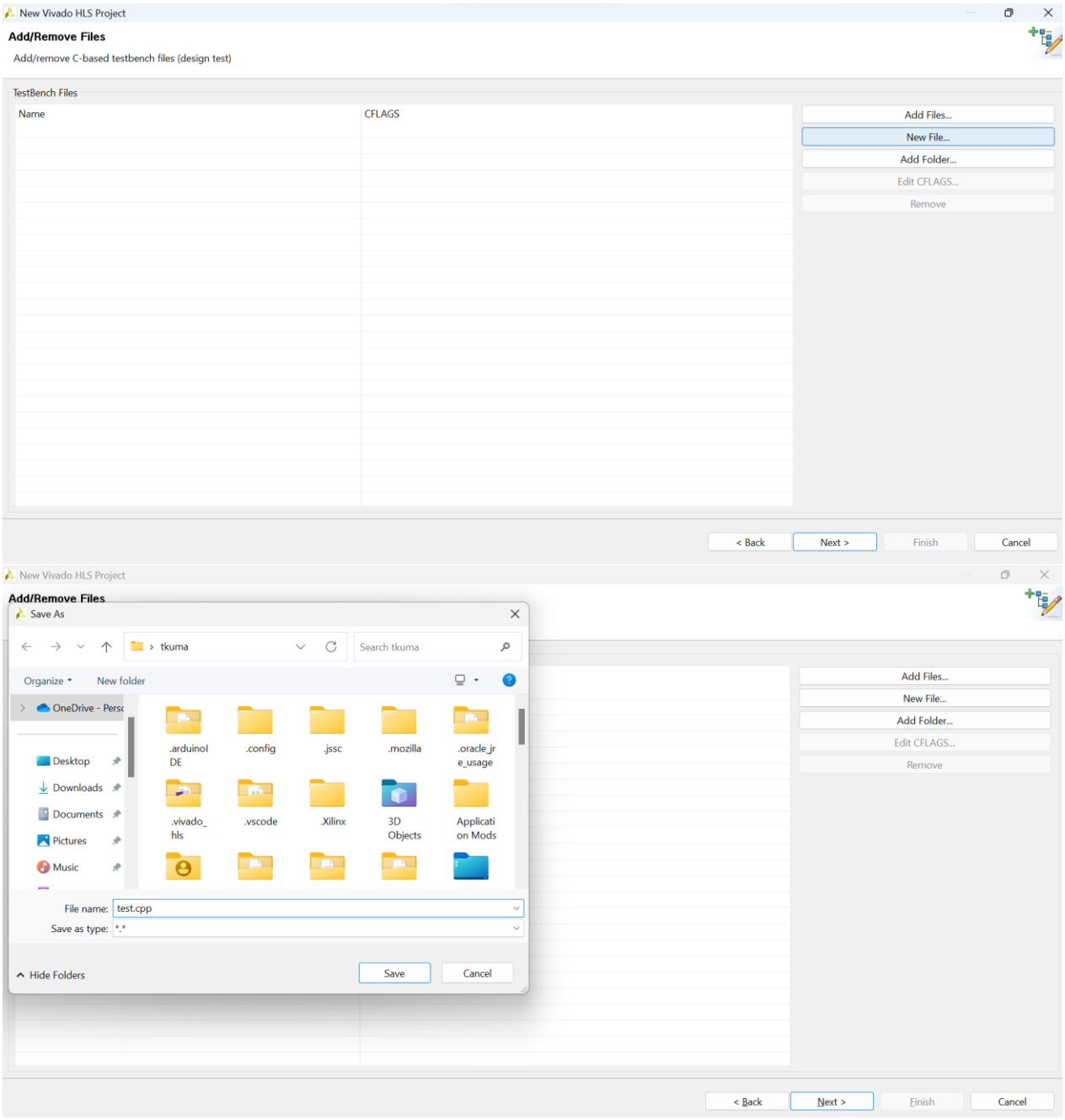



Then give a name to the project

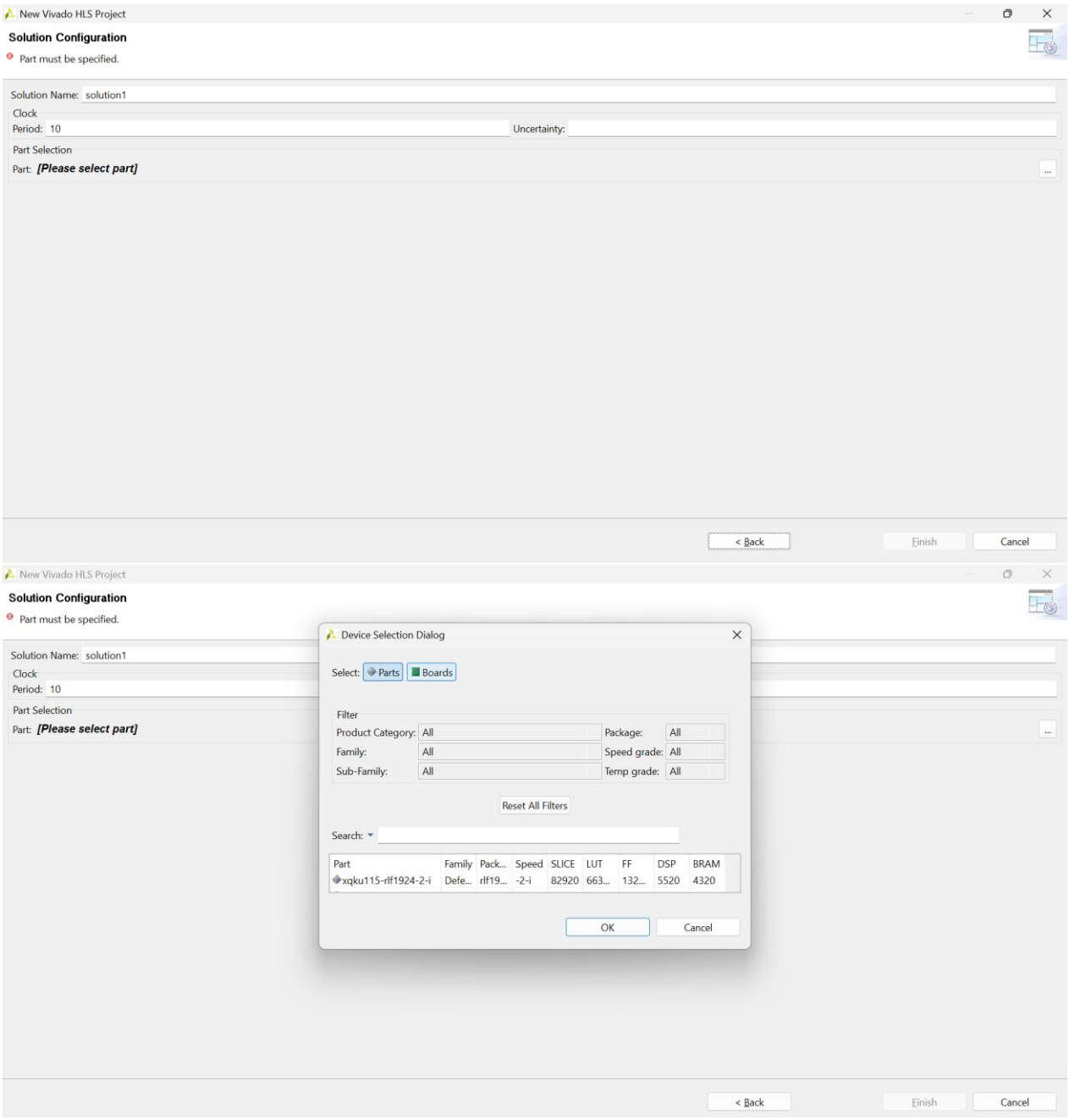


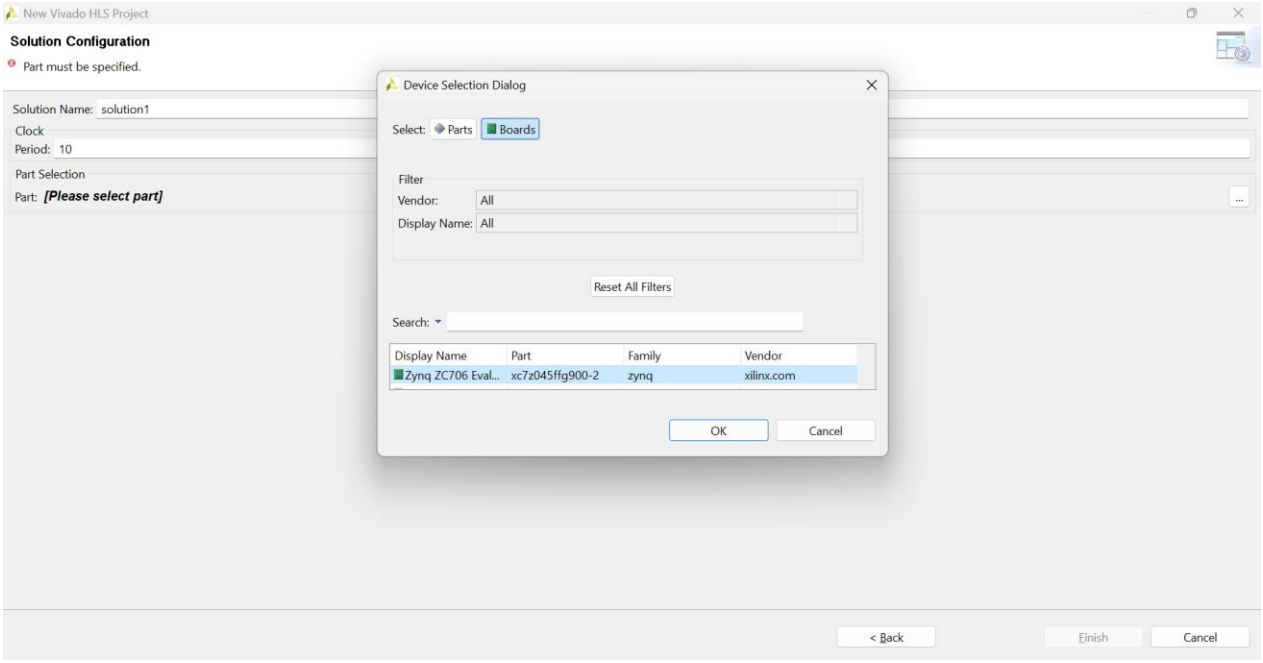

Assign the Top function and Testbench functions.

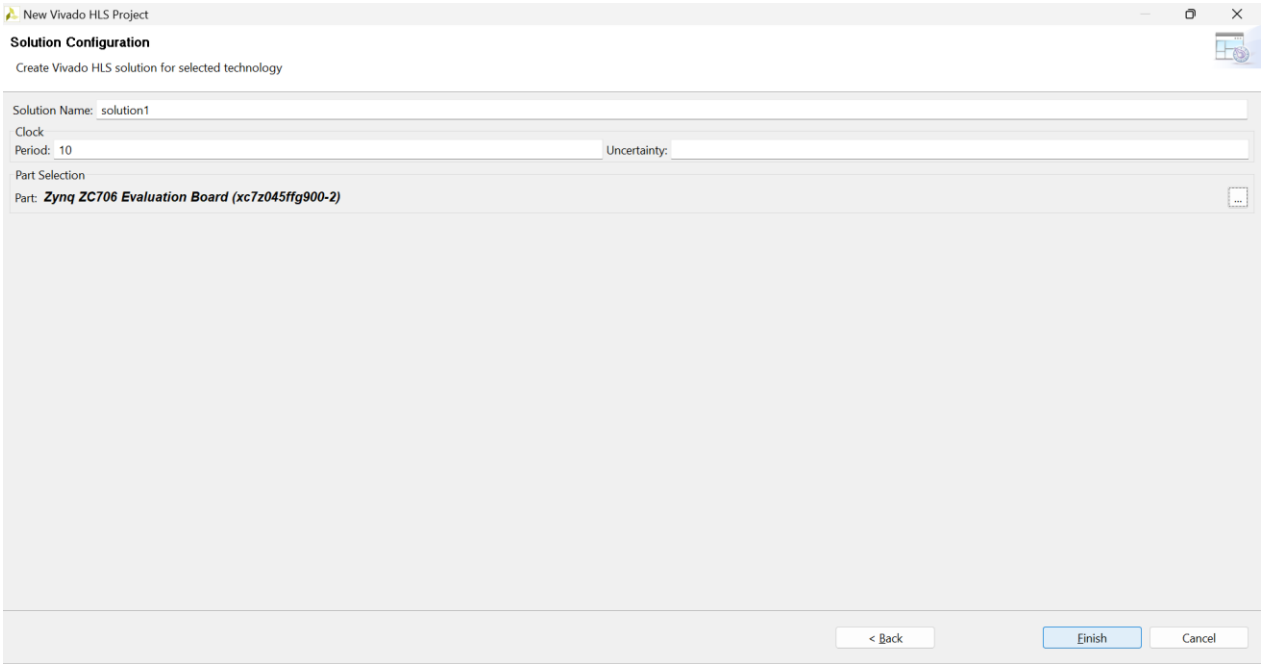

New Vivado HLS Project
Add/Remove Files
Add/remove C-based source files (design specification)
Top Function:
Browse...
Design Files
Name
CFLAGS
Add Files...
New File...
Edit CFLAGS...
Remove
< Back
Next >
Finish
Cancel
New Vivado HLS Project
Add/Remove Files
Save As
tkuma
Search tkuma
Organize
New folder
OneDrive - Perso
Desktop
Downloads
Documents
Pictures
Music
.arduinoIDE
.config
.jssc
.mozilla
.oracle_jre_usage
.vivado_hls
.vscode
.Xilinx
3D Objects
Application Mods
File name: dehaze.cpp
Save as type: *.*
Hide Folders
Save
Cancel
Browse...
Add Files...
New File...
Edit CFLAGS...
Remove
< Back
Next >
Finish
Cancel




Select required board for implamentation


New Vivado HLS Project
Solution Configuration
Part must be specified.
Solution Name: solution1
Clock
Period: 10
Uncertainty:
Part Selection
Part: [Please select part]
< Back
Finish
Cancel
New Vivado HLS Project
Solution Configuration
Part must be specified.
Device Selection Dialog
Select: Parts Boards
Filter
Product Category: All
Package: All
Family: All
Speed grade: All
Sub-Family: All
Temp grade: All
Reset All Filters
Search:
Part Family Pack... Speed SLICE LUT FF DSP BRAM
xqku115-rlf1924-2-i Defe... rlf19... -2-i 82920 663... 132... 5520 4320
OK
Cancel
Solution Name: solution1
Clock
Period: 10
Part Selection
Part: [Please select part]
< Back
Finish
Cancel




Then write the code for the project
Here for image dehazing, we have used Dark Channel Prior Method
Here is the step wise explaination
1. Dark Channel Computation:
 - Compute the dark channel of a hazy image. The dark channel is obtained by finding the minimum pixel value in a local neighborhood across each channel of the image. It represents areas with low intensity or minimal haze.

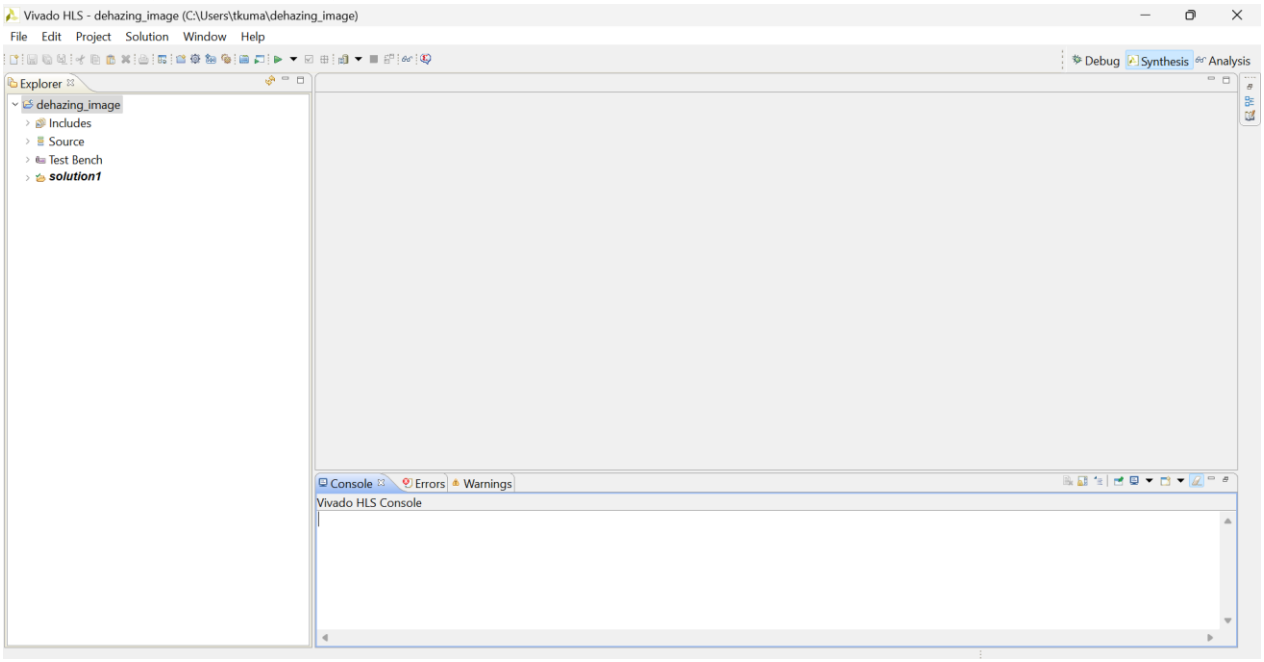
2. Atmospheric Light Estimation:
 - Estimate the atmospheric light, which represents the maximum intensity of light in the hazy scene. This can be done by finding the maximum pixel value across all channels of the hazy image.

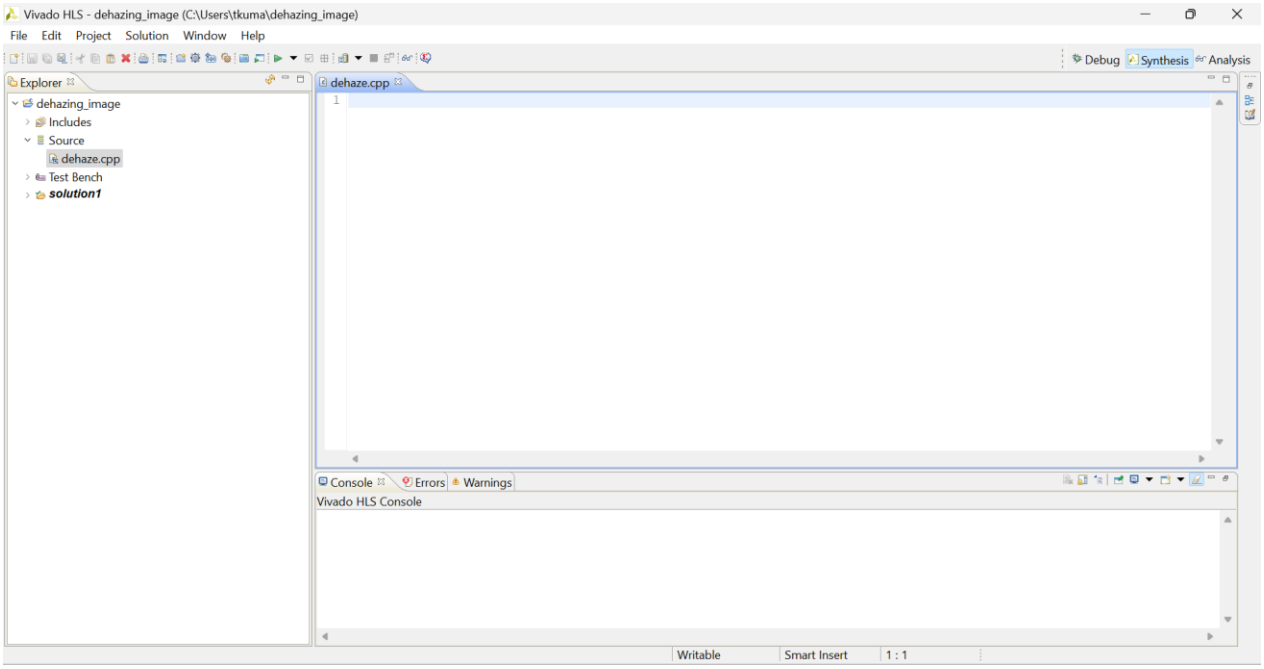
3. Transmission Map Estimation:
 - Calculate the transmission map, which describes the relative haze thickness at each pixel. The transmission map is estimated based on the dark channel and atmospheric light using a dehazing model.

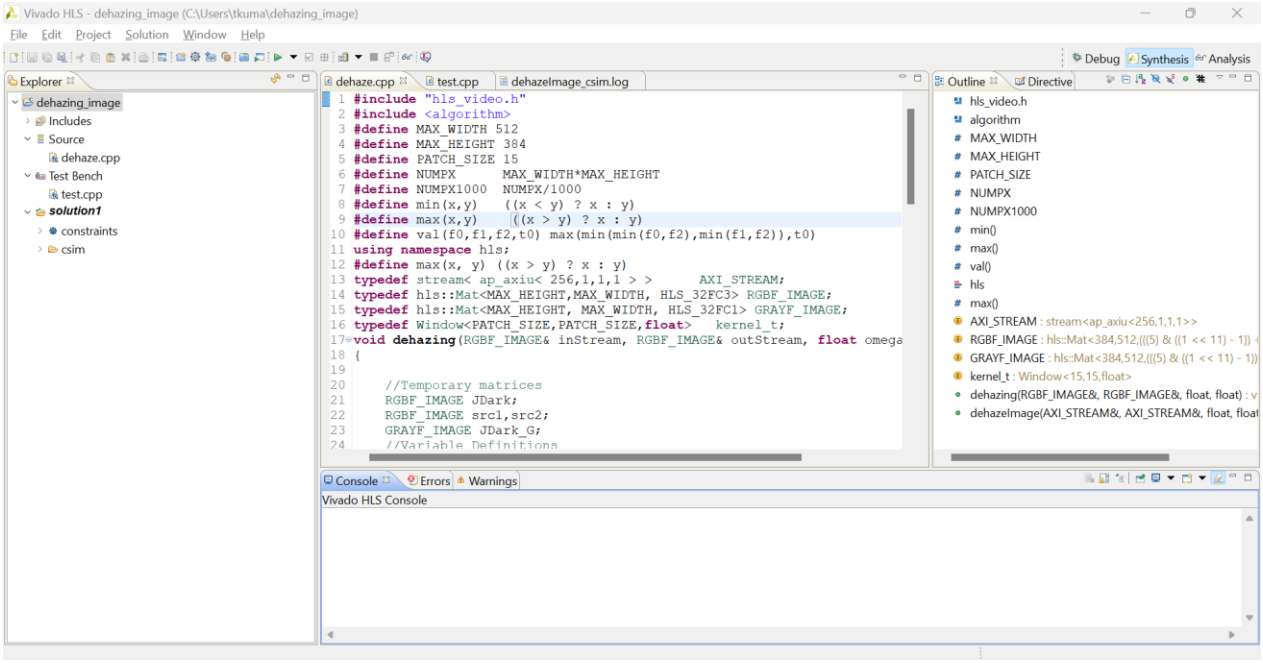
4. Haze Removal:
 - Remove the haze from the hazy image using the estimated transmission map. This step involves applying a dehazing algorithm to recover the haze-free scene.

5. Output:
 - The result is a dehazed image, which should have reduced or eliminated the atmospheric haze, revealing clearer and more detailed information in the scene.

It's important to note that the Dark Channel Prior is one of the popular techniques for image dehazing, but there are other methods as well, each with its strengths and weaknesses.
Here is the coding part of the dehazing



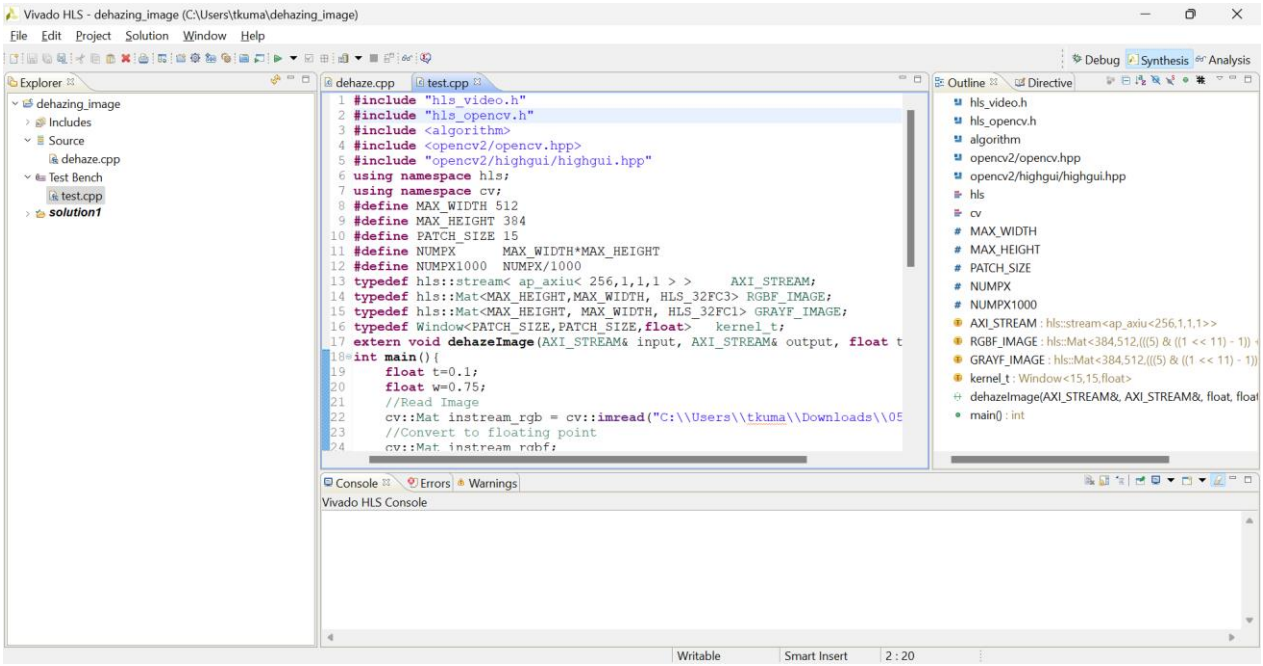






Code for dehaze.cpp:

```
#include "hls_video.h"
#include <algorithm>
#define MAX_WIDTH 512
#define MAX_HEIGHT 384
#define PATCH_SIZE 15
#define NUMPX            MAX_WIDTH*MAX_HEIGHT
#define NUMPX1000  NUMPX/1000
#define min(x,y)   ((x < y) ? x : y)
#define max(x,y)         ((x > y) ? x : y)
#define val(f0,f1,f2,t0) max(min(min(f0,f2),min(f1,f2)),t0)
using namespace hls;
#define max(x, y) ((x > y) ? x : y)
typedef stream< ap_axiu< 256,1,1,1 > >                    AXI_STREAM;
typedef hls::Mat<MAX_HEIGHT,MAX_WIDTH, HLS_32FC3> RGBF_IMAGE;
typedef hls::Mat<MAX_HEIGHT, MAX_WIDTH, HLS_32FC1> GRAYF_IMAGE;
typedef Window<PATCH_SIZE,PATCH_SIZE,float>   kernel_t;
void dehazing(RGBF_IMAGE& inStream, RGBF_IMAGE& outStream, float omega,float t0)
{

                //Temporary matrices
                RGBF_IMAGE JDark;
                RGBF_IMAGE src1,src2;
                GRAYF_IMAGE JDark_G;
                //Variable Definitions
                int i;

                Duplicate (inStream,src1,src2);
                //Errode Image
                kernel_t kernel;
                Point_<int> anchor((PATCH_SIZE+1)/2,(PATCH_SIZE+1)/2);
                morp_opr<erode_kernel,BORDER_REPLICATE,MORPH_RECT,1>(src1,JDark,kernel,anchor);

                GRAYF_IMAGE JDarkR,JDarkG,JDarkB,tmp;

                Split(JDark,JDarkR,JDarkG,JDarkB);

                Max(JDarkR,JDarkG,tmp);
```

```
		Max(JDarkB,tmp,JDark_G);
		//Taking atmosphere as 1
		//For now JDark1 is JDark since Atmosphere is 1
		GRAYF_IMAGE Omega,OmegaJDark,transmission;

		Scalar<1,float> one(1);

		Scalar<3,float> atmos(1.0,1.0,1.0),im; //Atmosphere is 1

		Scalar<1,float> OmegaS(omega);

		Set(OmegaS,Omega);
		Mul(Omega,JDark_G,OmegaJDark); //Omega*JDark
		SubRS(OmegaJDark,one,transmission); //Transmission= 1- Omega*JDark

		Scalar<1,float> temp; //Scalar to hold the pixels
		float value; //To store transmission values

		for (i=0;i < NUMPX; ++i){
			transmission >> temp;
			src2 >> im;
			value = max(temp.val[0],t0);
			outStream << (atmos+ ((im-atmos)/value));
		}
}
void dehazeImage(AXI_STREAM& input, AXI_STREAM& output, float t,float w) {
#pragma HLS INTERFACE axis port=input
#pragma HLS INTERFACE axis port=output
#pragma HLS INTERFACE s_axilite port=t bundle=control
#pragma HLS INTERFACE s_axilite port=w bundle=control
#pragma HLS INTERFACE s_axilite port=return bundle=control
  RGBF_IMAGE input_img;
  RGBF_IMAGE output_img;
#pragma HLS dataflow
  AXIvideo2Mat(input, input_img);
  dehazing(input_img,output_img,w,t);
  Mat2AXIvideo(output_img, output);
}
```

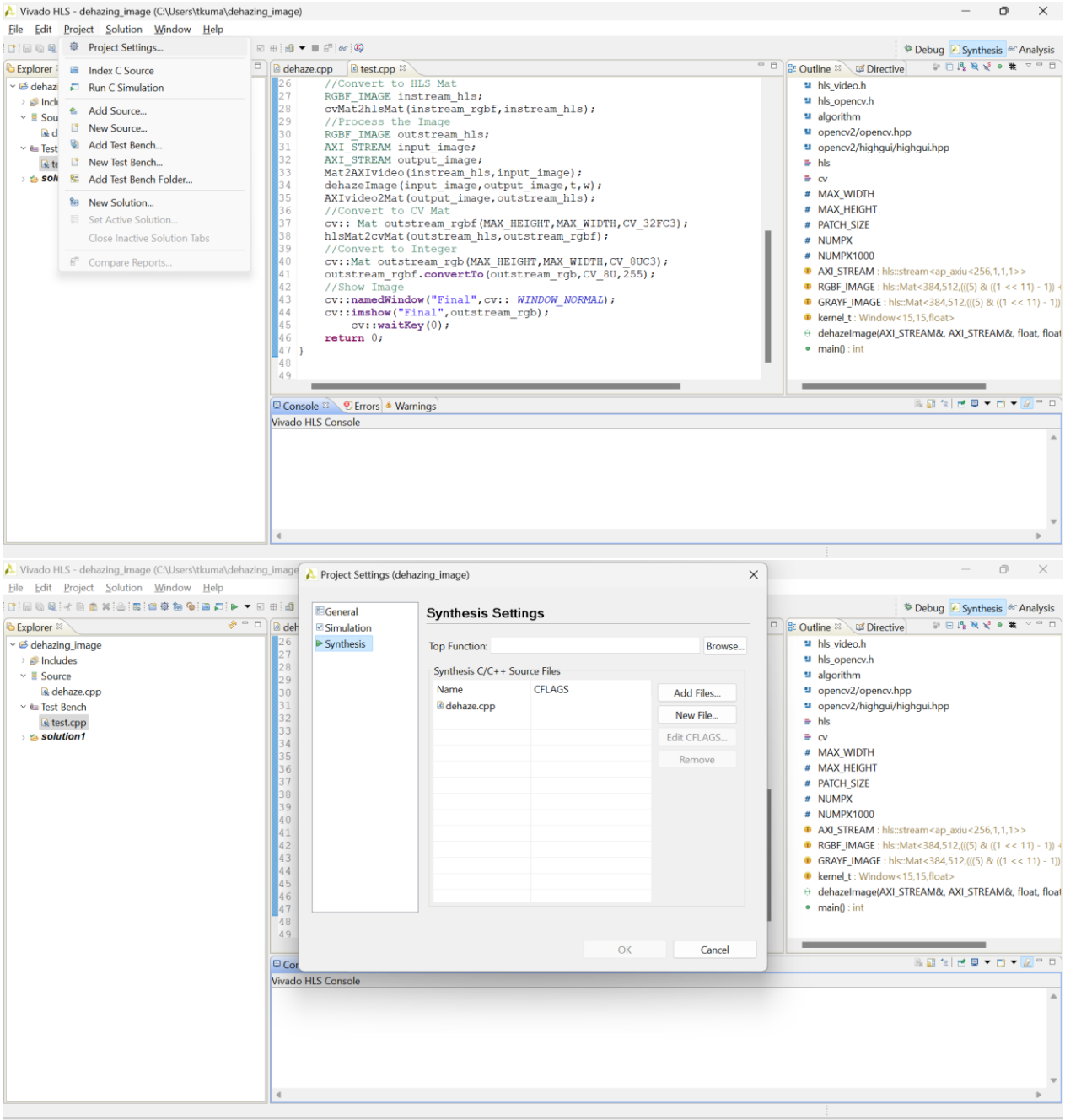


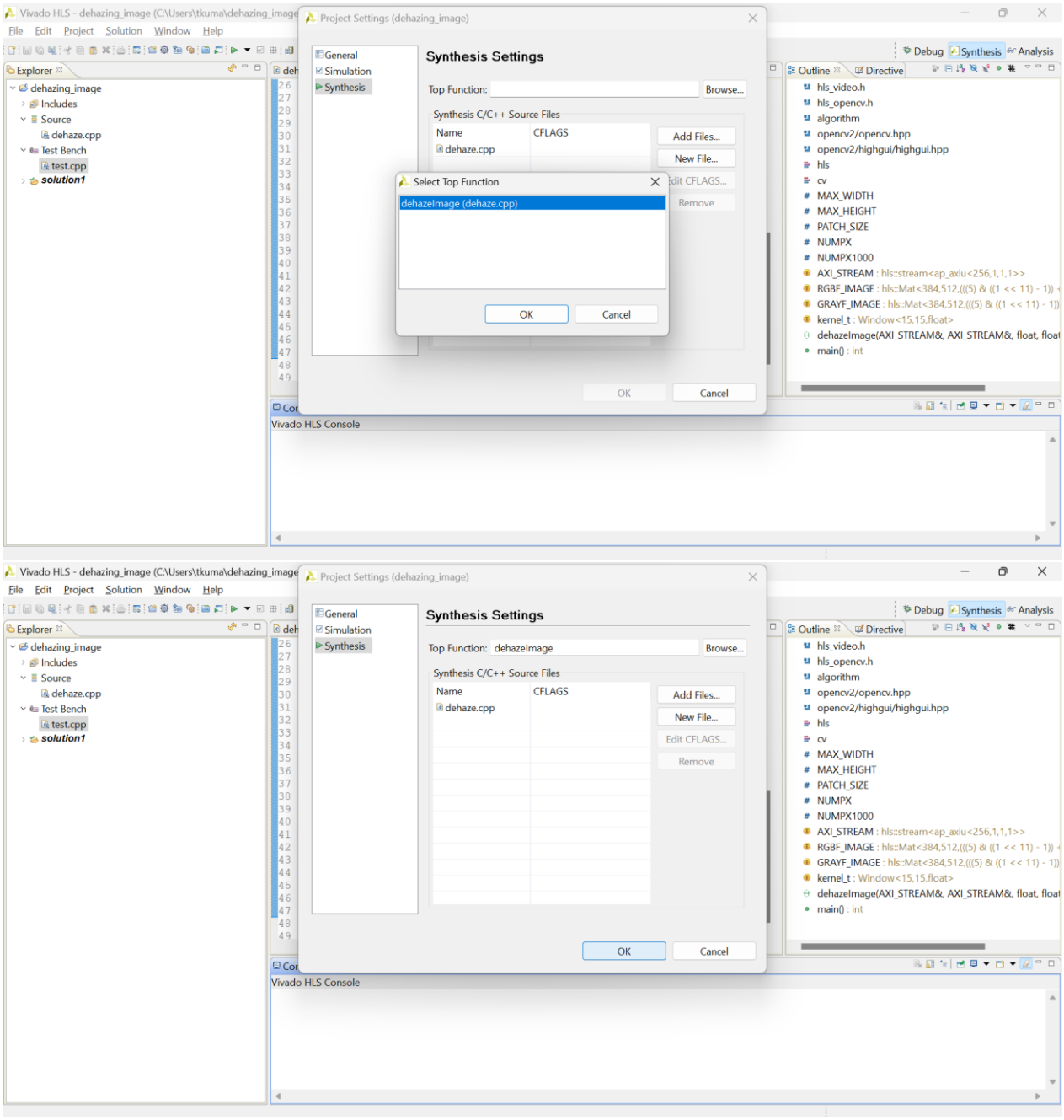
Code for testbench:

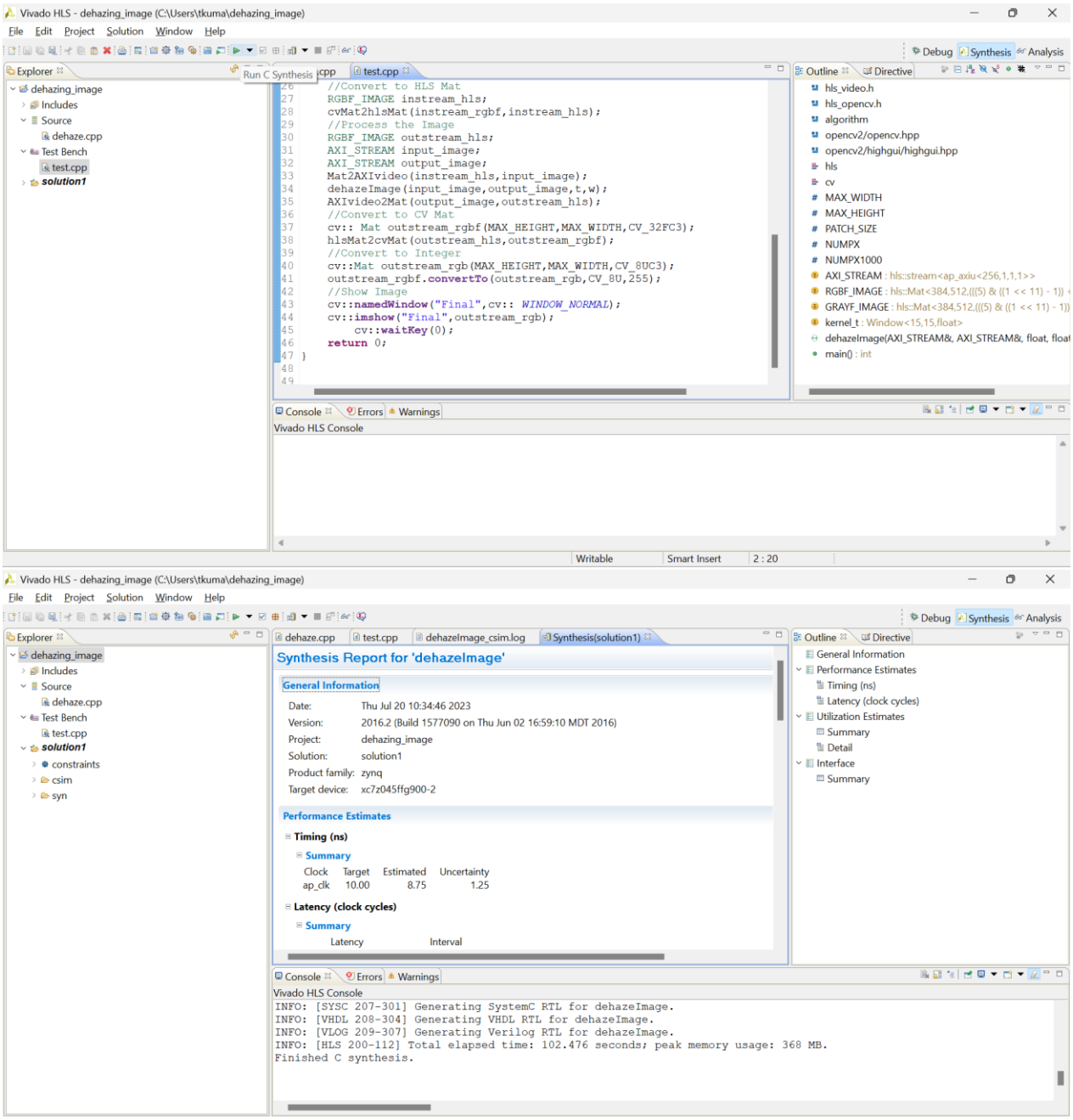
```
#include "hls_video.h"
#include "hls_opencv.h"
#include <algorithm>
#include <opencv2/opencv.hpp>
#include "opencv2/highgui/highgui.hpp"
using namespace hls;
using namespace cv;
#define MAX_WIDTH 512
#define MAX_HEIGHT 384
#define PATCH_SIZE 15
#define NUMPX            MAX_WIDTH*MAX_HEIGHT
#define NUMPX1000  NUMPX/1000
typedef hls::stream< ap_axiu< 256,1,1,1 > >                AXI_STREAM;
typedef hls::Mat<MAX_HEIGHT,MAX_WIDTH, HLS_32FC3> RGBF_IMAGE;
typedef hls::Mat<MAX_HEIGHT, MAX_WIDTH, HLS_32FC1> GRAYF_IMAGE;
typedef Window<PATCH_SIZE,PATCH_SIZE,float>   kernel_t;
extern void dehazeImage(AXI_STREAM& input, AXI_STREAM& output, float t,float w);
int main(){
        float t=0.1;
        float w=0.75;
        printf("t,w\n");
        //Read Image
        cv::Mat instream_rgb = cv::imread("C:\\Users\\tkuma\\Downloads\\haze.png");
        printf("1,2,3..\n");
        //Convert to floating point
        cv::Mat instream_rgbf;
        printf("4\n");
        instream_rgb.convertTo(instream_rgbf,CV_32F, 1.0/255);
        printf("5\n");
        //Convert to HLS Mat
        RGBF_IMAGE instream_hls;
        printf("6\n");
        cvMat2hlsMat(instream_rgbf,instream_hls);
        printf("7\n");
        //Process the Image
        RGBF_IMAGE outstream_hls;
        AXI_STREAM input_image;
        AXI_STREAM output_image;
        Mat2AXIvideo(instream_hls,input_image);
        printf("8\n");
        dehazeImage(input_image,output_image,t,w);
        printf("9\n");
        AXIvideo2Mat(output_image,outstream_hls);
        //Convert to CV Mat
        printf("10\n");
        cv:: Mat outstream_rgbf(MAX_HEIGHT,MAX_WIDTH,CV_32FC3);
        printf("11\n");
        hlsMat2cvMat(outstream_hls,outstream_rgbf);
        //Convert to Integer
        printf("12\n");
        cv::Mat outstream_rgb(MAX_HEIGHT,MAX_WIDTH,CV_8UC3);
        printf("13\n");
        outstream_rgbf.convertTo(outstream_rgb,CV_8U,255);
        printf("14\n");
        //Show Image
        cv::namedWindow("Final",cv:: WINDOW_NORMAL);
        printf("15\n");
        cv::imshow("Final",outstream_rgb);
        printf("16\n");
                cv::waitKey(0);
```

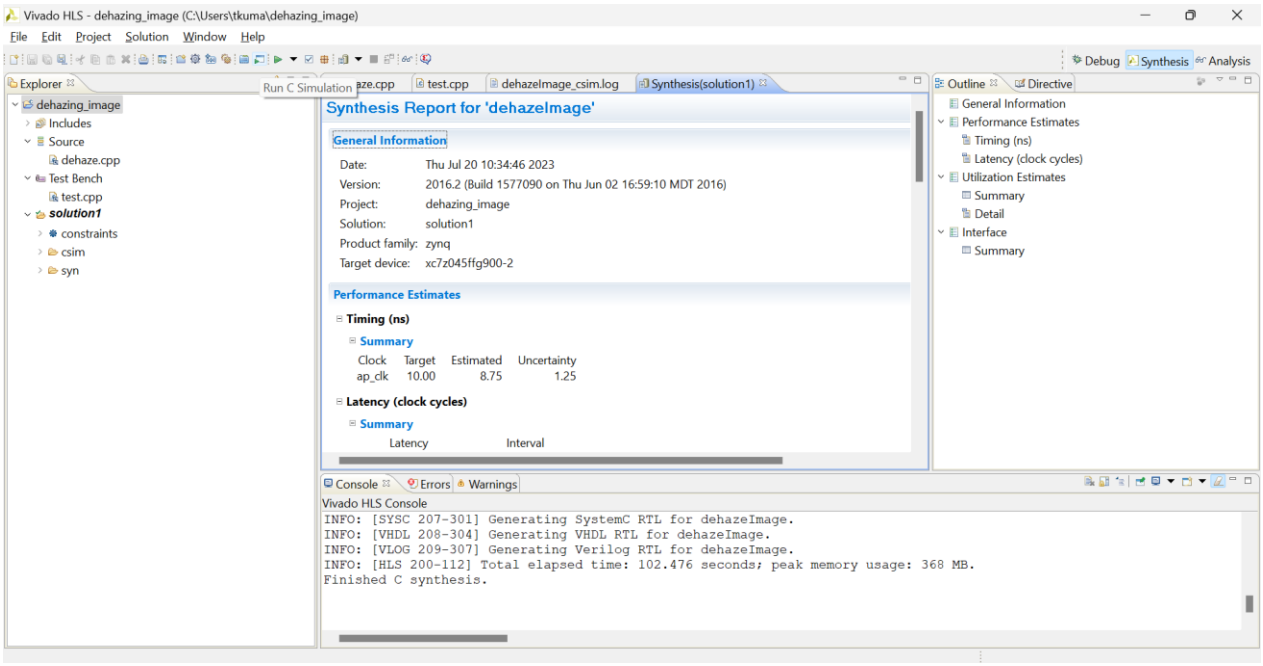
```
    return 0;
}
```

After coding the required logic, Set the dehazeImage as top module for synthesis.

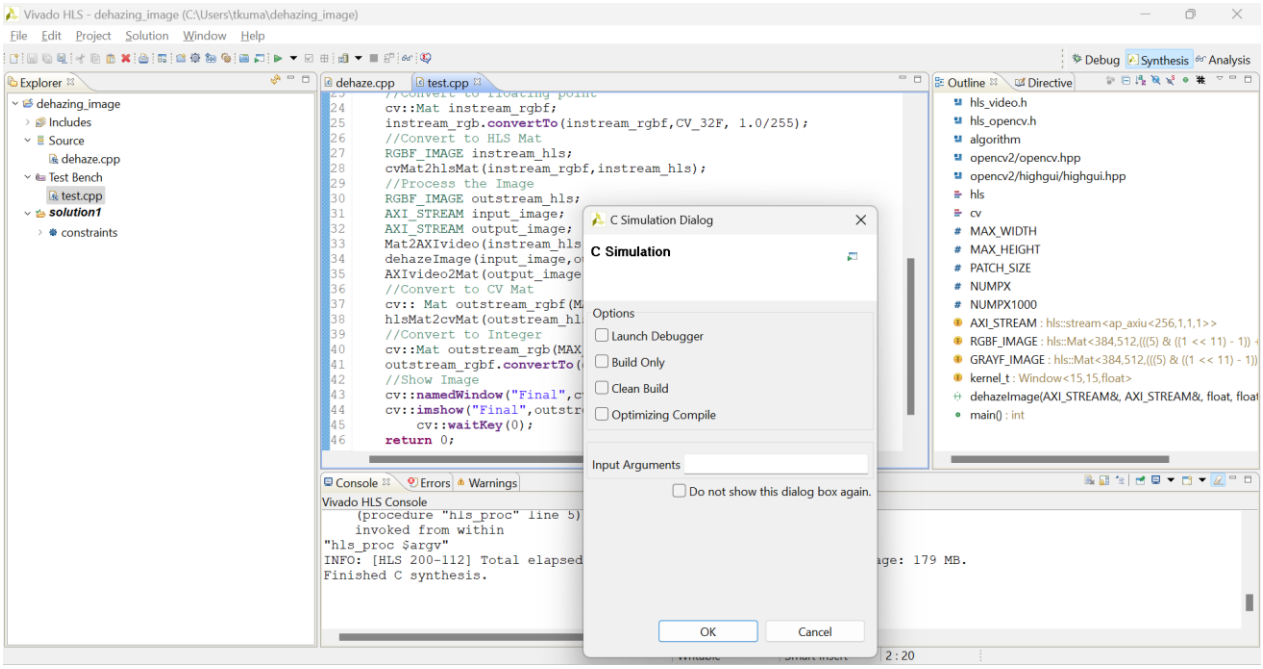


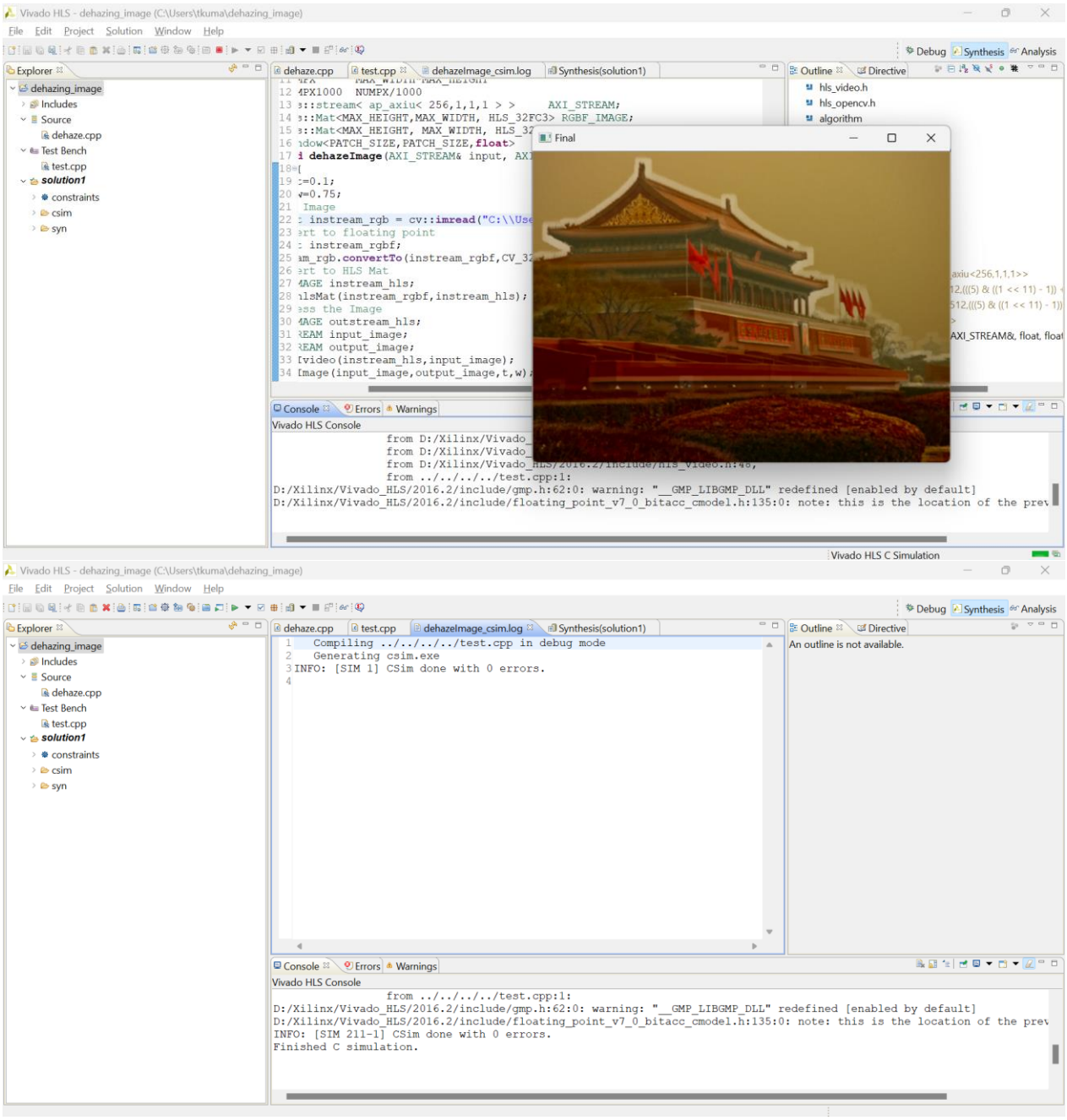

Vivado HLS - dehazing_image (C:\Users\tkuma\dehazing_image
File Edit Project Solution Window Help
Explorer
dehazing_image
Includes
Source
dehaze.cpp
Test Bench
test.cpp
solution1
Project Settings (dehazing_image)
General
Simulation
Synthesis
Synthesis Settings
Top Function:
Browse...
Synthesis C/C++ Source Files
Name
CFLAGS
dehaze.cpp
Add Files...
New File...
Edit CFLAGS...
Remove
Select Top Function
dehazeImage (dehaze.cpp)
OK
Cancel
Debug
Synthesis
Analysis
Outline
Directive
hls_video.h
hls_opencv.h
algorithm
opencv2/opencv.hpp
opencv2/highgui/highgui.hpp
hls
cv
MAX_WIDTH
MAX_HEIGHT
PATCH_SIZE
NUMPX
NUMPX1000
AXI_STREAM : hls::stream<ap_axiu<256,1,1,1>>
RGBF_IMAGE : hls::Mat<384,512,(((5) & ((1 << 11) - 1))
GRAYF_IMAGE : hls::Mat<384,512,(((5) & ((1 << 11) - 1))
kernel_t : Window<15,15,float>
dehazeImage(AXI_STREAM&, AXI_STREAM&, float, float
main() : int
Vivado HLS Console


Vivado HLS - dehazing_image (C:\Users\tkuma\dehazing_image)
File Edit Project Solution Window Help
Debug Synthesis Analysis
Explorer
Run C Synthesis
dehazing_image
Includes
Source
dehaze.cpp
Test Bench
test.cpp
solution1
test.cpp
//Convert to HLS Mat
RGBF_IMAGE instream_hls;
cvMat2hlsMat(instream_rgbf,instream_hls);
//Process the Image
RGBF_IMAGE outstream_hls;
AXI_STREAM input_image;
AXI_STREAM output_image;
Mat2AXIvideo(instream_hls,input_image);
dehazeImage(input_image,output_image,t,w);
AXIvideo2Mat(output_image,outstream_hls);
//Convert to CV Mat
cv:: Mat outstream_rgbf(MAX_HEIGHT,MAX_WIDTH,CV_32FC3);
hlsMat2cvMat(outstream_hls,outstream_rgbf);
//Convert to Integer
cv::Mat outstream_rgb(MAX_HEIGHT,MAX_WIDTH,CV_8UC3);
outstream_rgbf.convertTo(outstream_rgb,CV_8U,255);
//Show Image
cv::namedWindow("Final",cv:: WINDOW_NORMAL);
cv::imshow("Final",outstream_rgb);
cv::waitKey(0);
return 0;
}
Outline Directive
hls_video.h
hls_opencv.h
algorithm
opencv2/opencv.hpp
opencv2/highgui/highgui.hpp
hls
cv
MAX_WIDTH
MAX_HEIGHT
PATCH_SIZE
NUMPX
NUMPX1000
AXI_STREAM : hls::stream<ap_axiu<256,1,1,1>>
RGBF_IMAGE : hls::Mat<384,512,(((5) & ((1 << 11) - 1))
GRAYF_IMAGE : hls::Mat<384,512,(((5) & ((1 << 11) - 1))
kernel_t : Window<15,15,float>
dehazeImage(AXI_STREAM&, AXI_STREAM&, float, float
main() : int
Console Errors Warnings
Vivado HLS Console
Writable Smart Insert 2 : 20
Vivado HLS - dehazing_image (C:\Users\tkuma\dehazing_image)
File Edit Project Solution Window Help
Debug Synthesis Analysis
Explorer
dehazing_image
Includes
Source
dehaze.cpp
Test Bench
test.cpp
solution1
constraints
csim
syn
dehaze.cpp test.cpp dehazeImage_csim.log Synthesis(solution1)
Synthesis Report for 'dehazeImage'
General Information
Date: Thu Jul 20 10:34:46 2023
Version: 2016.2 (Build 1577090 on Thu Jun 02 16:59:10 MDT 2016)
Project: dehazing_image
Solution: solution1
Product family: zynq
Target device: xc7z045ffg900-2
Performance Estimates
Timing (ns)
Summary
Clock Target Estimated Uncertainty
ap_clk 10.00 8.75 1.25
Latency (clock cycles)
Summary
Latency Interval
Outline Directive
General Information
Performance Estimates
Timing (ns)
Latency (clock cycles)
Utilization Estimates
Summary
Detail
Interface
Summary
Console Errors Warnings
Vivado HLS Console
INFO: [SYSC 207-301] Generating SystemC RTL for dehazeImage.
INFO: [VHDL 208-304] Generating VHDL RTL for dehazeImage.
INFO: [VLOG 209-307] Generating Verilog RTL for dehazeImage.
INFO: [HLS 200-112] Total elapsed time: 102.476 seconds; peak memory usage: 368 MB.
Finished C synthesis.



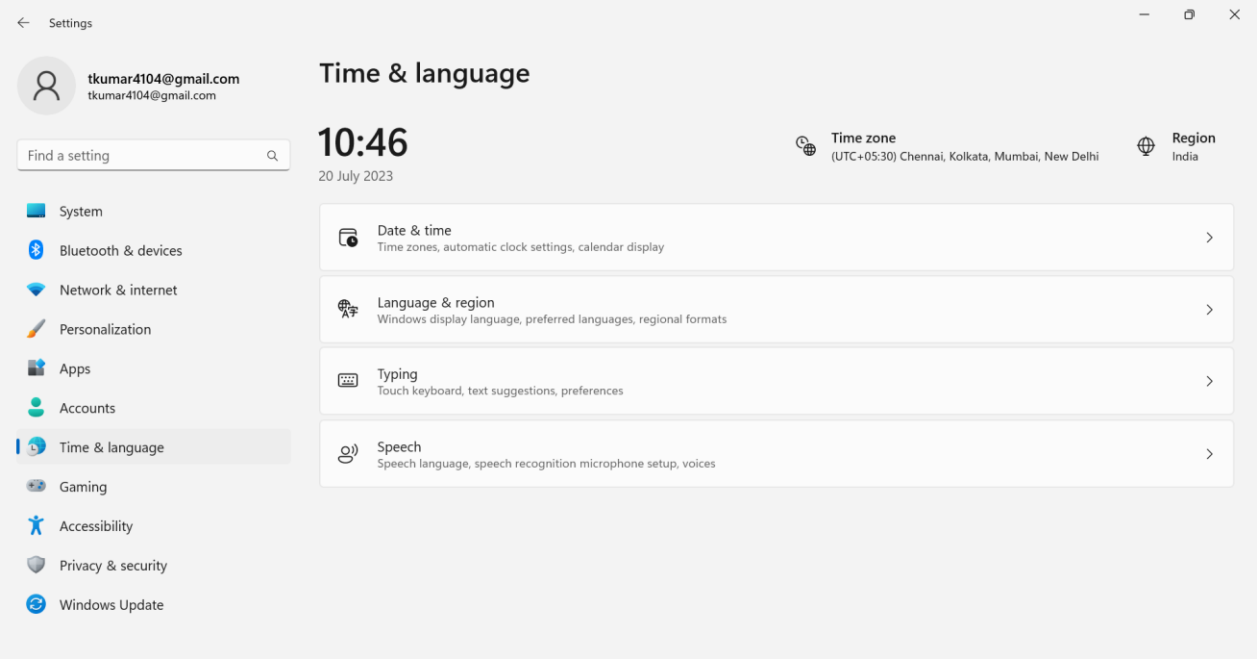

After synthesizing, check for the results to verify our result, for that click on run simulation.

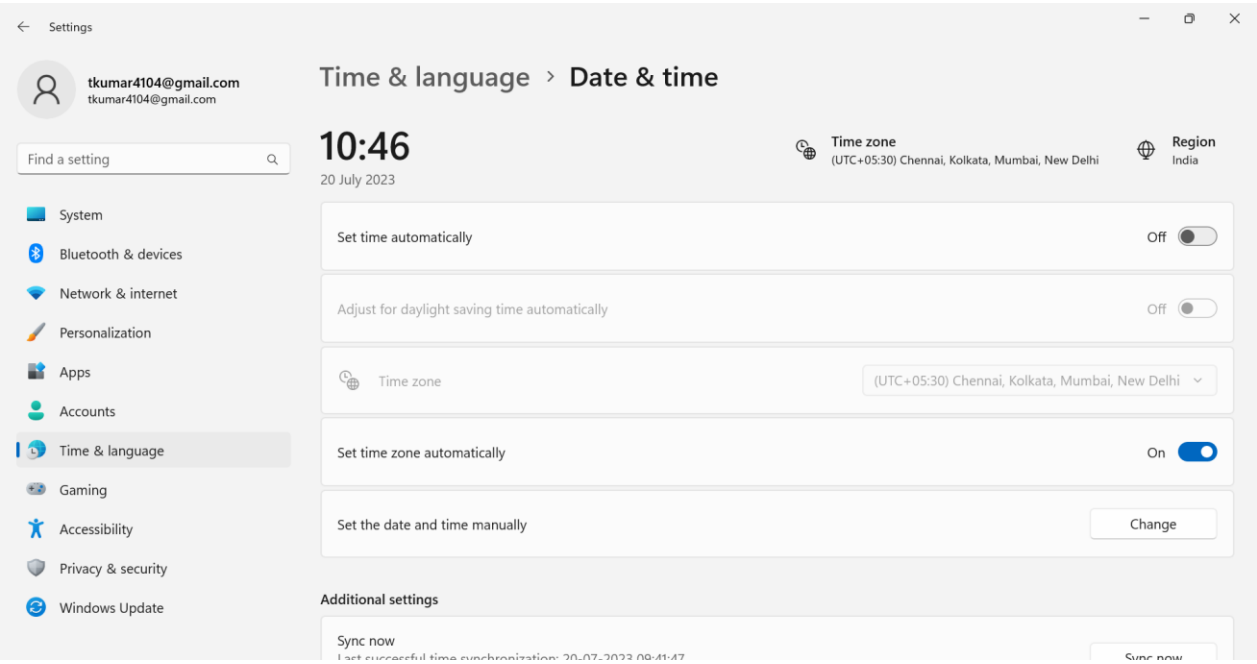


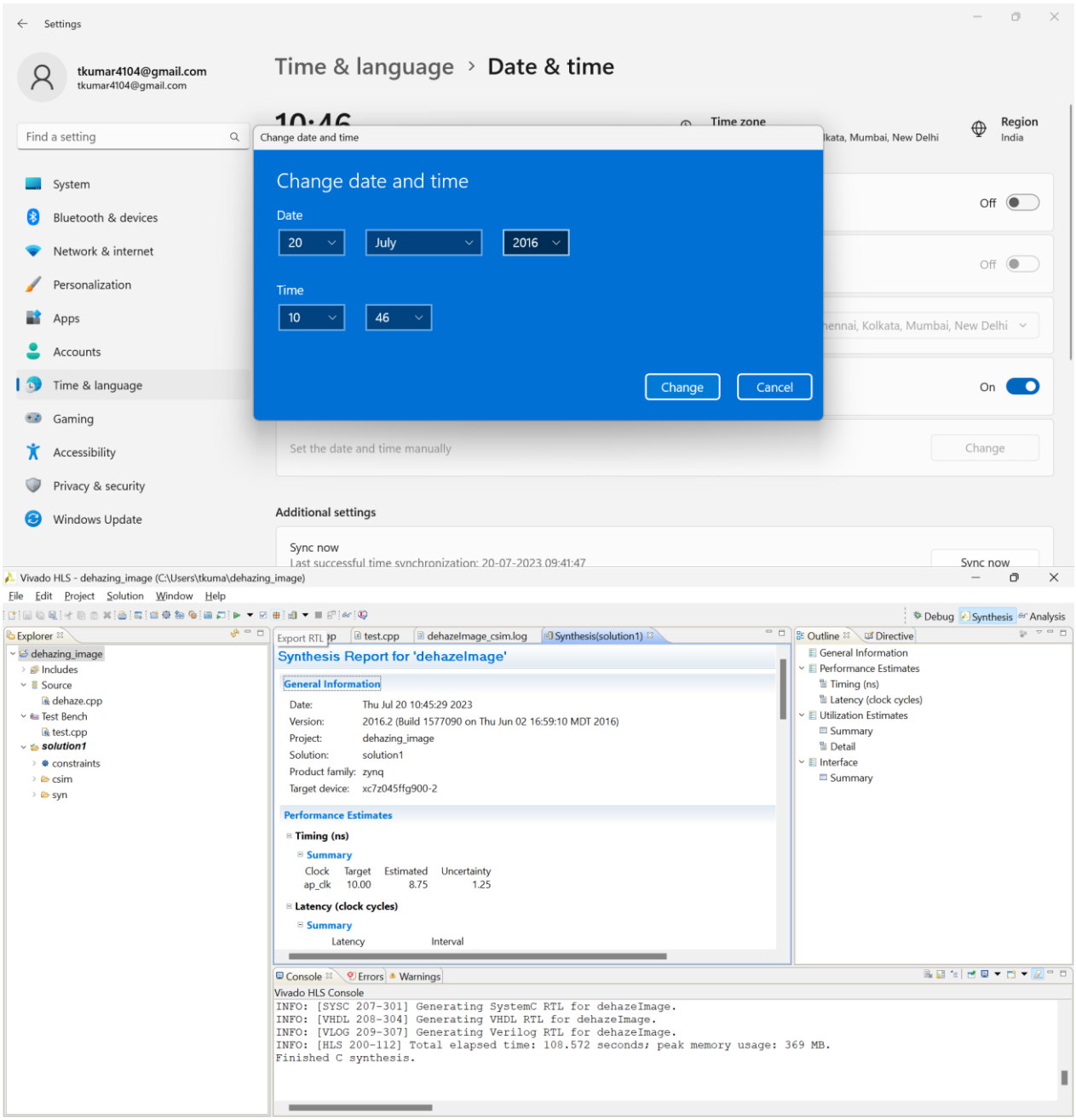


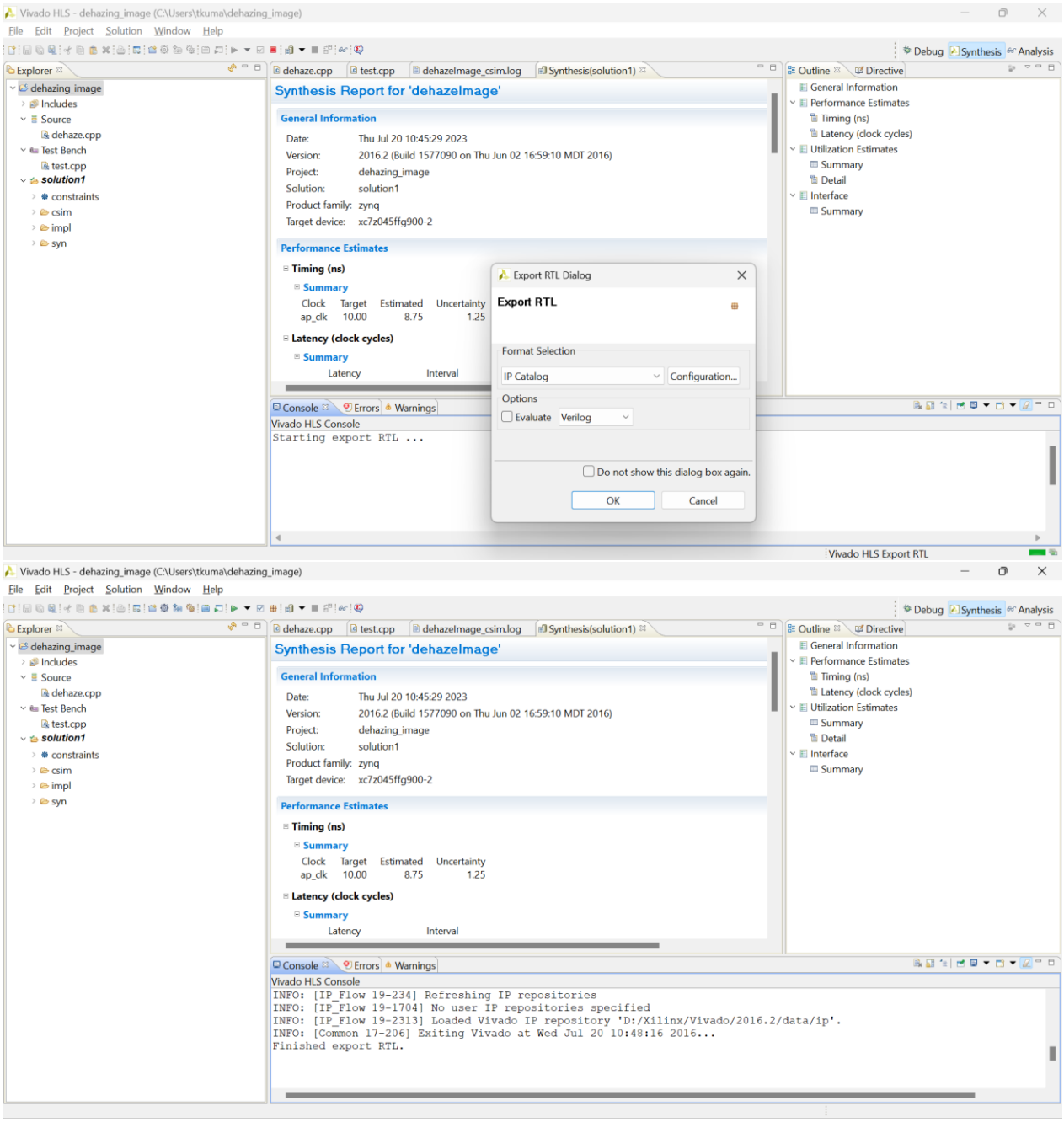

The reason for White boundary layer is absence of preprocessing image

Before Exporting the code into IP, we have changed the system time to 2016, because the libraries, that help In exporting have copyroght issue upto 2016.

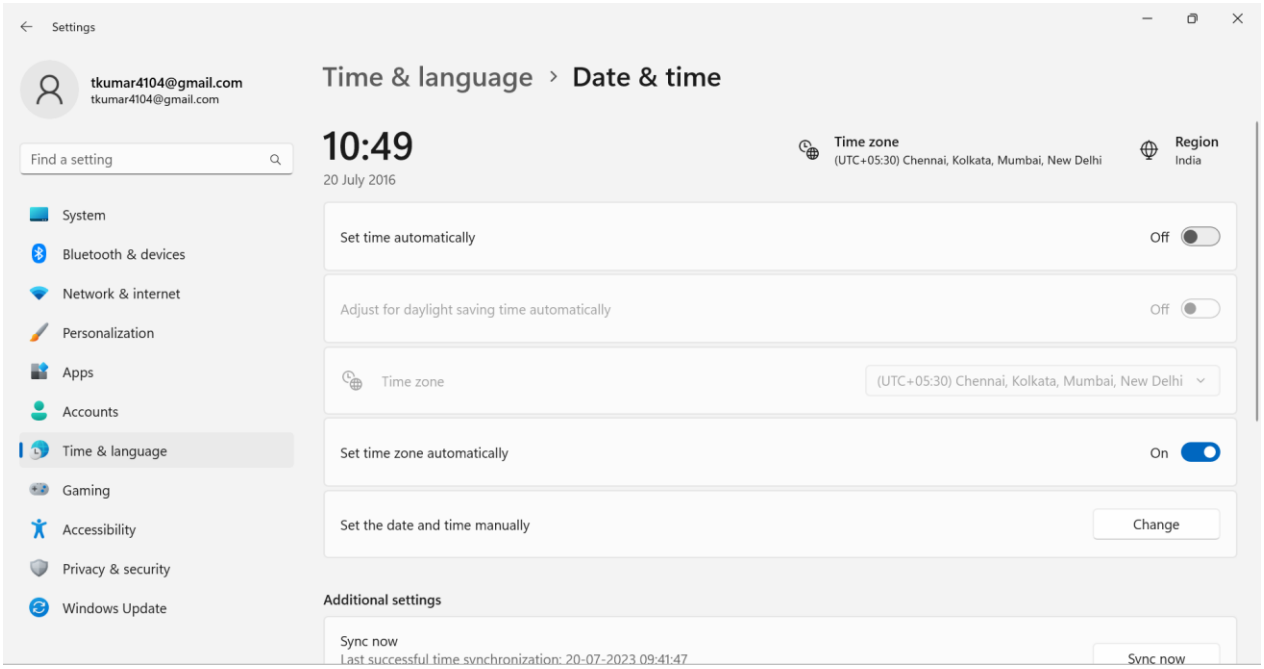

Settings
tkumar4104@gmail.com
tkumar4104@gmail.com
Find a setting
System
Bluetooth & devices
Network & internet
Personalization
Apps
Accounts
Time & language
Gaming
Accessibility
Privacy & security
Windows Update
Time & language
10:46
20 July 2023
Time zone
(UTC+05:30) Chennai, Kolkata, Mumbai, New Delhi
Region
India
Date & time
Time zones, automatic clock settings, calendar display
Language & region
Windows display language, preferred languages, regional formats
Typing
Touch keyboard, text suggestions, preferences
Speech
Speech language, speech recognition microphone setup, voices


Settings
tkumar4104@gmail.com
tkumar4104@gmail.com
Find a setting
System
Bluetooth & devices
Network & internet
Personalization
Apps
Accounts
Time & language
Gaming
Accessibility
Privacy & security
Windows Update
Time & language › Date & time
10:46
20 July 2023
Time zone
(UTC+05:30) Chennai, Kolkata, Mumbai, New Delhi
Region
India
Set time automatically
Off
Adjust for daylight saving time automatically
Off
Time zone
(UTC+05:30) Chennai, Kolkata, Mumbai, New Delhi
Set time zone automatically
On
Set the date and time manually
Change
Additional settings
Sync now

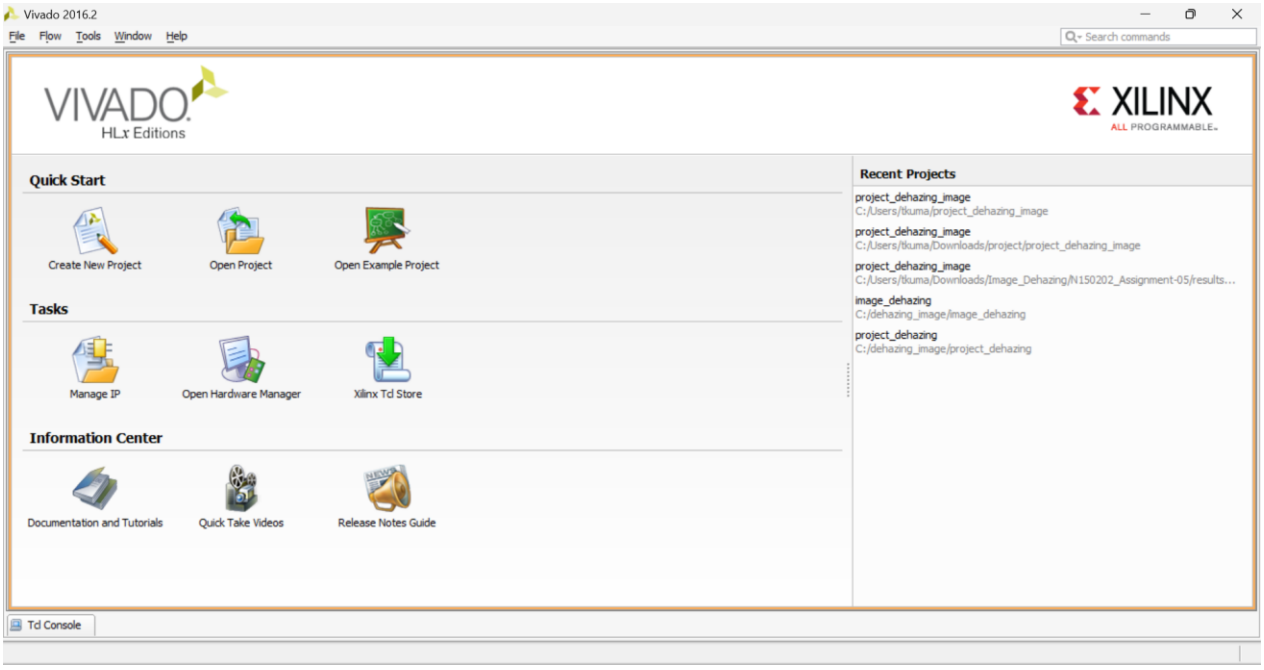

Settings
tkumar4104@gmail.com
tkumar4104@gmail.com
Time & language › Date & time
Find a setting
System
Bluetooth & devices
Network & internet
Personalization
Apps
Accounts
Time & language
Gaming
Accessibility
Privacy & security
Windows Update
Time zone
Region
India
Change date and time
Change date and time
Date
20
July
2016
Time
10
46
Change
Cancel
Off
Off
On
Set the date and time manually
Change
Additional settings
Sync now
Sync now
Vivado HLS - dehazing_image (C:\Users\tkuma\dehazing_image)
File Edit Project Solution Window Help
Debug Synthesis Analysis
Explorer
dehazing_image
Includes
Source
dehaze.cpp
Test Bench
test.cpp
solution1
constraints
csim
syn
Export RTL
test.cpp
dehazeImage_csim.log
Synthesis(solution1)
Synthesis Report for 'dehazeImage'
General Information
Date: Thu Jul 20 10:45:29 2023
Version: 2016.2 (Build 1577090 on Thu Jun 02 16:59:10 MDT 2016)
Project: dehazing_image
Solution: solution1
Product family: zynq
Target device: xc7z045ffg900-2
Performance Estimates
Timing (ns)
Summary
Clock Target Estimated Uncertainty
ap_clk 10.00 8.75 1.25
Latency (clock cycles)
Summary
Latency Interval
Outline
Directive
General Information
Performance Estimates
Timing (ns)
Latency (clock cycles)
Utilization Estimates
Summary
Detail
Interface
Summary
Console
Errors
Warnings
Vivado HLS Console
INFO: [SYSC 207-301] Generating SystemC RTL for dehazeImage.
INFO: [VHDL 208-304] Generating VHDL RTL for dehazeImage.
INFO: [VLOG 209-307] Generating Verilog RTL for dehazeImage.
INFO: [HLS 200-112] Total elapsed time: 108.572 seconds; peak memory usage: 369 MB.
Finished C synthesis.

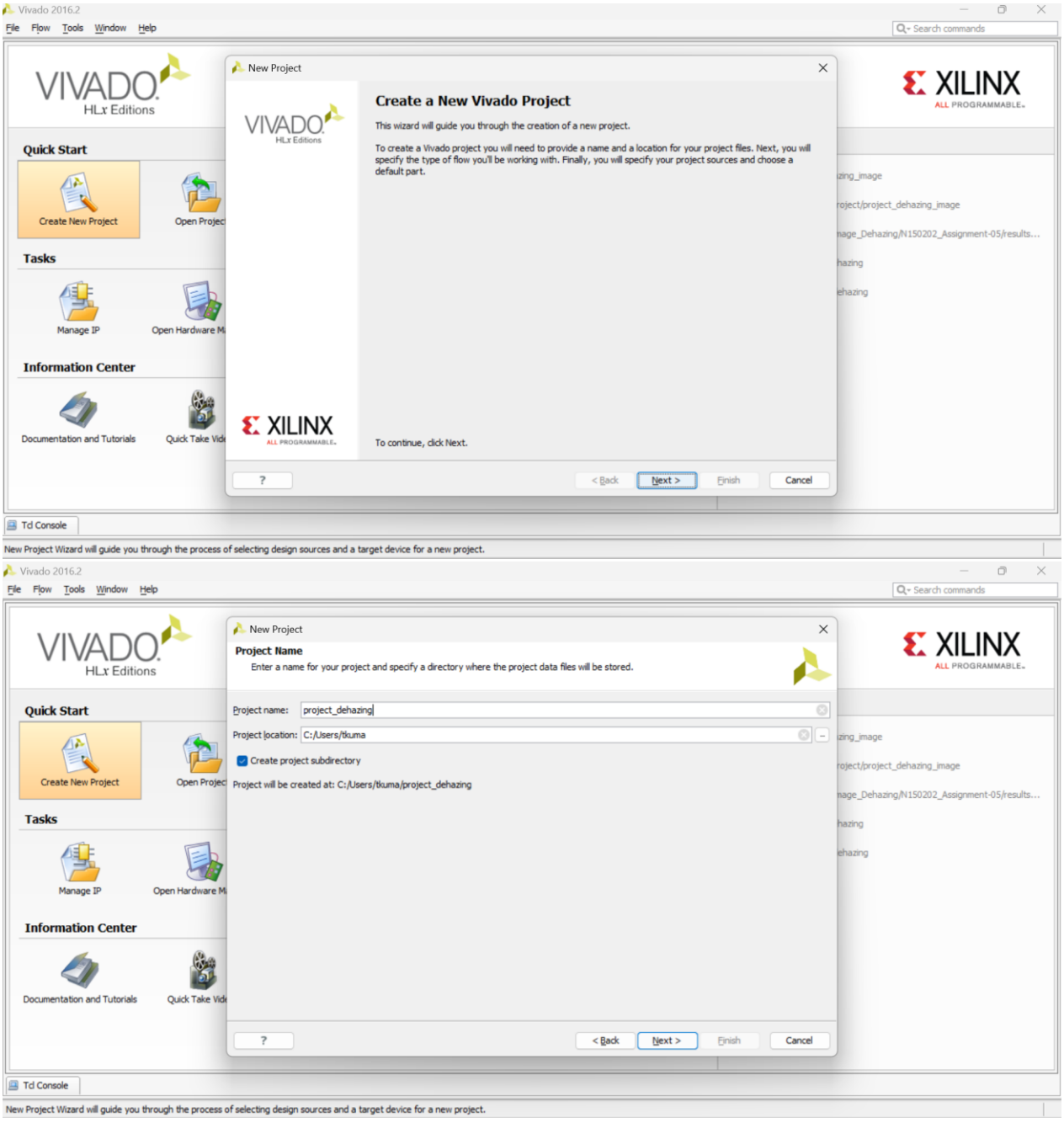

Vivado HLS - dehazing_image (C:\Users\tkuma\dehazing_image)
File Edit Project Solution Window Help
Debug Synthesis Analysis
Explorer
dehazing_image
Includes
Source
dehaze.cpp
Test Bench
test.cpp
solution1
constraints
csim
impl
syn
dehaze.cpp test.cpp dehazeImage_csim.log Synthesis(solution1)
Synthesis Report for 'dehazeImage'
General Information
Date: Thu Jul 20 10:45:29 2023
Version: 2016.2 (Build 1577090 on Thu Jun 02 16:59:10 MDT 2016)
Project: dehazing_image
Solution: solution1
Product family: zynq
Target device: xc7z045ffg900-2
Performance Estimates
Timing (ns)
Summary
Clock Target Estimated Uncertainty
ap_clk 10.00 8.75 1.25
Latency (clock cycles)
Summary
Latency Interval
Export RTL Dialog
Export RTL
Format Selection
IP Catalog
Configuration...
Options
Evaluate
Verilog
Do not show this dialog box again.
OK
Cancel
Outline Directive
General Information
Performance Estimates
Timing (ns)
Latency (clock cycles)
Utilization Estimates
Summary
Detail
Interface
Summary
Console Errors Warnings
Vivado HLS Console
Starting export RTL ...
Vivado HLS Export RTL
Vivado HLS - dehazing_image (C:\Users\tkuma\dehazing_image)
File Edit Project Solution Window Help
Debug Synthesis Analysis
Explorer
dehazing_image
Includes
Source
dehaze.cpp
Test Bench
test.cpp
solution1
constraints
csim
impl
syn
dehaze.cpp test.cpp dehazeImage_csim.log Synthesis(solution1)
Synthesis Report for 'dehazeImage'
General Information
Date: Thu Jul 20 10:45:29 2023
Version: 2016.2 (Build 1577090 on Thu Jun 02 16:59:10 MDT 2016)
Project: dehazing_image
Solution: solution1
Product family: zynq
Target device: xc7z045ffg900-2
Performance Estimates
Timing (ns)
Summary
Clock Target Estimated Uncertainty
ap_clk 10.00 8.75 1.25
Latency (clock cycles)
Summary
Latency Interval
Outline Directive
General Information
Performance Estimates
Timing (ns)
Latency (clock cycles)
Utilization Estimates
Summary
Detail
Interface
Summary
Console Errors Warnings
Vivado HLS Console
INFO: [IP_Flow 19-234] Refreshing IP repositories
INFO: [IP_Flow 19-1704] No user IP repositories specified
INFO: [IP_Flow 19-2313] Loaded Vivado IP repository 'D:/Xilinx/Vivado/2016.2/data/ip'.
INFO: [Common 17-206] Exiting Vivado at Wed Jul 20 10:48:16 2016...
Finished export RTL.

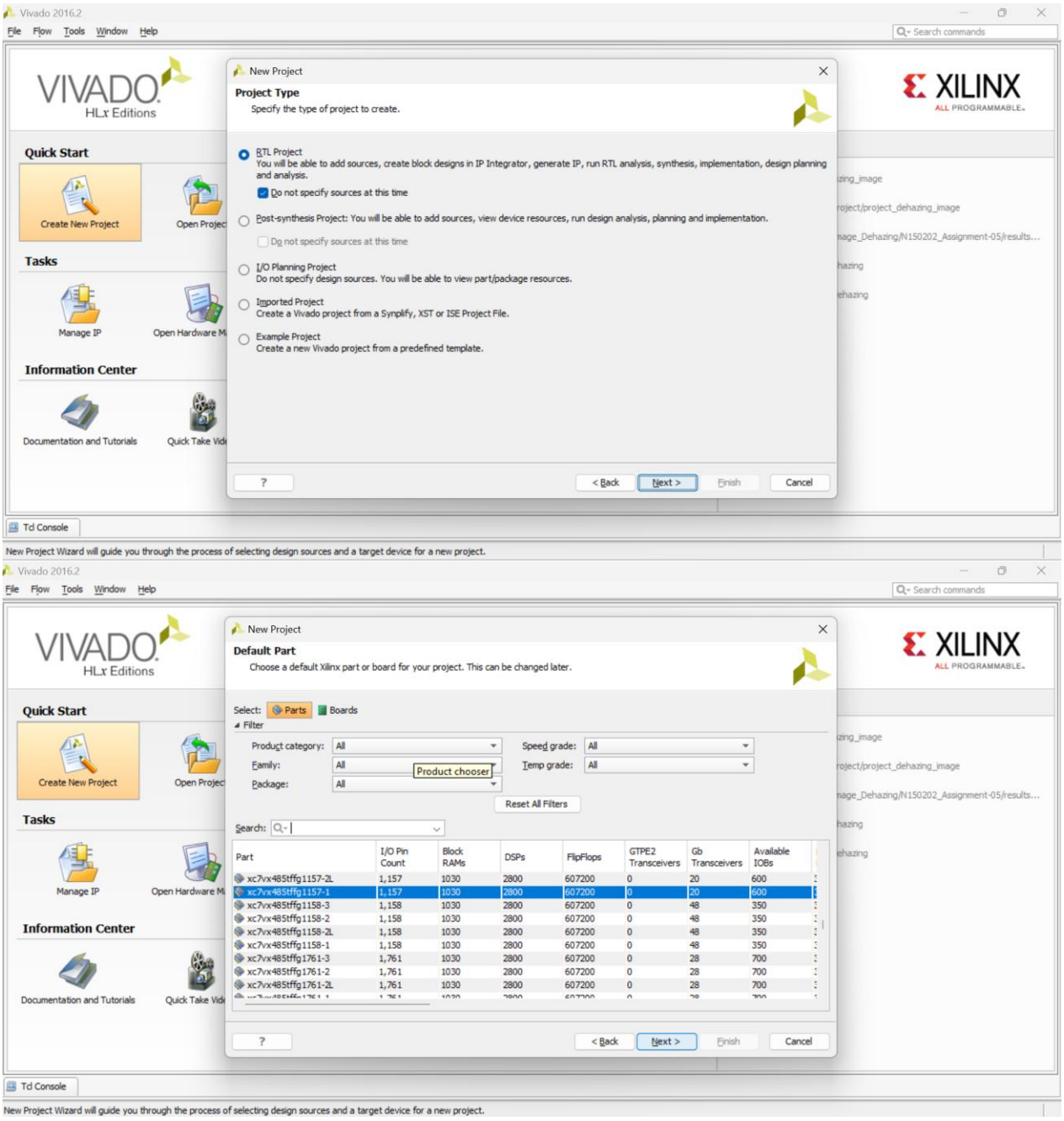
After exporting the code into IP, we have updated the time so that our license is valid for vivado

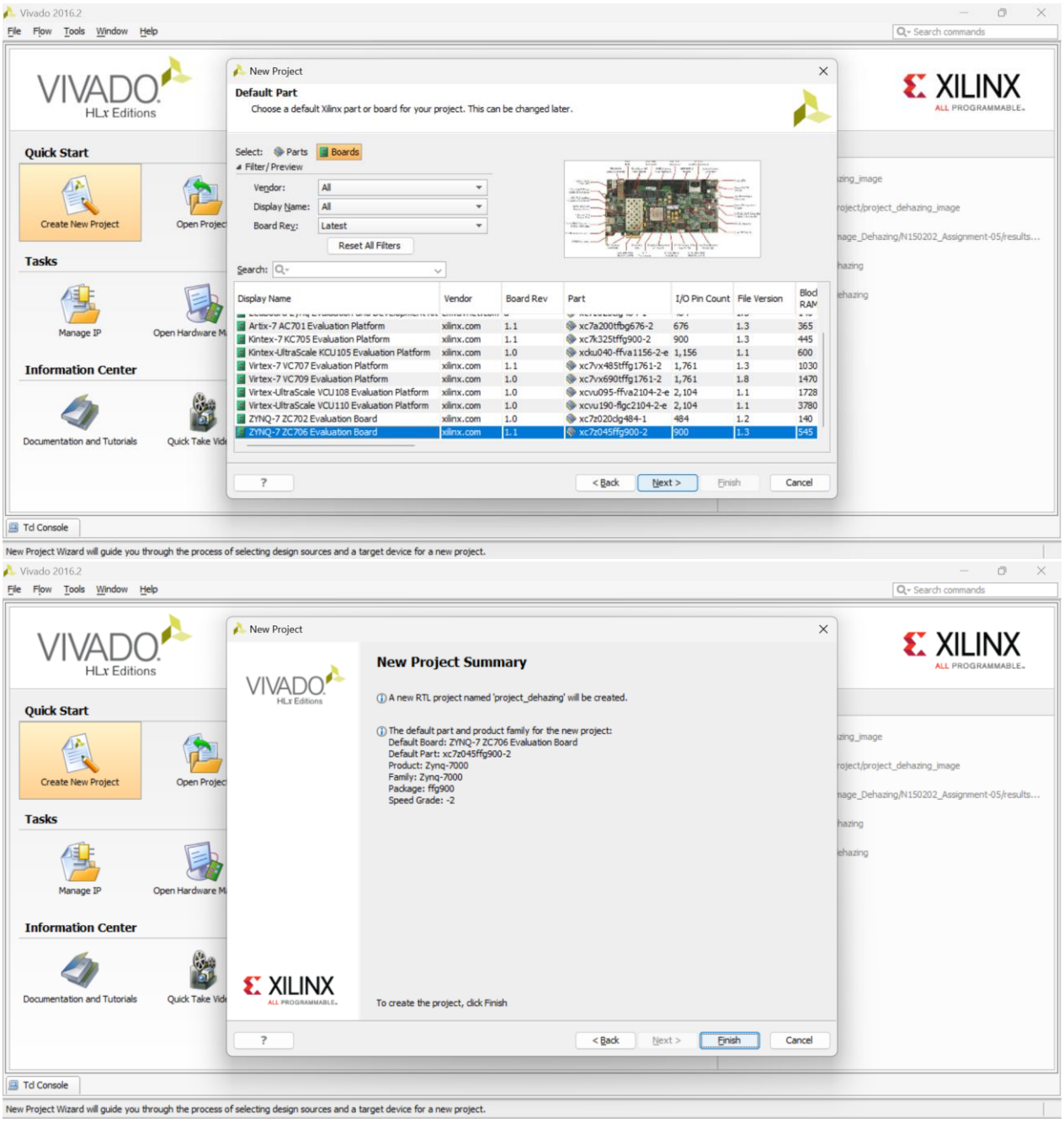



Oen Vivado and click create new project

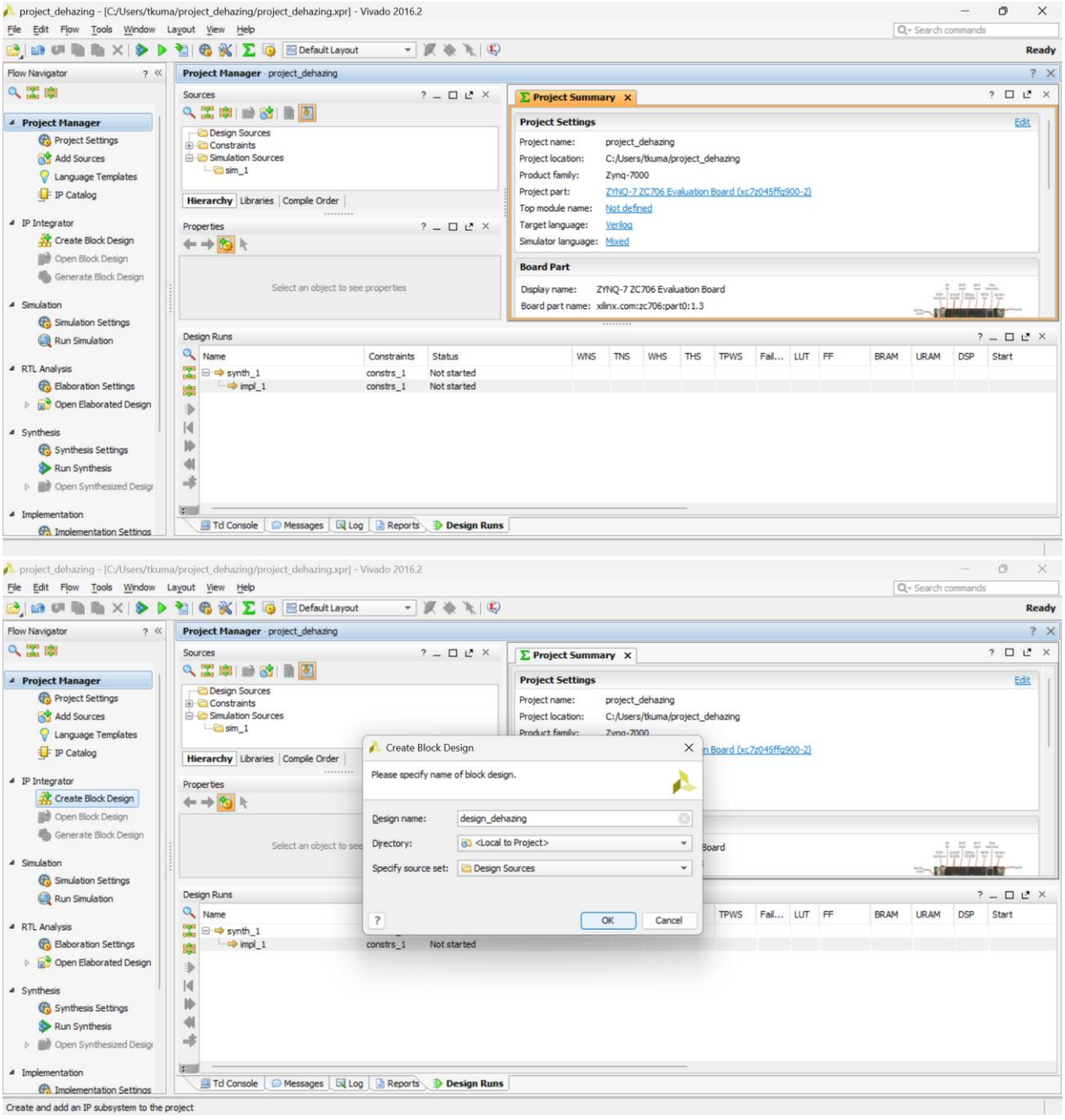



Vivado 2016.2
File Flow Tools Window Help
Search commands
VIVADO HLx Editions
XILINX ALL PROGRAMMABLE
Quick Start
Create New Project
Open Project
Tasks
Manage IP
Open Hardware M
Information Center
Documentation and Tutorials
Quick Take Vid
New Project
Create a New Vivado Project
This wizard will guide you through the creation of a new project.
To create a Vivado project you will need to provide a name and a location for your project files. Next, you will specify the type of flow you'll be working with. Finally, you will specify your project sources and choose a default part.
To continue, click Next.
? < Back Next > Finish Cancel
Tcl Console
New Project Wizard will guide you through the process of selecting design sources and a target device for a 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_dehazing
Project location: C:/Users/tkuma
Create project subdirectory
Project will be created at: C:/Users/tkuma/project_dehazing

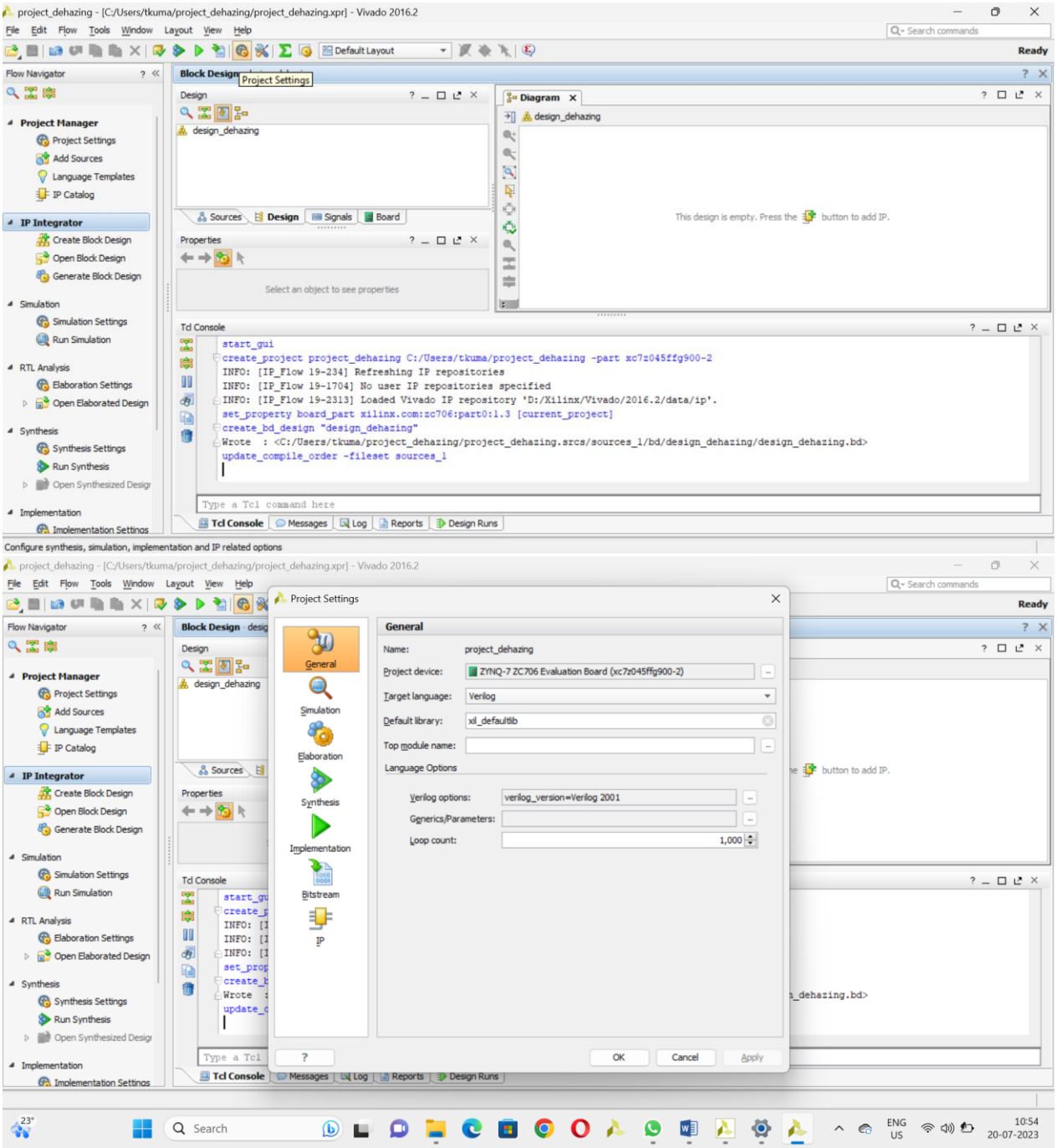

Vivado 2016.2
File Flow Tools Window Help
Search commands
VIVADO HLx Editions
XILINX ALL PROGRAMMABLE
Quick Start
Create New Project
Open Project
Tasks
Manage IP
Open Hardware M
Information Center
Documentation and Tutorials
Quick Take Vid
New Project
Project Type
Specify the type of project to create.
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
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
I/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
Tcl Console
New Project Wizard will guide you through the process of selecting design sources and a target device for a new project.
Default Part
Choose a default Xilinx part or board for your project. This can be changed later.
Select: Parts Boards
Filter
Product category: All
Family: All
Package: All
Speed grade: All
Temp grade: All
Product chooser
Reset All Filters
Search:
Part
I/O Pin Count
Block RAMs
DSPs
FlipFlops
GTPE2 Transceivers
Gb Transceivers
Available IOBs
xc7vx485tffg1157-2L 1,157 1030 2800 607200 0 20 600
xc7vx485tffg1157-1 1,157 1030 2800 607200 0 20 600
xc7vx485tffg1158-3 1,158 1030 2800 607200 0 48 350
xc7vx485tffg1158-2 1,158 1030 2800 607200 0 48 350
xc7vx485tffg1158-2L 1,158 1030 2800 607200 0 48 350
xc7vx485tffg1158-1 1,158 1030 2800 607200 0 48 350
xc7vx485tffg1761-3 1,761 1030 2800 607200 0 28 700
xc7vx485tffg1761-2 1,761 1030 2800 607200 0 28 700
xc7vx485tffg1761-2L 1,761 1030 2800 607200 0 28 700

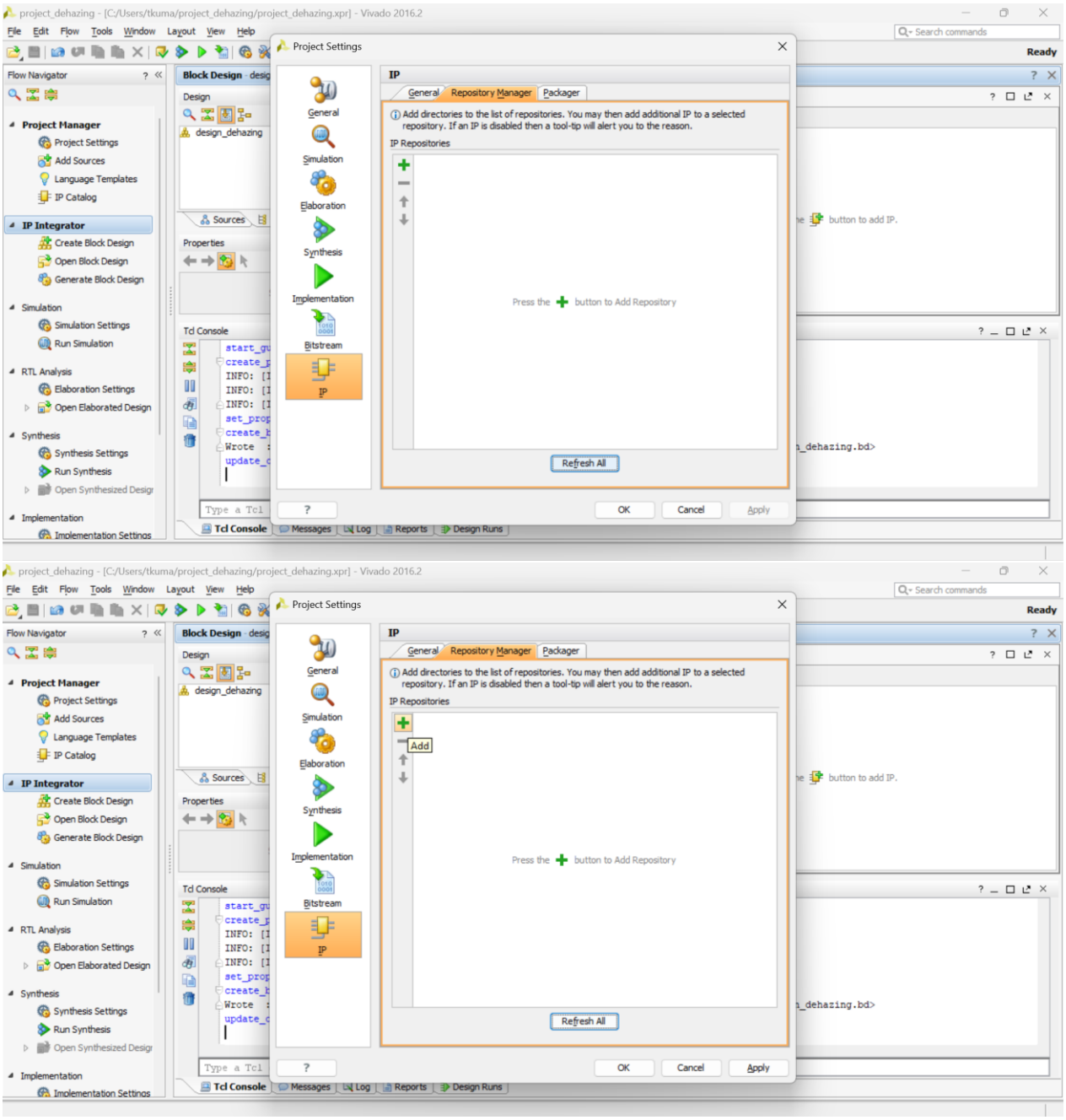



After creating project. Click on create block design for implementing the code to FPGA using AXI Stream interface.

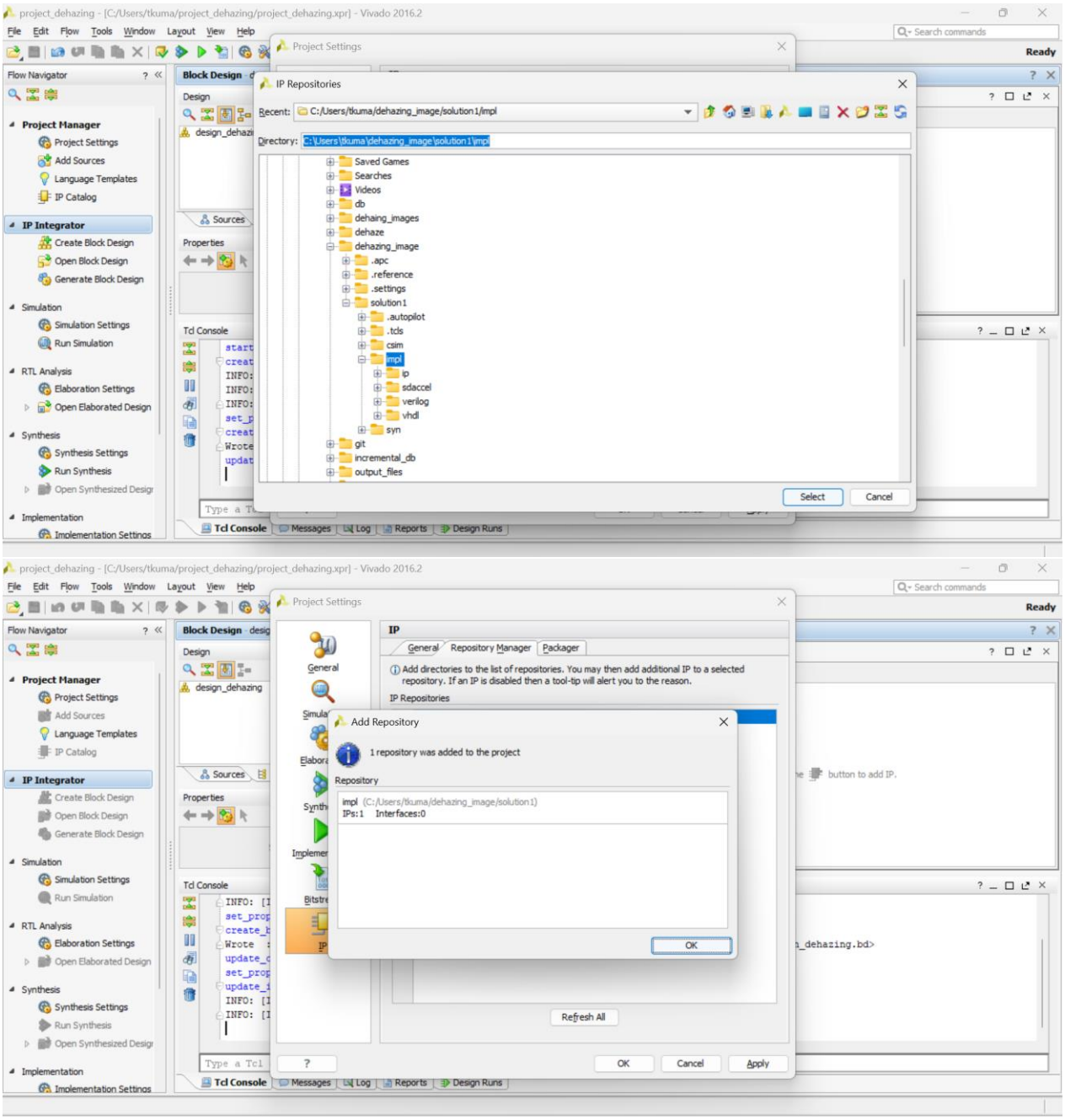

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Project Manager - project_dehazing
Sources
Design Sources
Constraints
Simulation Sources
sim_1
Hierarchy Libraries Compile Order
Properties
Select an object to see properties
Project Summary
Project Settings
Edit
Project name: project_dehazing
Project location: C:/Users/tkuma/project_dehazing
Product family: Zynq-7000
Project part: ZYNQ-7 ZC706 Evaluation Board (xc7z045ffg900-2)
Top module name: Not defined
Target language: Verilog
Simulator language: Mixed
Board Part
Display name: ZYNQ-7 ZC706 Evaluation Board
Board part name: xilinx.com:zc706:part0:1.3
Design Runs
Name Constraints Status WNS TNS WHS THS TPWS Fail... LUT FF BRAM URAM DSP Start
synth_1 constrs_1 Not started
impl_1 constrs_1 Not started
Tcl Console Messages Log Reports Design Runs
Create Block Design
Please specify name of block design.
Design name: design_dehazing
Directory: <Local to Project>
Specify source set: Design Sources
OK Cancel
Create and add an IP subsystem to the project

The below one is the process of adding the exported code IP to out block design.

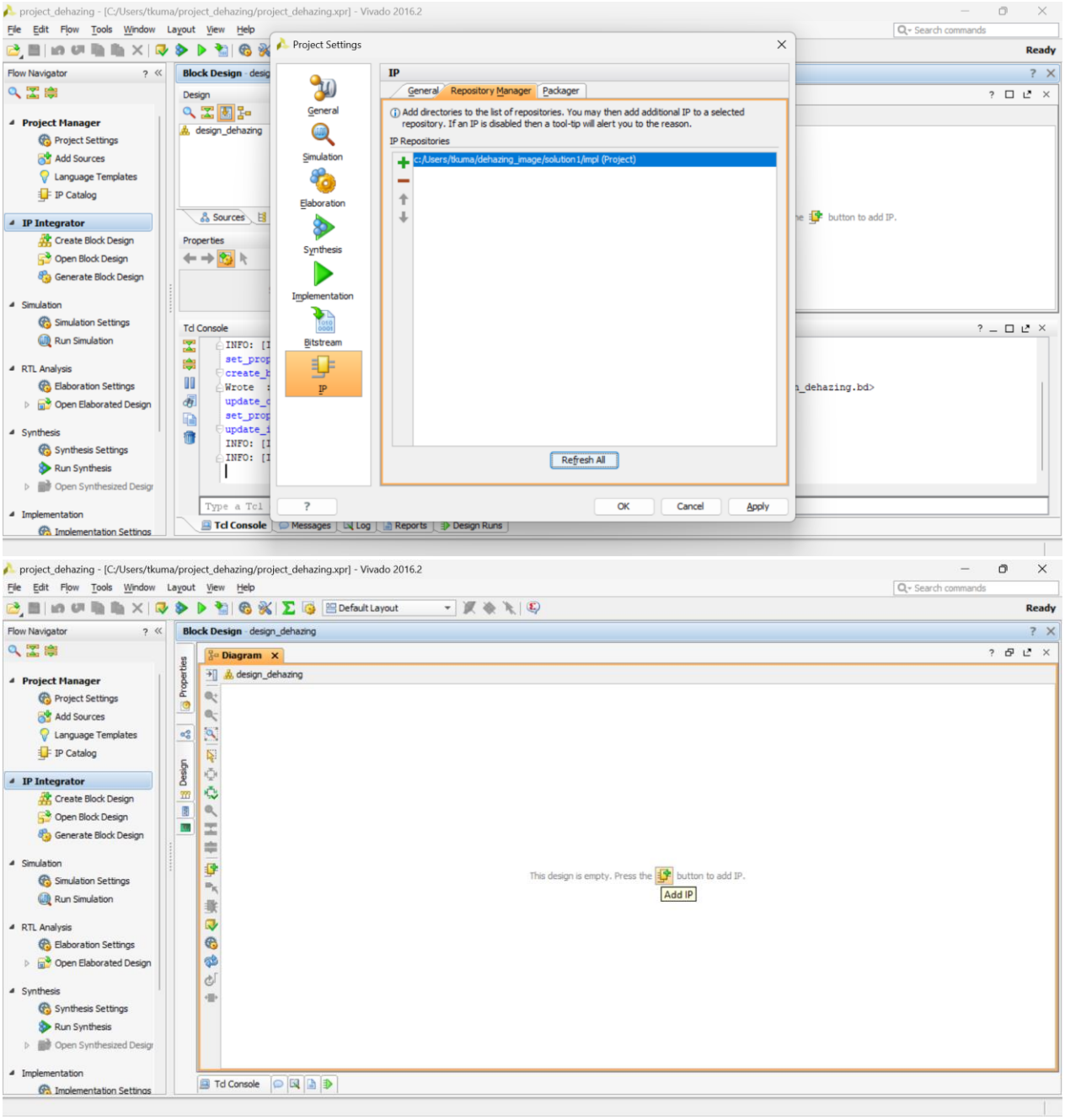



project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Ready
Project Settings
IP
General
Repository Manager
Packager
Add directories to the list of repositories. You may then add additional IP to a selected repository. If an IP is disabled then a tool-tip will alert you to the reason.
IP Repositories
Press the + button to Add Repository
Refresh All
OK
Cancel
Apply
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Implementation
Implementation Settings
Block Design
Design
design_dehazing
Sources
Properties
Tcl Console
Type a Tcl
Tcl Console
Messages
Log
Reports
Design Runs
button to add IP.
_dehazing.bd>

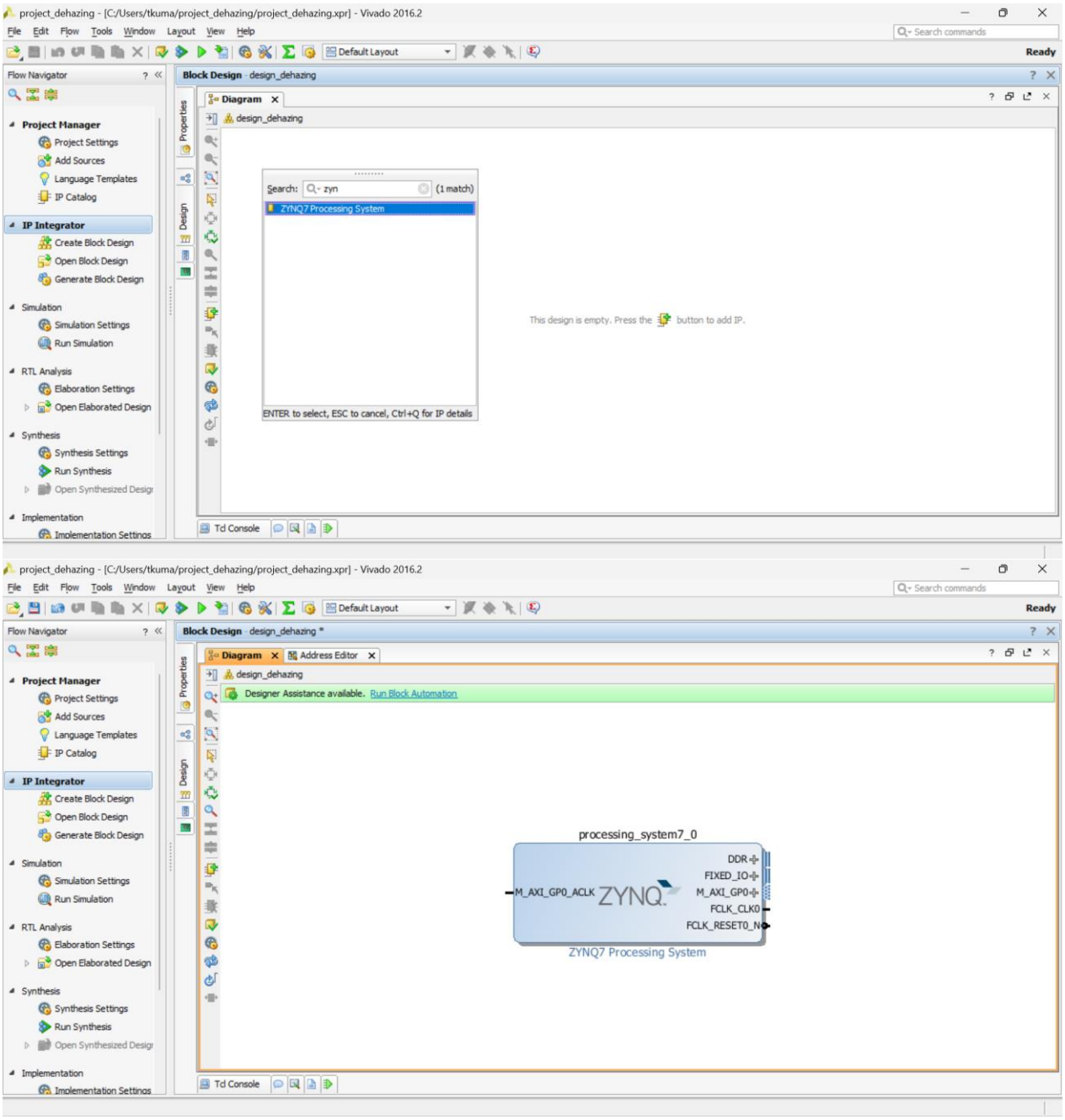

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Ready
Project Settings
IP
General
Repository Manager
Packager
Add directories to the list of repositories. You may then add additional IP to a selected repository. If an IP is disabled then a tool-tip will alert you to the reason.
IP Repositories
Add
Press the + button to Add Repository
Refresh All
OK
Cancel
Apply
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Implementation
Implementation Settings
Block Design
Design
design_dehazing
Sources
Properties
Tcl Console
Type a Tcl
Tcl Console
Messages
Log
Reports
Design Runs
button to add IP.
_dehazing.bd>

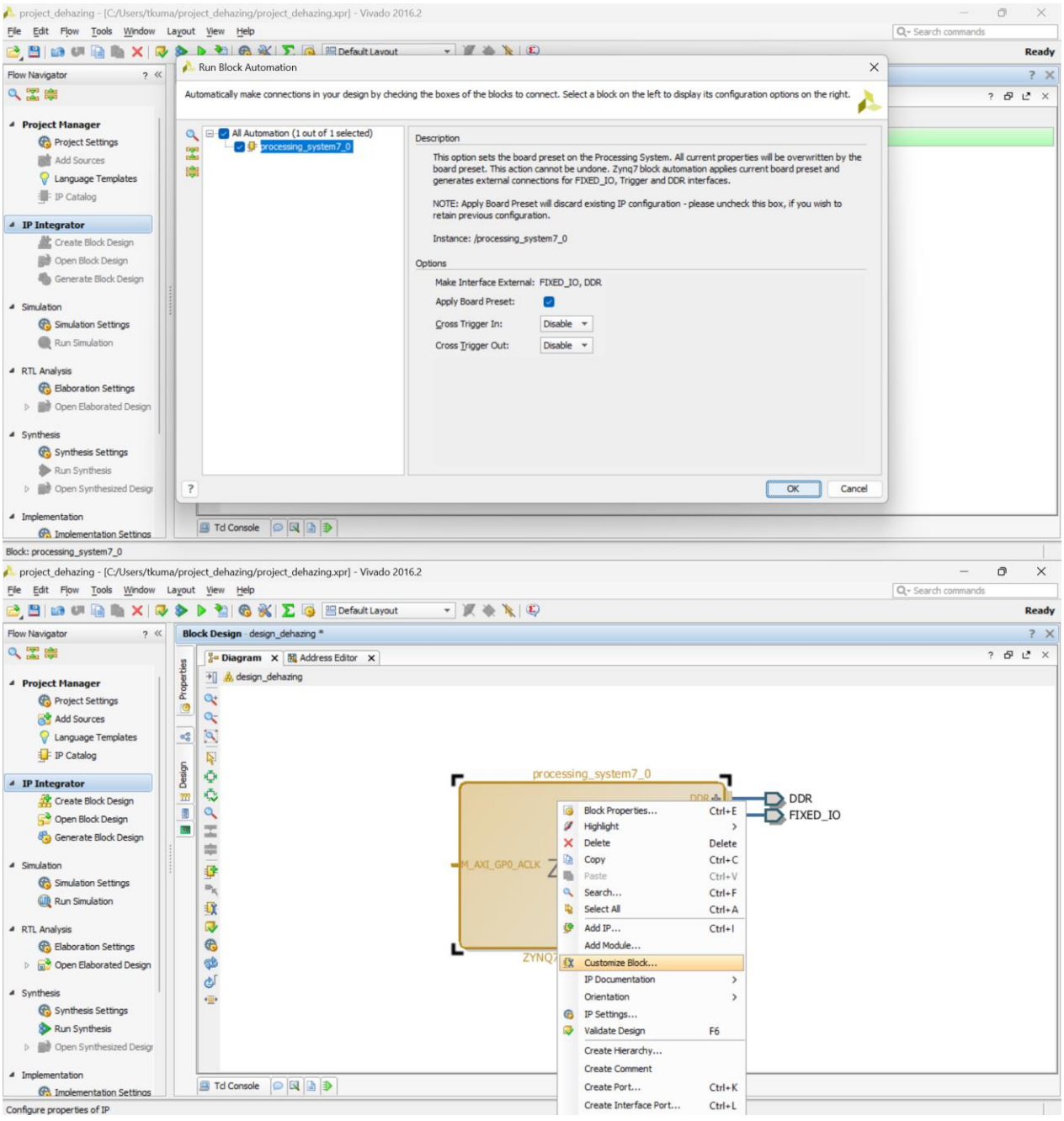

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design
Design
Sources
Properties
Tcl Console
Project Settings
IP Repositories
Recent: C:/Users/tkuma/dehazing_image/solution1/impl
Directory: C:\Users\tkuma\dehazing_image\solution1\impl
Saved Games
Searches
Videos
db
dehaing_images
dehaze
dehazing_image
.apc
.reference
.settings
solution1
.autopilot
.tcls
csim
impl
ip
sdaccel
verilog
vhdl
syn
git
incremental_db
output_files
Select
Cancel
Tcl Console
Messages
Log
Reports
Design Runs

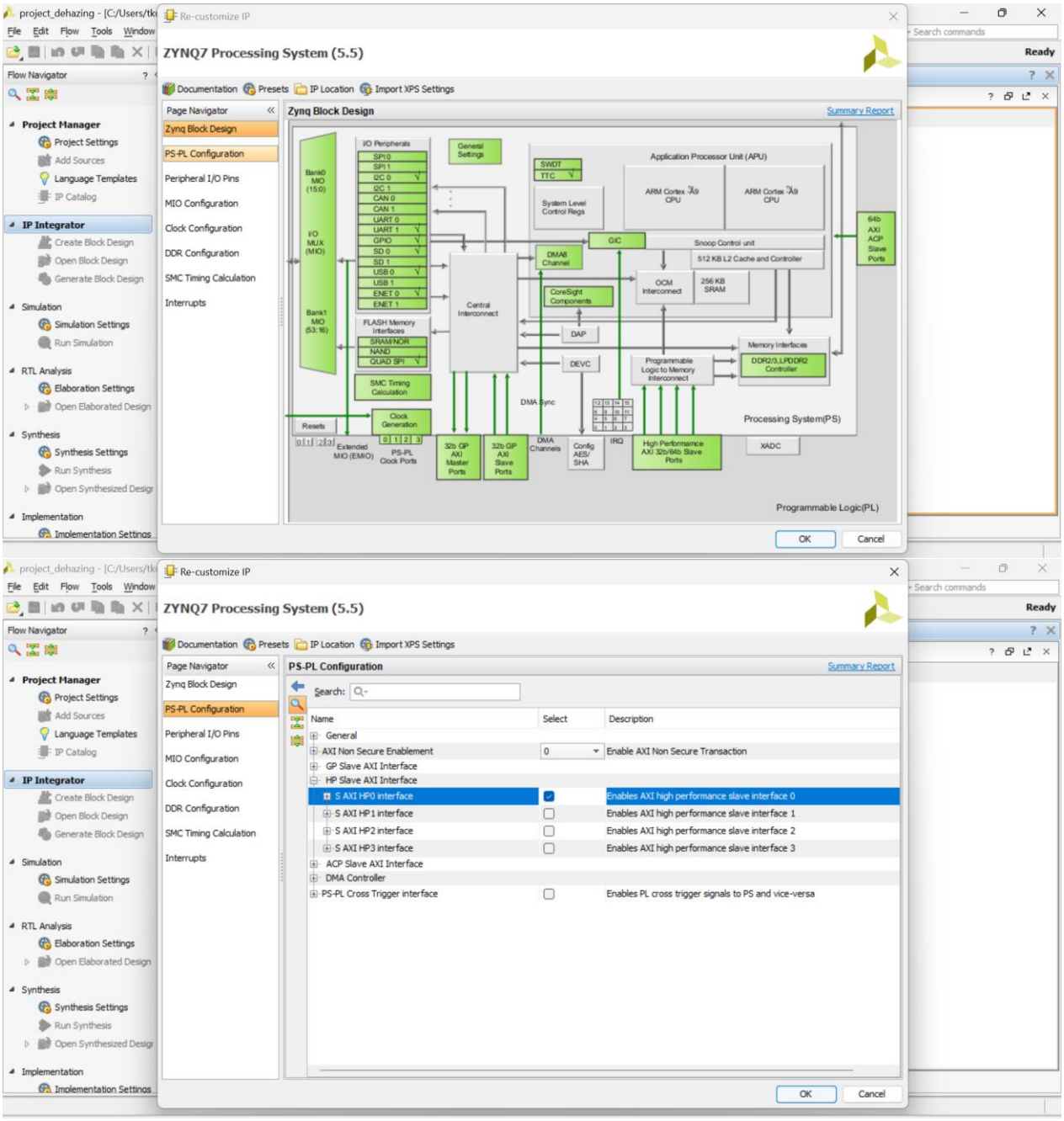

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design
Design
Sources
Properties
Tcl Console
Project Settings
General
IP
General
Repository Manager
Packager
Add directories to the list of repositories. You may then add additional IP to a selected repository. If an IP is disabled then a tool-tip will alert you to the reason.
IP Repositories
Add Repository
1 repository was added to the project
Repository
impl (C:/Users/tkuma/dehazing_image/solution1)
IPs:1 Interfaces:0
OK
Refresh All
OK
Cancel
Apply
Tcl Console
Messages
Log
Reports
Design Runs

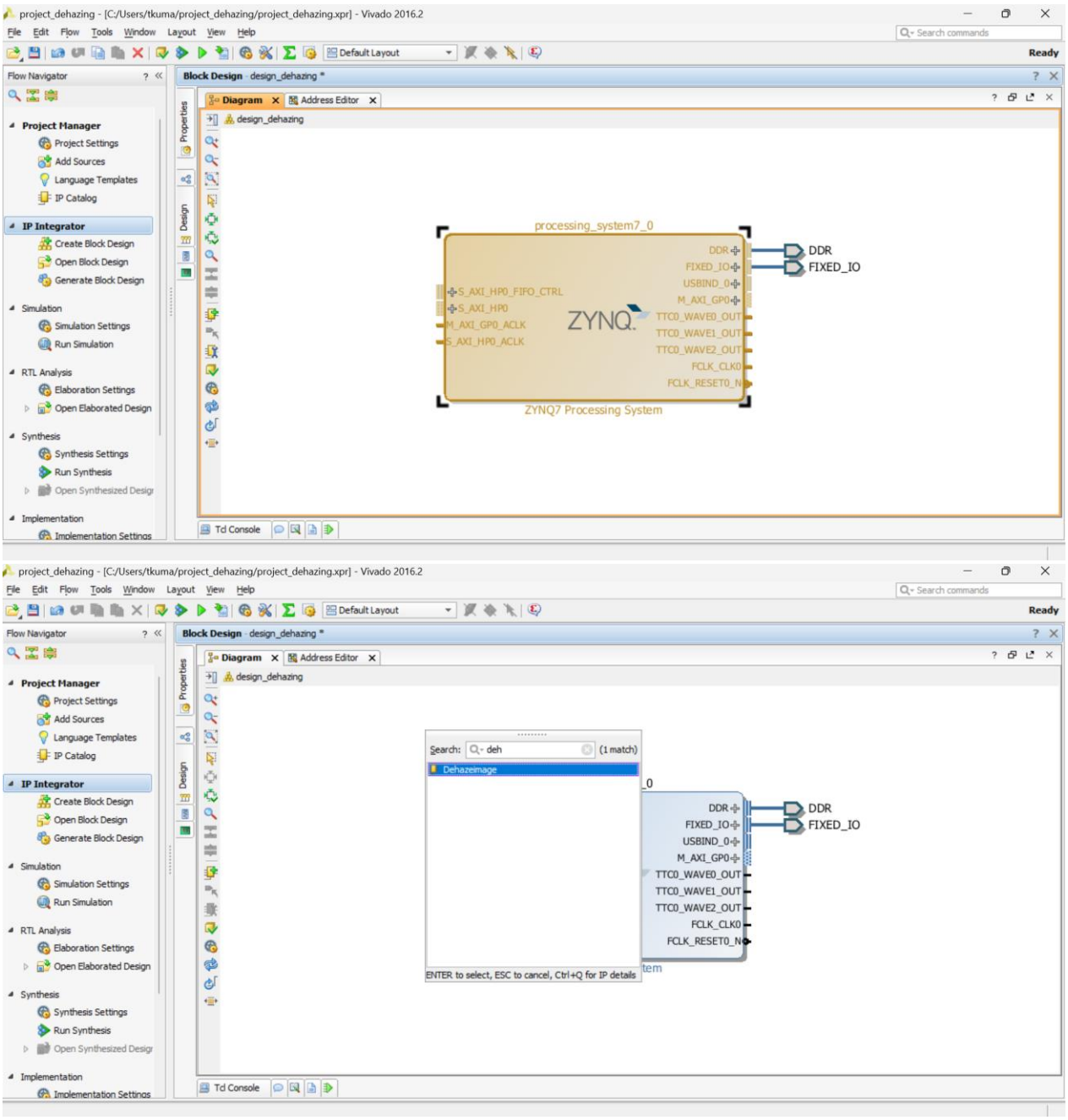


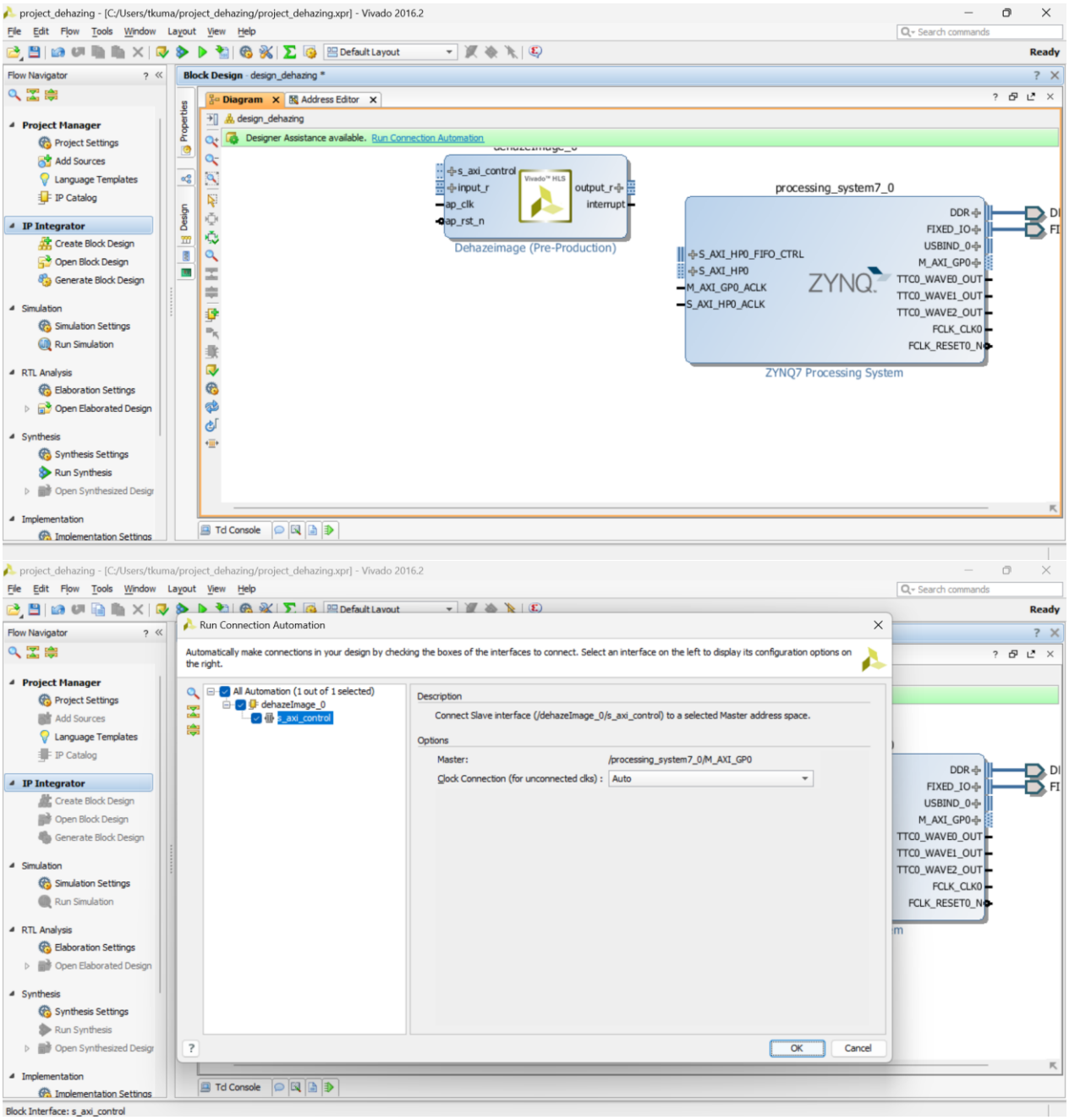

The below the block diagram that is built for AXI Stream Interfece.

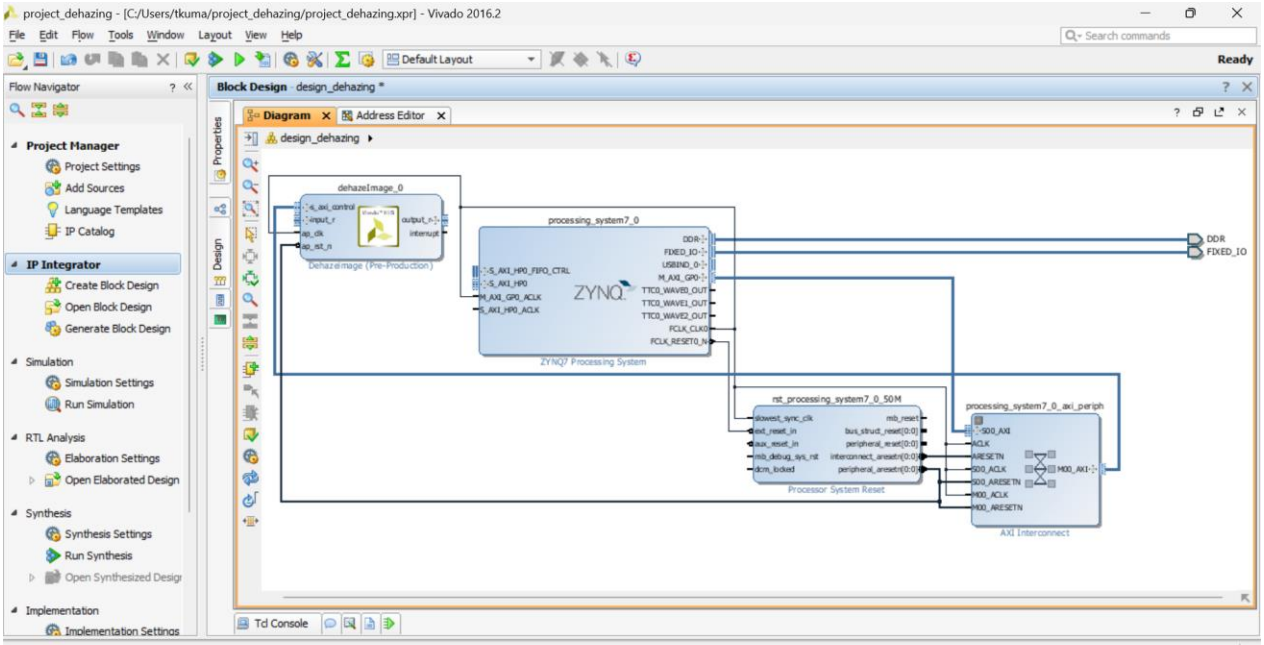

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing
Diagram
design_dehazing
Search: zyn (1 match)
ZYNQ7 Processing System
ENTER to select, ESC to cancel, Ctrl+Q for IP details
This design is empty. Press the button to add IP.
Tcl Console
project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Designer Assistance available. Run Block Automation
processing_system7_0
M_AXI_GP0_ACLK
ZYNQ
DDR
FIXED_IO
M_AXI_GP0
FCLK_CLK0
FCLK_RESET0_N
ZYNQ7 Processing System
Tcl Console

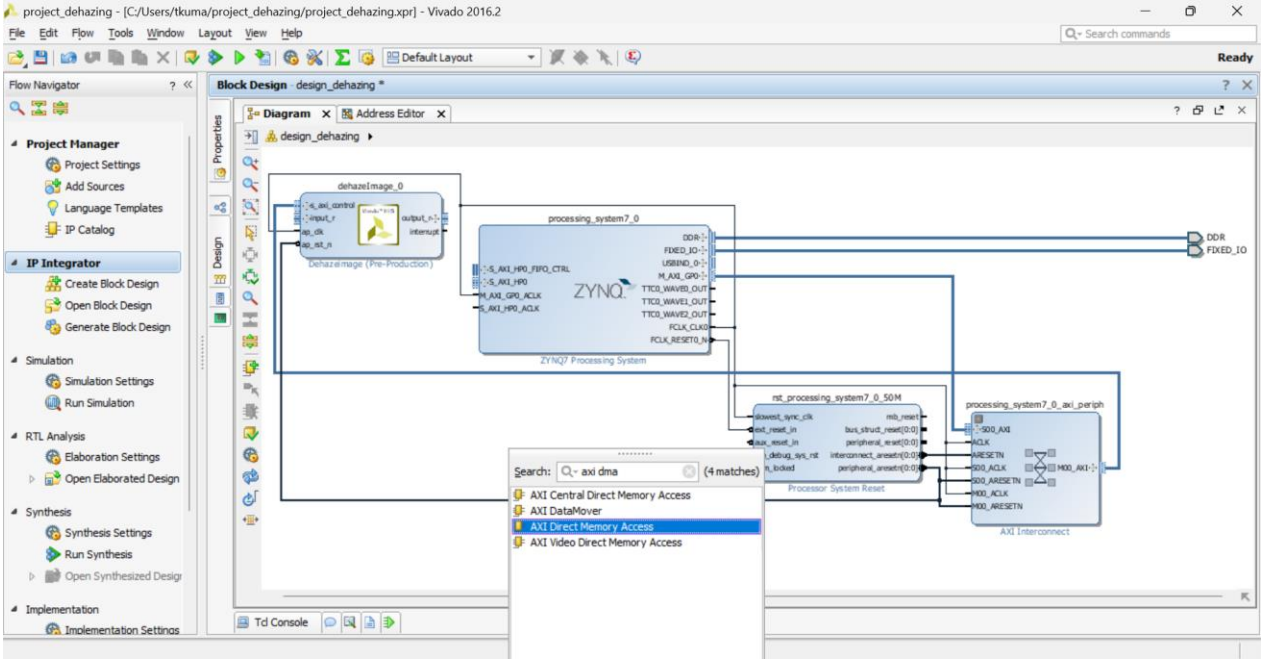

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Ready
Default Layout
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Run Block Automation
Automatically make connections in your design by checking the boxes of the blocks to connect. Select a block on the left to display its configuration options on the right.
All Automation (1 out of 1 selected)
processing_system7_0
Description
This option sets the board preset on the Processing System. All current properties will be overwritten by the board preset. This action cannot be undone. Zynq7 block automation applies current board preset and generates external connections for FIXED_IO, Trigger and DDR interfaces.
NOTE: Apply Board Preset will discard existing IP configuration - please uncheck this box, if you wish to retain previous configuration.
Instance: /processing_system7_0
Options
Make Interface External: FIXED_IO, DDR
Apply Board Preset:
Cross Trigger In: Disable
Cross Trigger Out: Disable
OK
Cancel
Tcl Console
Block: processing_system7_0
project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Properties
Design
processing_system7_0
DDR
FIXED_IO
M_AXI_GP0_ACLK
ZYNQ7
Block Properties... Ctrl+E
Highlight
Delete Delete
Copy Ctrl+C
Paste Ctrl+V
Search... Ctrl+F
Select All Ctrl+A
Add IP... Ctrl+I
Add Module...
Customize Block...
IP Documentation
Orientation
IP Settings...
Validate Design F6
Create Hierarchy...
Create Comment
Create Port... Ctrl+K
Create Interface Port... Ctrl+L
Configure properties of IP

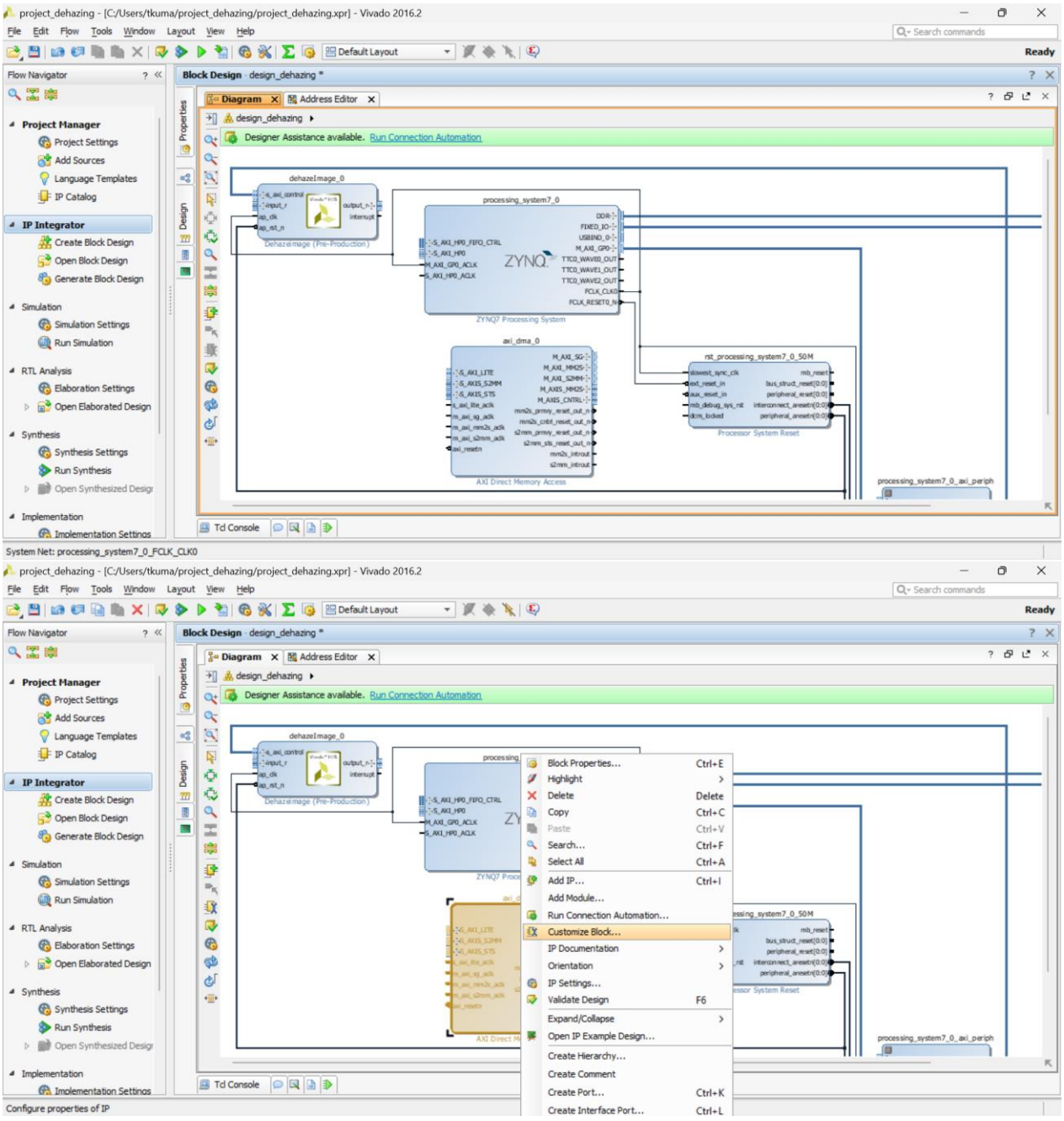

Re-customize IP
ZYNQ7 Processing System (5.5)
Documentation
Presets
IP Location
Import XPS Settings
Page Navigator
Zynq Block Design
PS-PL Configuration
Peripheral I/O Pins
MIO Configuration
Clock Configuration
DDR Configuration
SMC Timing Calculation
Interrupts
Zynq Block Design
Summary Report
Processing System(PS)
Programmable Logic(PL)
OK
Cancel
PS-PL Configuration
Search:
Name
Select
Description
General
AXI Non Secure Enablement
Enable AXI Non Secure Transaction
GP Slave AXI Interface
HP Slave AXI Interface
S AXI HP0 interface
Enables AXI high performance slave interface 0
S AXI HP1 interface
Enables AXI high performance slave interface 1
S AXI HP2 interface
Enables AXI high performance slave interface 2
S AXI HP3 interface
Enables AXI high performance slave interface 3
ACP Slave AXI Interface
DMA Controller
PS-PL Cross Trigger interface
Enables PL cross trigger signals to PS and vice-versa

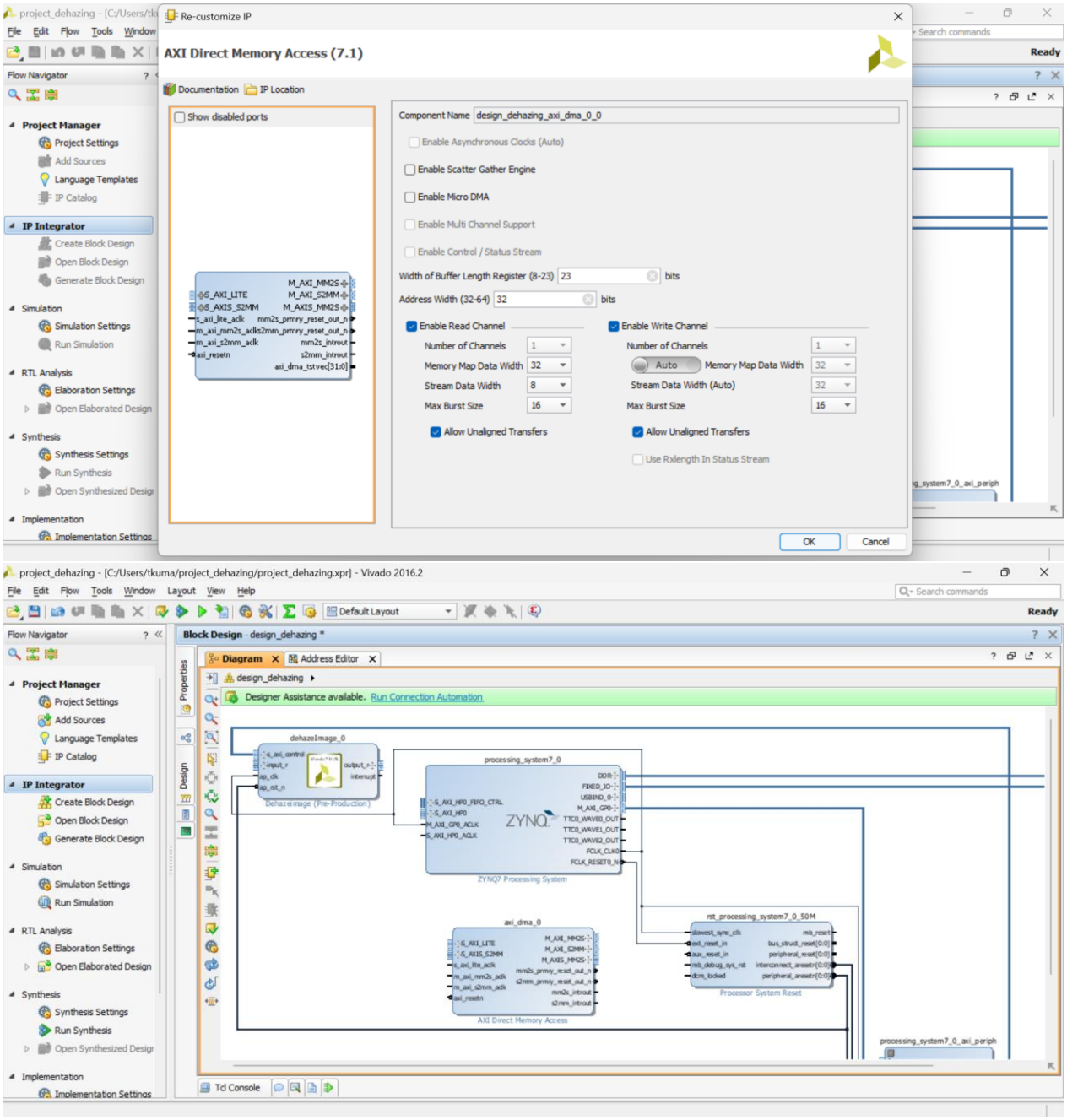

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Properties
Design
processing_system7_0
S_AXI_HP0_FIFO_CTRL
S_AXI_HP0
M_AXI_GP0_ACLK
S_AXI_HP0_ACLK
ZYNQ
DDR
FIXED_IO
USBIND_0
M_AXI_GP0
TTC0_WAVE0_OUT
TTC0_WAVE1_OUT
TTC0_WAVE2_OUT
FCLK_CLK0
FCLK_RESET0_N
ZYNQ7 Processing System
Tcl Console
project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
Search: deh
(1 match)
Dehazeimage
ENTER to select, ESC to cancel, Ctrl+Q for IP details

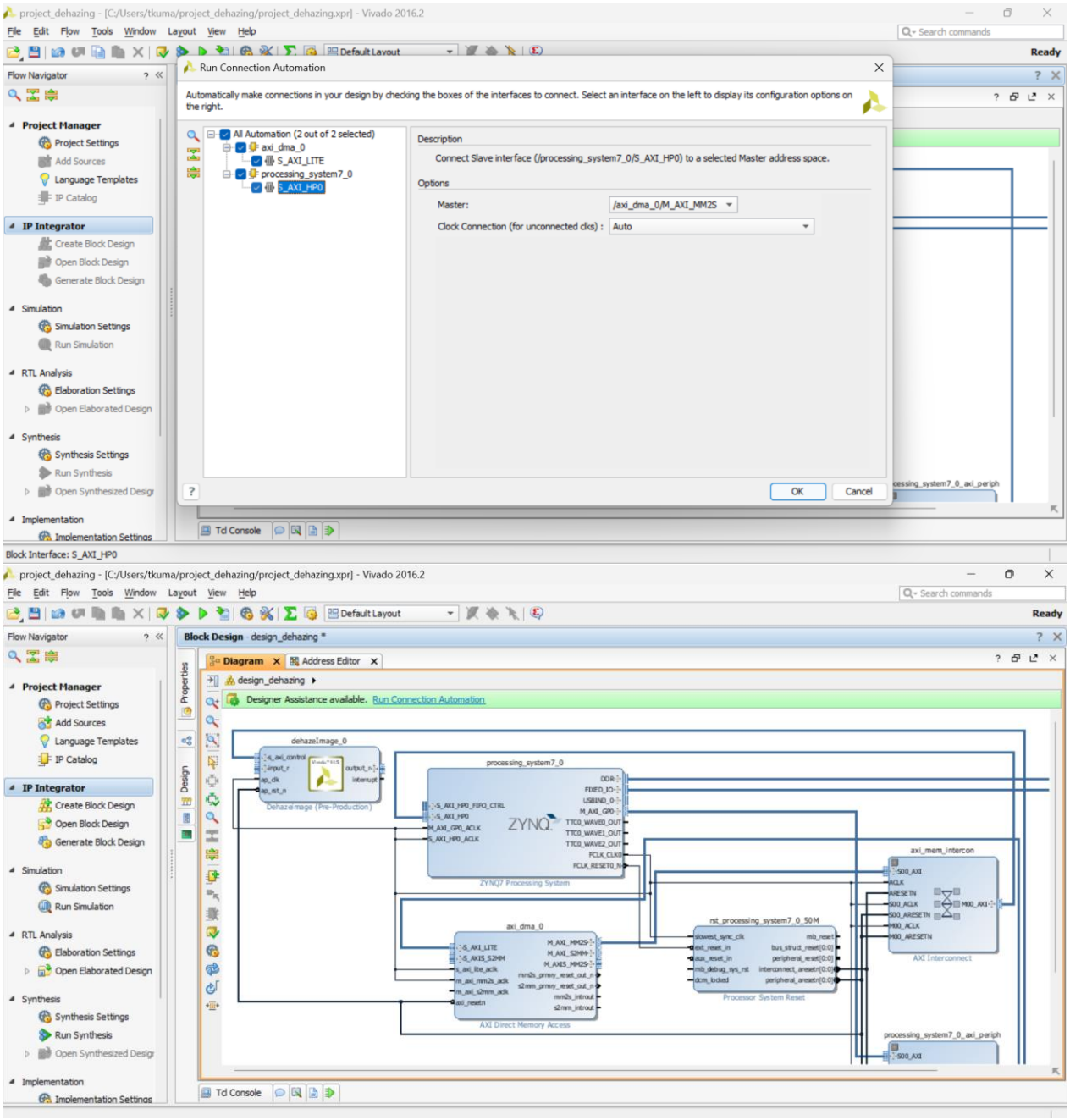

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Designer Assistance available. Run Connection Automation
s_axi_control
input_r
ap_clk
ap_rst_n
output_r
interrupt
Dehazeimage (Pre-Production)
processing_system7_0
S_AXI_HP0_FIFO_CTRL
S_AXI_HP0
M_AXI_GP0_ACLK
S_AXI_HP0_ACLK
ZYNQ
DDR
FIXED_IO
USBIND_0
M_AXI_GP0
TTC0_WAVE0_OUT
TTC0_WAVE1_OUT
TTC0_WAVE2_OUT
FCLK_CLK0
FCLK_RESET0_N
ZYNQ7 Processing System
Tcl Console
Run Connection Automation
Automatically make connections in your design by checking the boxes of the interfaces to connect. Select an interface on the left to display its configuration options on the right.
All Automation (1 out of 1 selected)
dehazeImage_0
s_axi_control
Description
Connect Slave interface (/dehazeImage_0/s_axi_control) to a selected Master address space.
Options
Master:
/processing_system7_0/M_AXI_GP0
Clock Connection (for unconnected clks) :
Auto
OK
Cancel
Block Interface: s_axi_control

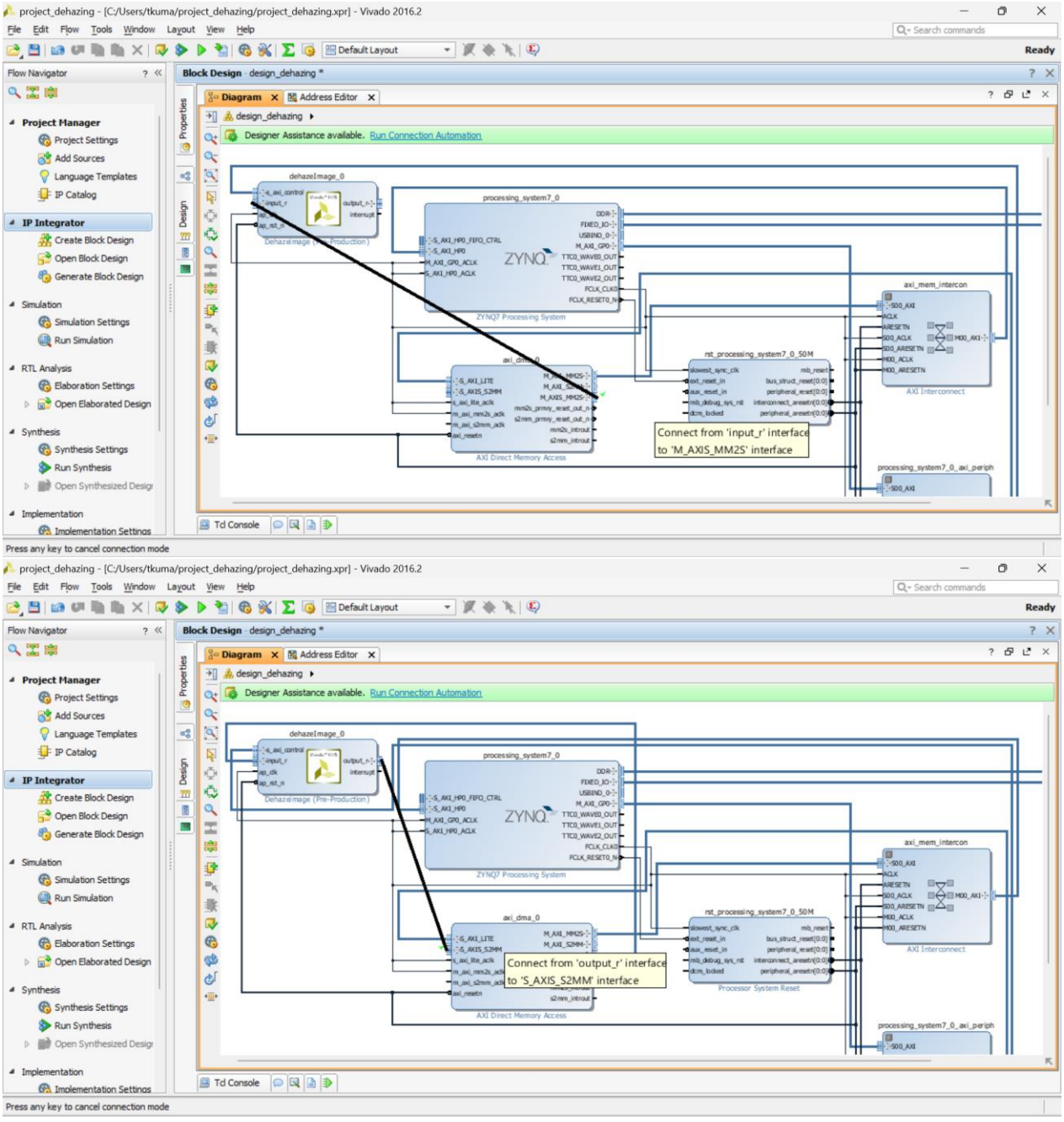

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Properties
Design
dehazeImage_0
DehazeImage (Pre-Production)
processing_system7_0
ZYNQ7 Processing System
rst_processing_system7_0_50M
Processor System Reset
processing_system7_0_axi_periph
AXI Interconnect
DDR
FIXED_IO
Tcl Console


project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Properties
Design
dehazeImage_0
DehazeImage (Pre-Production)
processing_system7_0
ZYNQ7 Processing System
rst_processing_system7_0_50M
Processor System Reset
processing_system7_0_axi_periph
AXI Interconnect
DDR
FIXED_IO
Search: axi dma
(4 matches)
AXI Central Direct Memory Access
AXI DataMover
AXI Direct Memory Access
AXI Video Direct Memory Access
Tcl Console

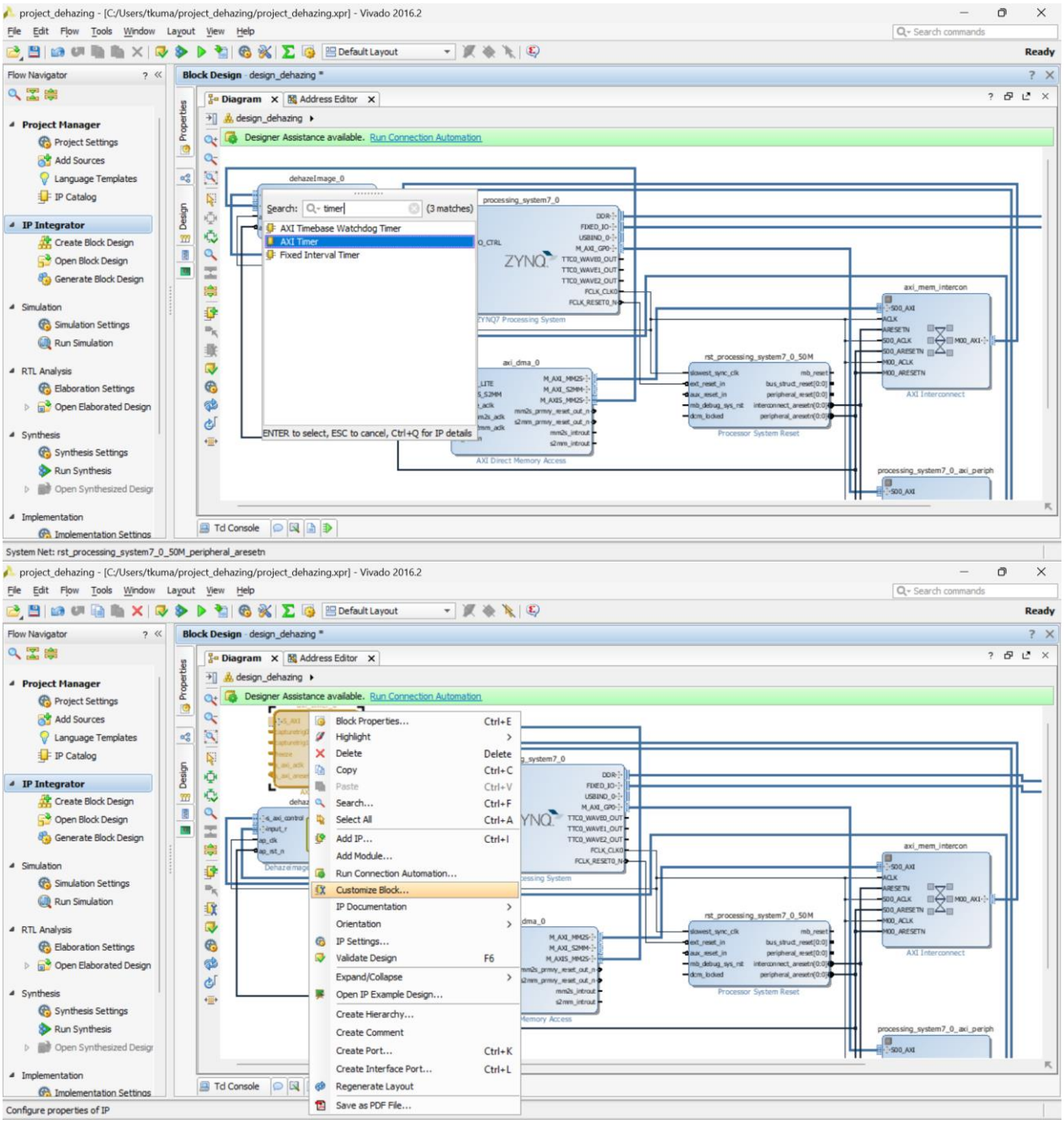

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Designer Assistance available. Run Connection Automation
dehazeImage_0
processing_system7_0
ZYNQ7 Processing System
axi_dma_0
AXI Direct Memory Access
rst_processing_system7_0_50M
Processor System Reset
processing_system7_0_axi_periph
Tcl Console
System Net: processing_system7_0_FCLK_CLK0
project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
Block Properties... Ctrl+E
Highlight
Delete Delete
Copy Ctrl+C
Paste Ctrl+V
Search... Ctrl+F
Select All Ctrl+A
Add IP... Ctrl+I
Add Module...
Run Connection Automation...
Customize Block...
IP Documentation
Orientation
IP Settings...
Validate Design F6
Expand/Collapse
Open IP Example Design...
Create Hierarchy...
Create Comment
Create Port... Ctrl+K
Create Interface Port... Ctrl+L
Configure properties of IP

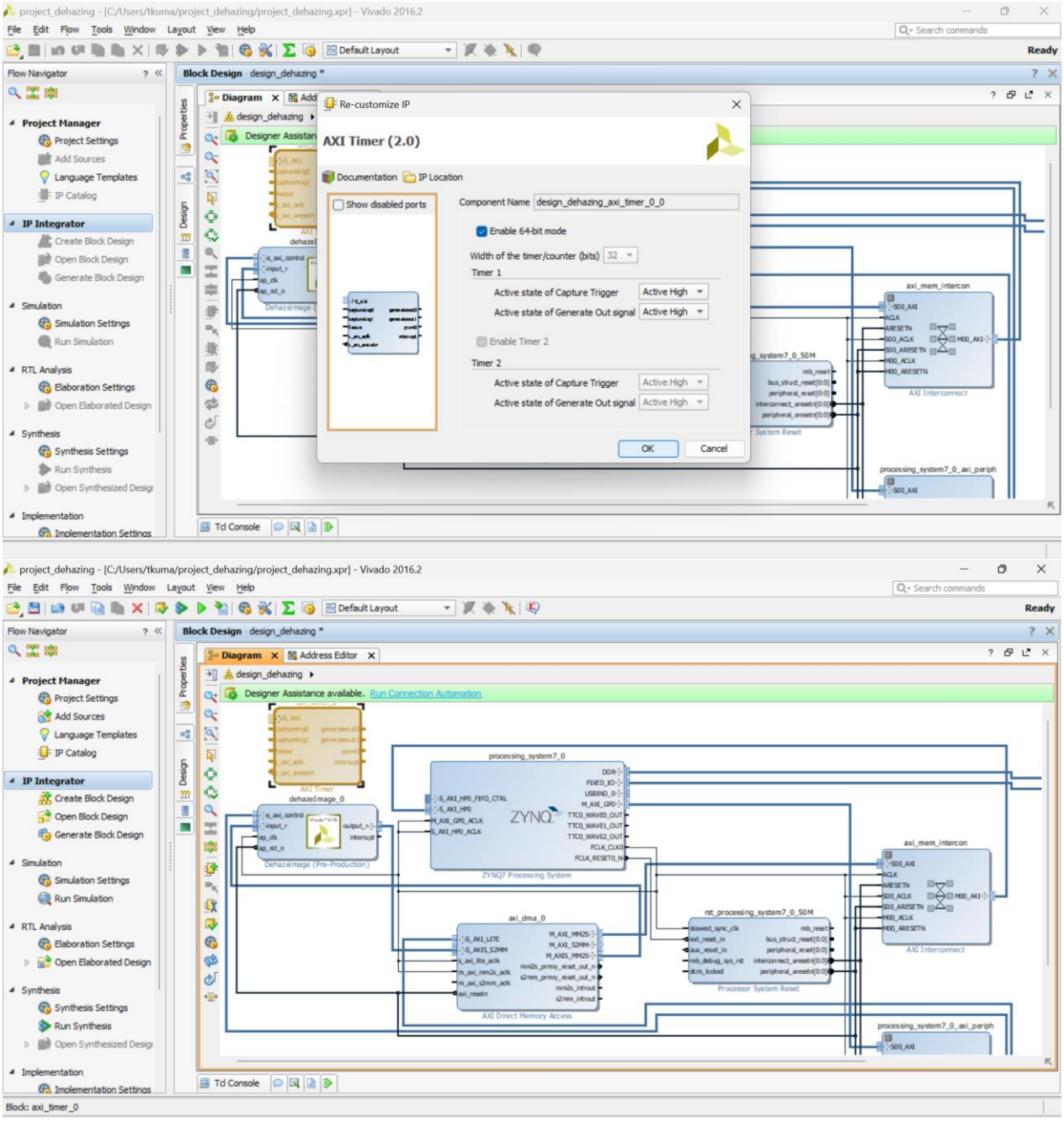

Re-customize IP
AXI Direct Memory Access (7.1)
Documentation
IP Location
Show disabled ports
Component Name
design_dehazing_axi_dma_0_0
Enable Asynchronous Clocks (Auto)
Enable Scatter Gather Engine
Enable Micro DMA
Enable Multi Channel Support
Enable Control / Status Stream
Width of Buffer Length Register (8-23)
Address Width (32-64)
Enable Read Channel
Enable Write Channel
Number of Channels
Memory Map Data Width
Stream Data Width
Max Burst Size
Allow Unaligned Transfers
Use Rxlength In Status Stream
OK
Cancel
project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
Block Design - design_dehazing *
Diagram
Address Editor
Designer Assistance available.
Run Connection Automation
dehazeImage_0
processing_system7_0
axi_dma_0
rst_processing_system7_0_50M
processing_system7_0_axi_periph
Tcl Console


project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Ready
Default Layout
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Run Connection Automation
Automatically make connections in your design by checking the boxes of the interfaces to connect. Select an interface on the left to display its configuration options on the right.
All Automation (2 out of 2 selected)
axi_dma_0
S_AXI_LITE
processing_system7_0
S_AXI_HP0
Description
Connect Slave interface (/processing_system7_0/S_AXI_HP0) to a selected Master address space.
Options
Master:
/axi_dma_0/M_AXI_MM2S
Clock Connection (for unconnected clks) :
Auto
OK
Cancel
Tcl Console
Block Interface: S_AXI_HP0
project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Designer Assistance available. Run Connection Automation
dehazeImage_0
processing_system7_0
ZYNQ7 Processing System
axi_dma_0
AXI Direct Memory Access
rst_processing_system7_0_50M
Processor System Reset
axi_mem_intercon
AXI Interconnect
processing_system7_0_axi_periph
Tcl Console

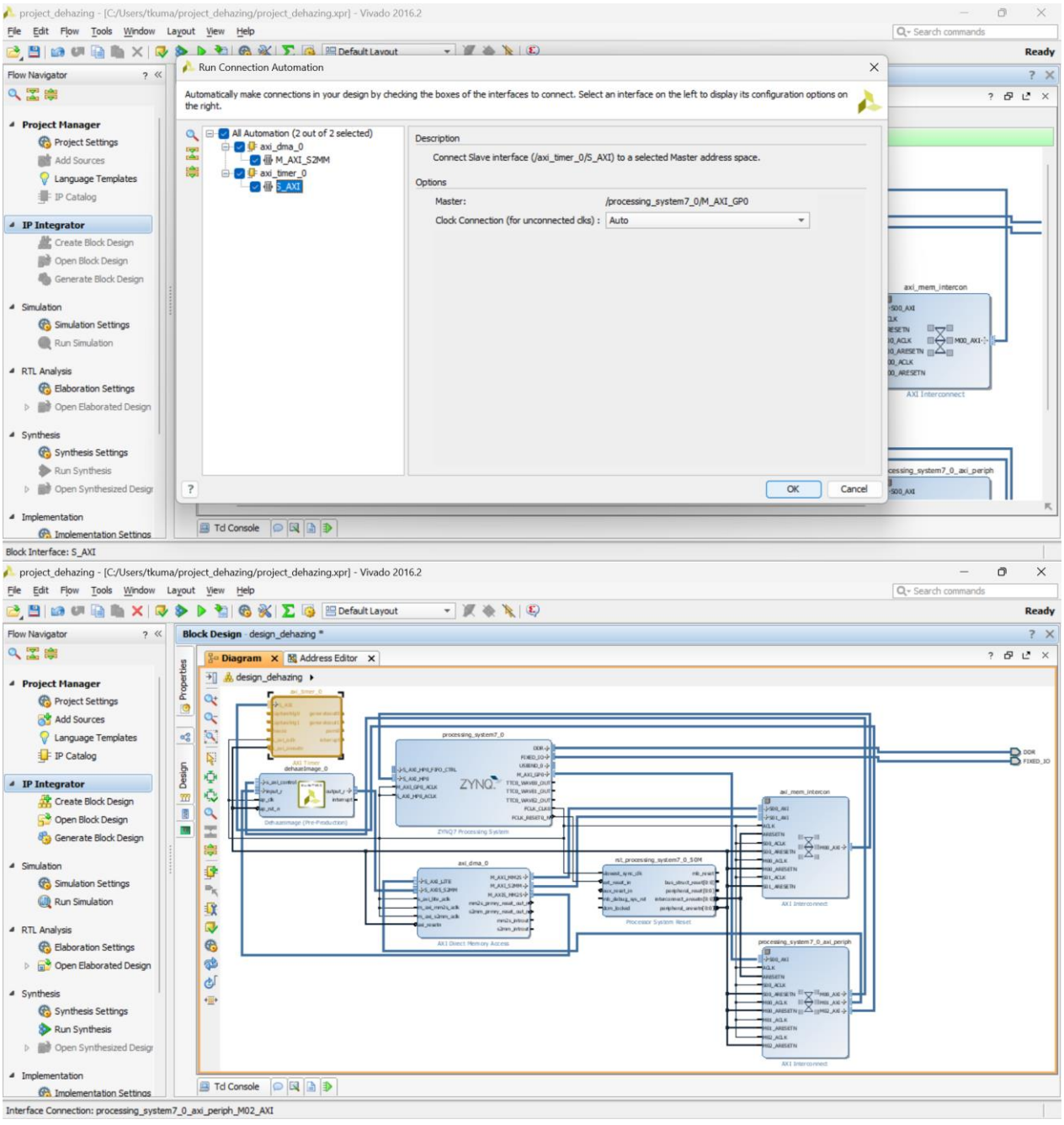

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Designer Assistance available. Run Connection Automation
dehazeImage_0
processing_system7_0
ZYNQ7 Processing System
axi_dma_0
AXI Direct Memory Access
axi_mem_intercon
AXI Interconnect
rst_processing_system7_0_50M
processing_system7_0_axi_periph
Connect from 'input_r' interface
to 'M_AXIS_MM2S' interface
Tcl Console
Press any key to cancel connection mode

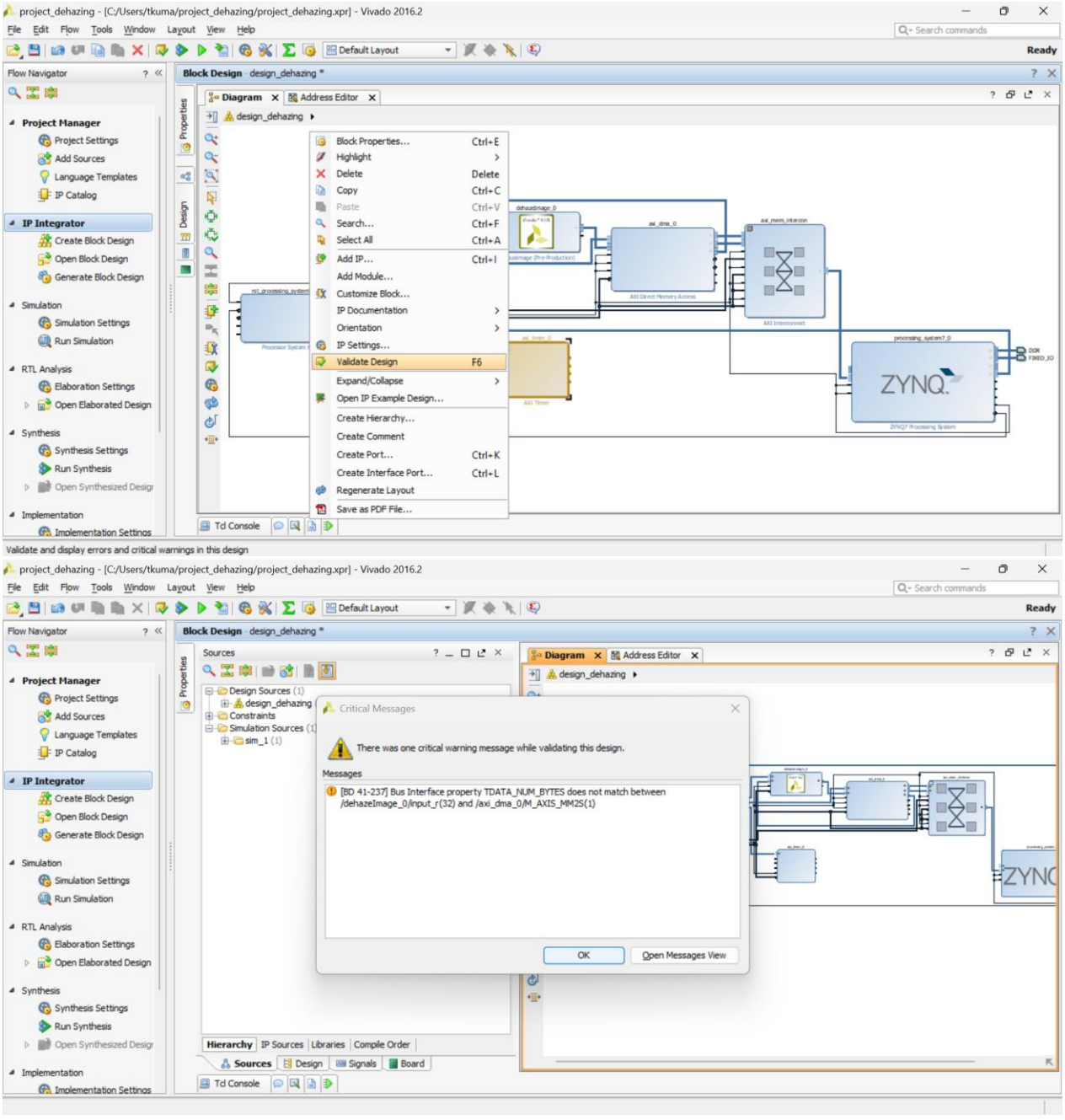

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Designer Assistance available. Run Connection Automation
dehazeImage_0
processing_system7_0
ZYNQ7 Processing System
axi_dma_0
AXI Direct Memory Access
axi_mem_intercon
AXI Interconnect
rst_processing_system7_0_50M
Processor System Reset
processing_system7_0_axi_periph
Connect from 'output_r' interface
to 'S_AXIS_S2MM' interface
Tcl Console
Press any key to cancel connection mode


project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Designer Assistance available. Run Connection Automation
dehazeImage_0
Search: timer (3 matches)
AXI Timebase Watchdog Timer
AXI Timer
Fixed Interval Timer
ENTER to select, ESC to cancel, Ctrl+Q for IP details
processing_system7_0
ZYNQ
ZYNQ7 Processing System
axi_mem_intercon
AXI Interconnect
axi_dma_0
AXI Direct Memory Access
rst_processing_system7_0_50M
Processor System Reset
processing_system7_0_axi_periph
Tcl Console
System Net: rst_processing_system7_0_50M_peripheral_aresetn
Block Properties... Ctrl+E
Highlight
Delete Delete
Copy Ctrl+C
Paste Ctrl+V
Search... Ctrl+F
Select All Ctrl+A
Add IP... Ctrl+I
Add Module...
Run Connection Automation...
Customize Block...
IP Documentation
Orientation
IP Settings...
Validate Design F6
Expand/Collapse
Open IP Example Design...
Create Hierarchy...
Create Comment
Create Port... Ctrl+K
Create Interface Port... Ctrl+L
Regenerate Layout
Save as PDF File...
Configure properties of IP

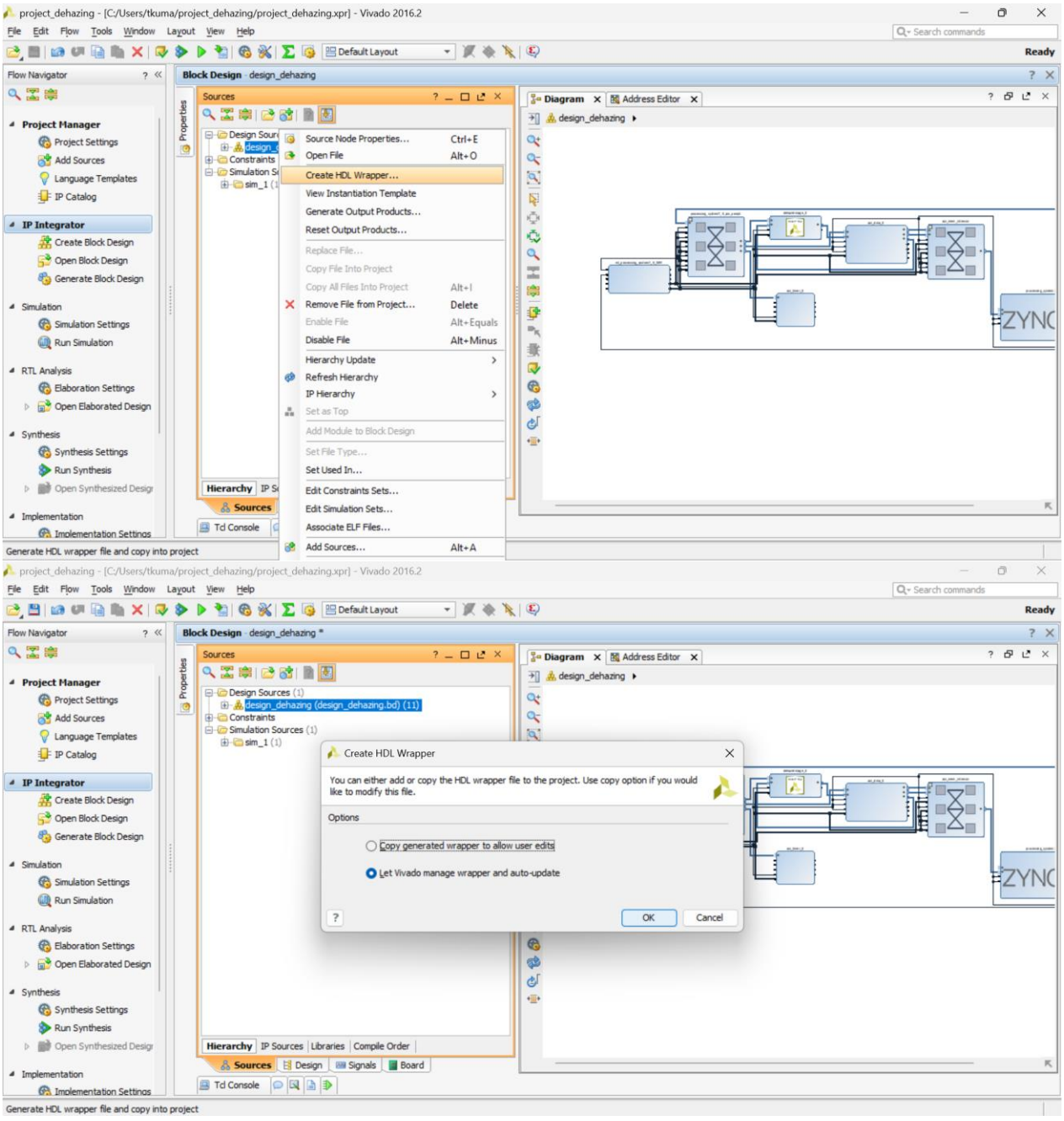

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing *
Diagram
design_dehazing
Re-customize IP
AXI Timer (2.0)
Documentation
IP Location
Show disabled ports
Component Name
design_dehazing_axi_timer_0_0
Enable 64-bit mode
Width of the timer/counter (bits)
32
Timer 1
Active state of Capture Trigger
Active High
Active state of Generate Out signal
Active High
Enable Timer 2
Timer 2
Active state of Capture Trigger
Active High
Active state of Generate Out signal
Active High
OK
Cancel
axi_mem_intercon
AXI Interconnect
processing_system7_0_axi_periph
Tcl Console

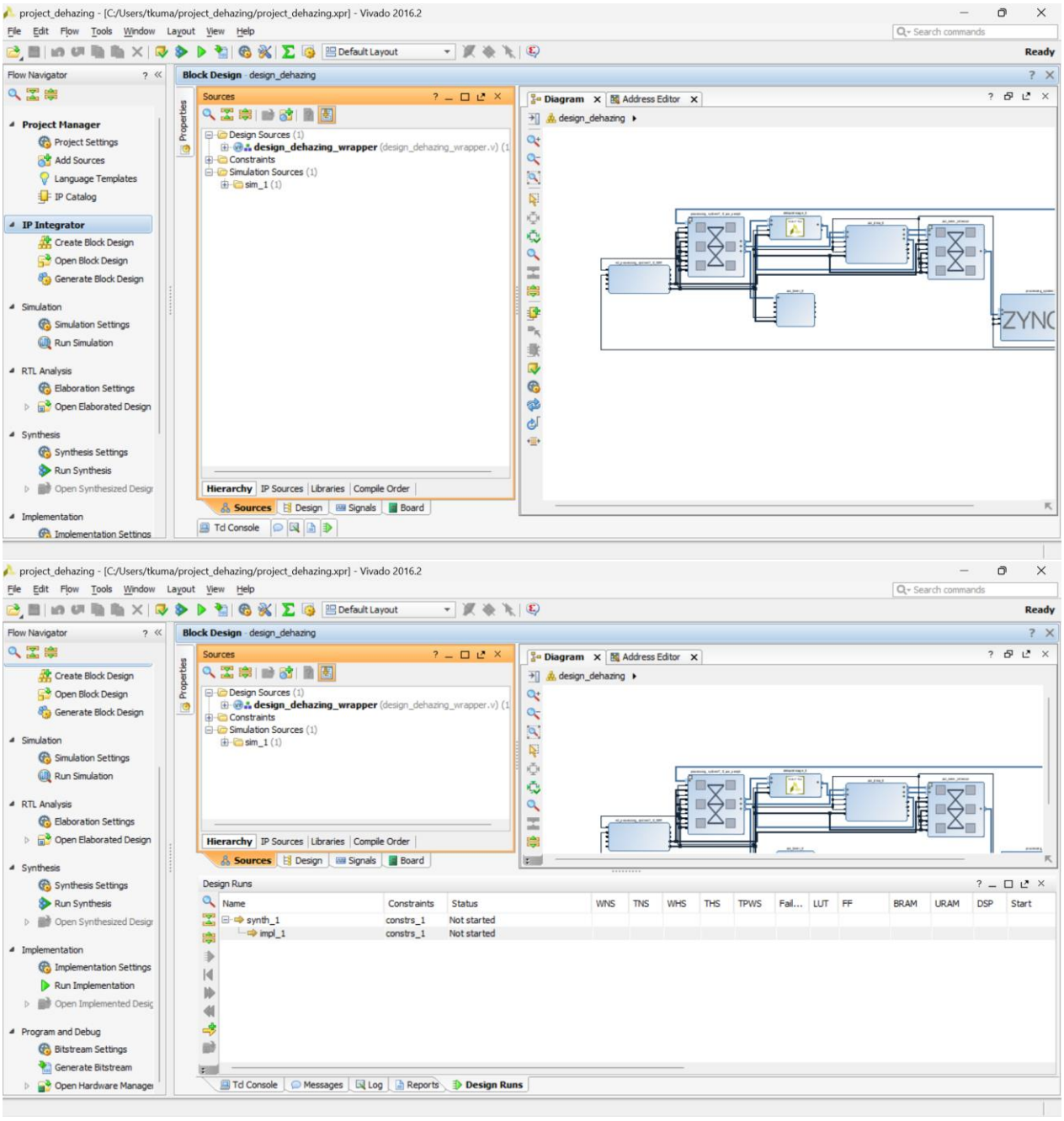

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Designer Assistance available.
Run Connection Automation
AXI Timer
dehazeImage_0
Dehaze Image (Pre-Production)
processing_system7_0
ZYNQ7 Processing System
axi_dma_0
AXI Direct Memory Access
rst_processing_system7_0_50M
Processor System Reset
axi_mem_intercon
AXI Interconnect
processing_system7_0_axi_periph
Tcl Console
Block: axi_timer_0

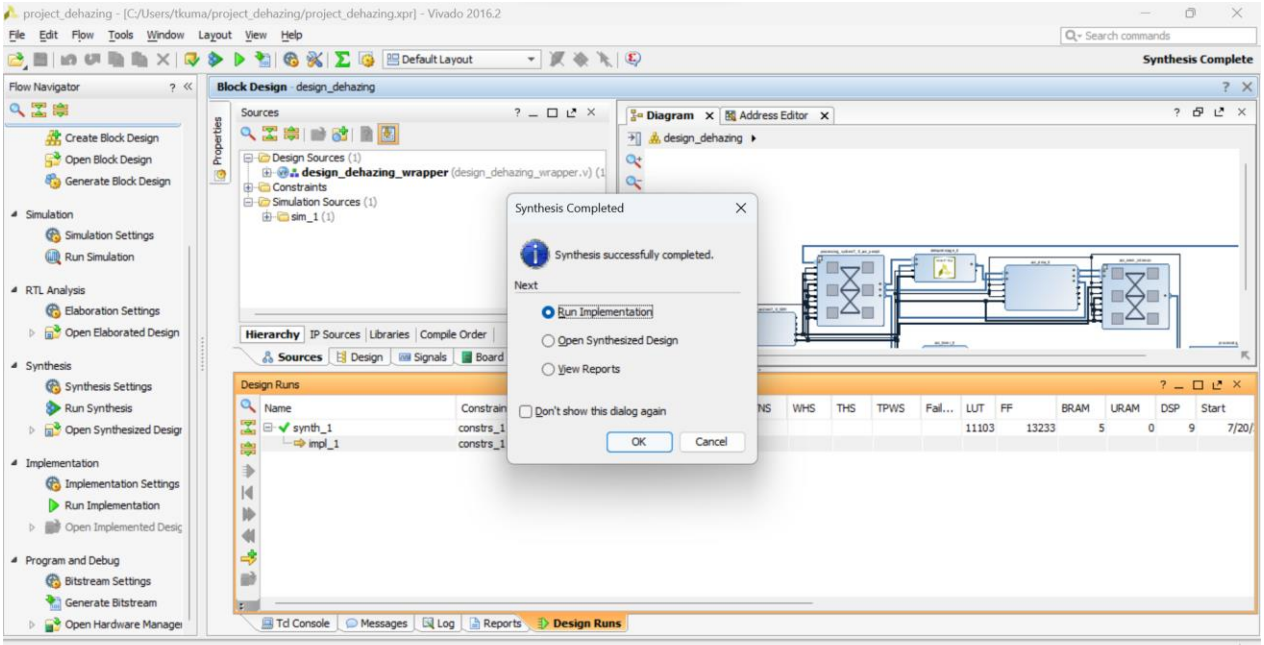

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Ready
Default Layout
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Run Connection Automation
Automatically make connections in your design by checking the boxes of the interfaces to connect. Select an interface on the left to display its configuration options on the right.
All Automation (2 out of 2 selected)
axi_dma_0
M_AXI_S2MM
axi_timer_0
S_AXI
Description
Connect Slave interface (/axi_timer_0/S_AXI) to a selected Master address space.
Options
Master:
/processing_system7_0/M_AXI_GP0
Clock Connection (for unconnected clks) :
Auto
OK
Cancel
Tcl Console
Block Interface: S_AXI
project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
Block Design - design_dehazing *
Diagram
Address Editor
design_dehazing
Properties
Design
processing_system7_0
axi_dma_0
axi_mem_intercon
processing_system7_0_axi_periph
Tcl Console
Interface Connection: processing_system7_0_axi_periph_M02_AXI

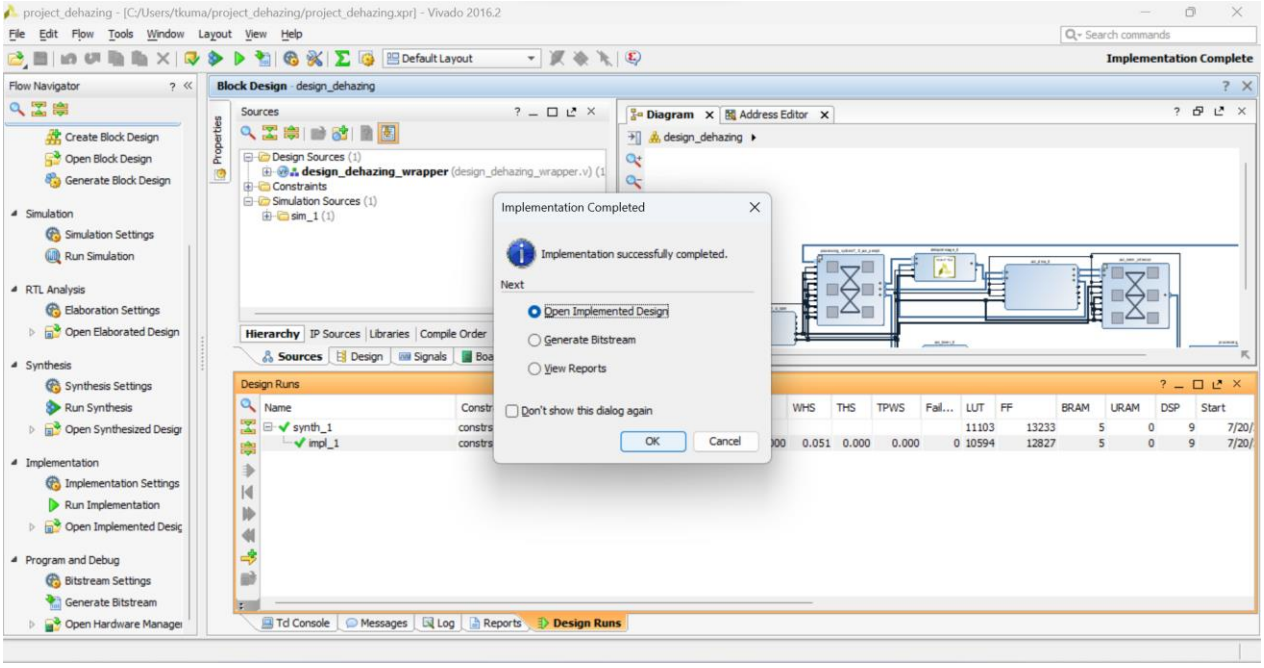


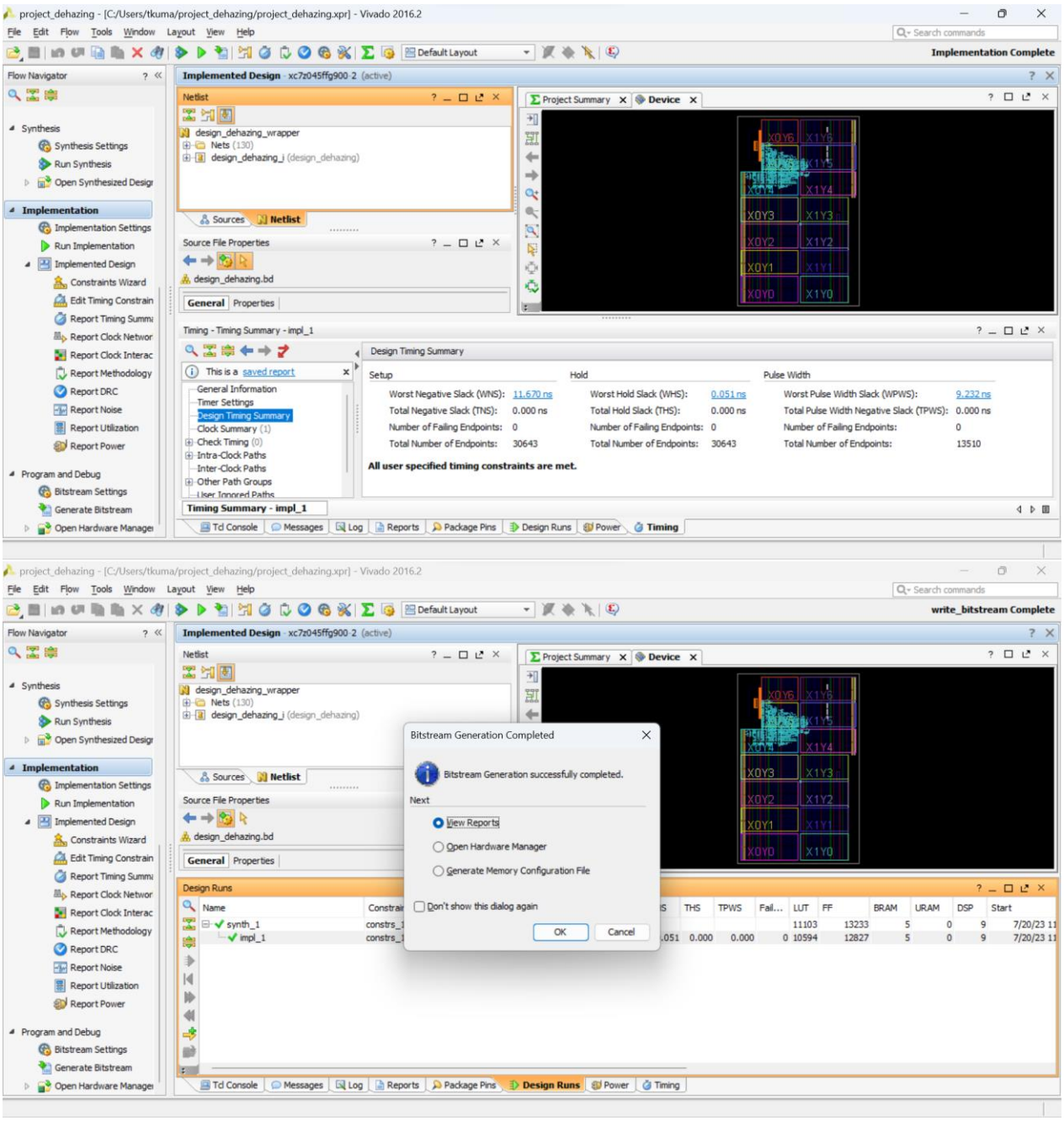

After completion of block diagram, the below is the process of wrapping and generating bitstream.


project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing
Sources
Design Sources
Constraints
Simulation Sources
sim_1 (1)
Source Node Properties... Ctrl+E
Open File Alt+O
Create HDL Wrapper...
View Instantiation Template
Generate Output Products...
Reset Output Products...
Replace File...
Copy File Into Project
Copy All Files Into Project Alt+I
Remove File from Project... Delete
Enable File Alt+Equals
Disable File Alt+Minus
Hierarchy Update
Refresh Hierarchy
IP Hierarchy
Set as Top
Add Module to Block Design
Set File Type...
Set Used In...
Edit Constraints Sets...
Edit Simulation Sets...
Associate ELF Files...
Add Sources... Alt+A
Diagram
Address Editor
design_dehazing
Hierarchy
Sources
Tcl Console
Generate HDL wrapper file and copy into project
project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
Block Design - design_dehazing *
design_dehazing (design_dehazing.bd) (11)
Create HDL Wrapper
You can either add or copy the HDL wrapper file to the project. Use copy option if you would like to modify this file.
Options
Copy generated wrapper to allow user edits
Let Vivado manage wrapper and auto-update
OK
Cancel
Hierarchy IP Sources Libraries Compile Order
Sources Design Signals Board
Generate HDL wrapper file and copy into project

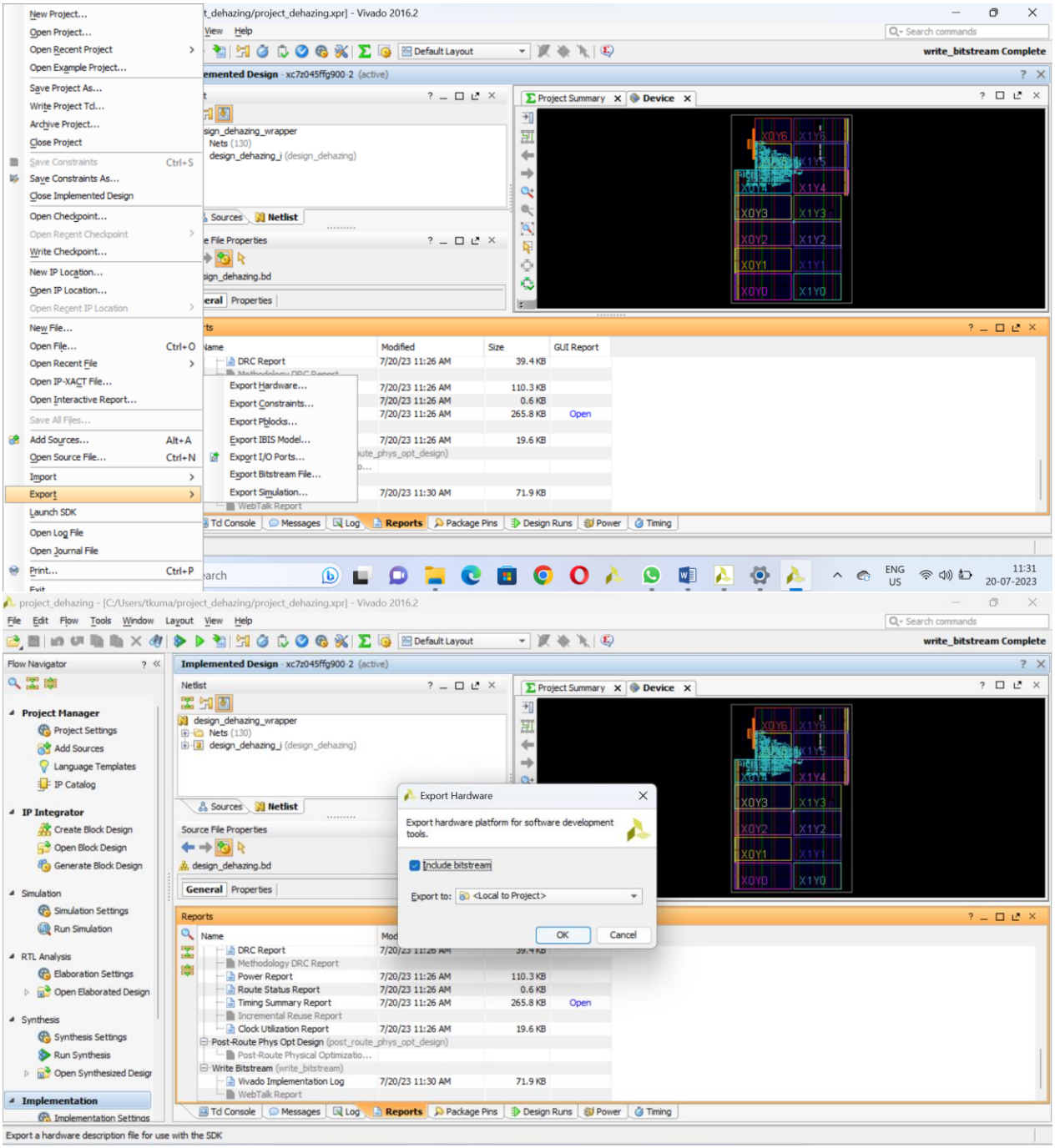

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Search commands
Default Layout
Ready
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Block Design - design_dehazing
Sources
Design Sources (1)
design_dehazing_wrapper (design_dehazing_wrapper.v) (1
Constraints
Simulation Sources (1)
sim_1 (1)
Hierarchy IP Sources Libraries Compile Order
Sources Design Signals Board
Tcl Console
Diagram
Address Editor
design_dehazing
ZYNQ
Run Implementation
Open Implemented Design
Program and Debug
Bitstream Settings
Generate Bitstream
Open Hardware Manager
Design Runs
Name Constraints Status WNS TNS WHS THS TPWS Fail... LUT FF BRAM URAM DSP Start
synth_1 constrs_1 Not started
impl_1 constrs_1 Not started
Tcl Console Messages Log Reports Design Runs


project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
Synthesis Complete
Synthesis Completed
Synthesis successfully completed.
Next
Run Implementation
Open Synthesized Design
View Reports
Don't show this dialog again
OK
Cancel

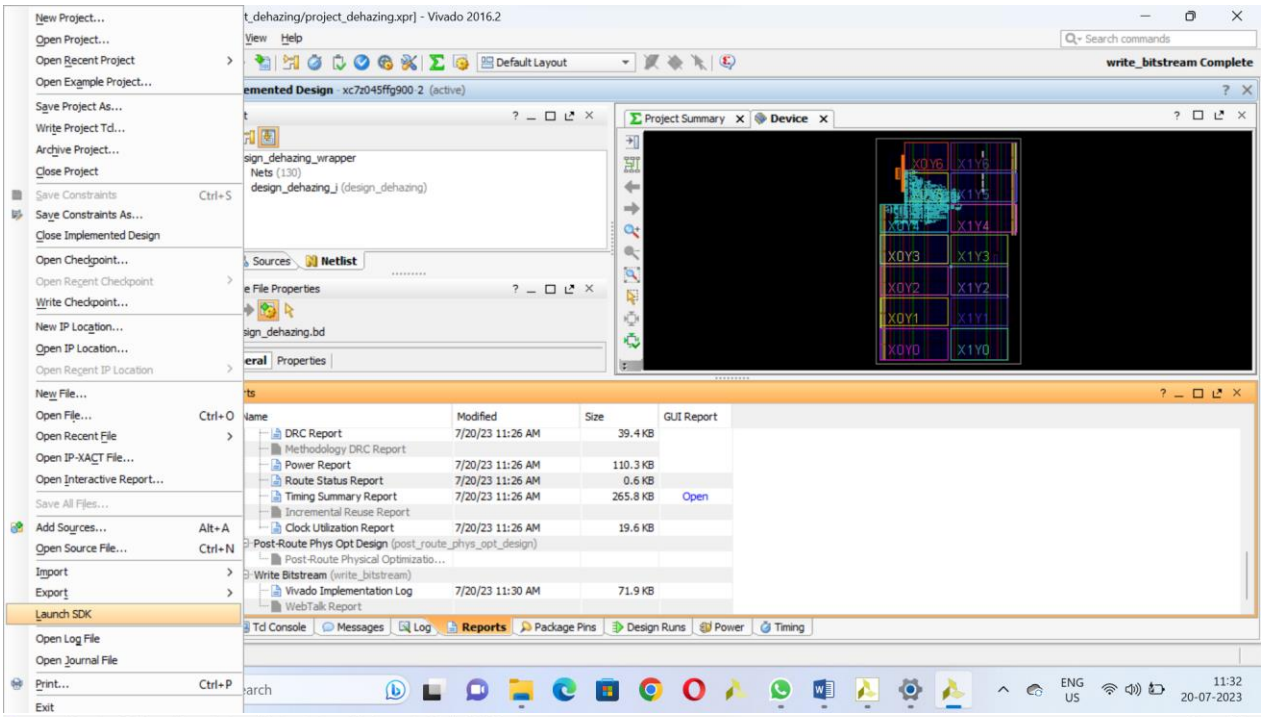

project_dehazing - [C:/Users/tkuma/project_dehazing/project_dehazing.xpr] - Vivado 2016.2
Implementation Complete
Implementation Completed
Implementation successfully completed.
Next
Open Implemented Design
Generate Bitstream
View Reports
Don't show this dialog again
OK
Cancel

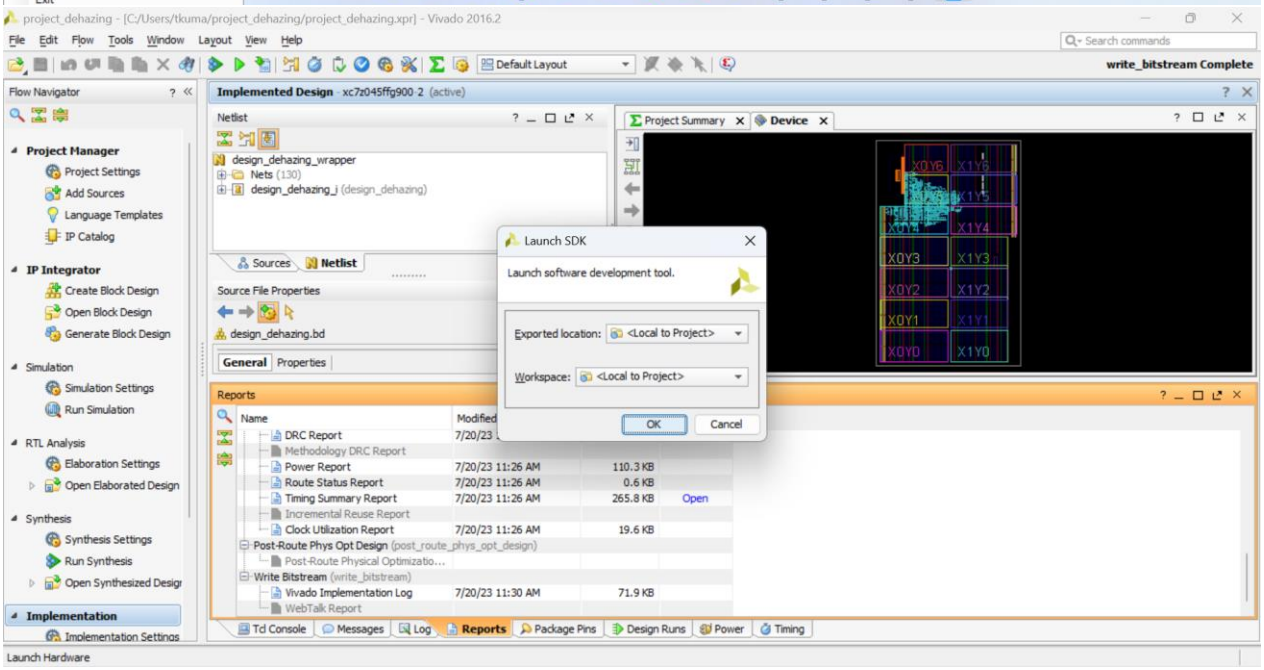


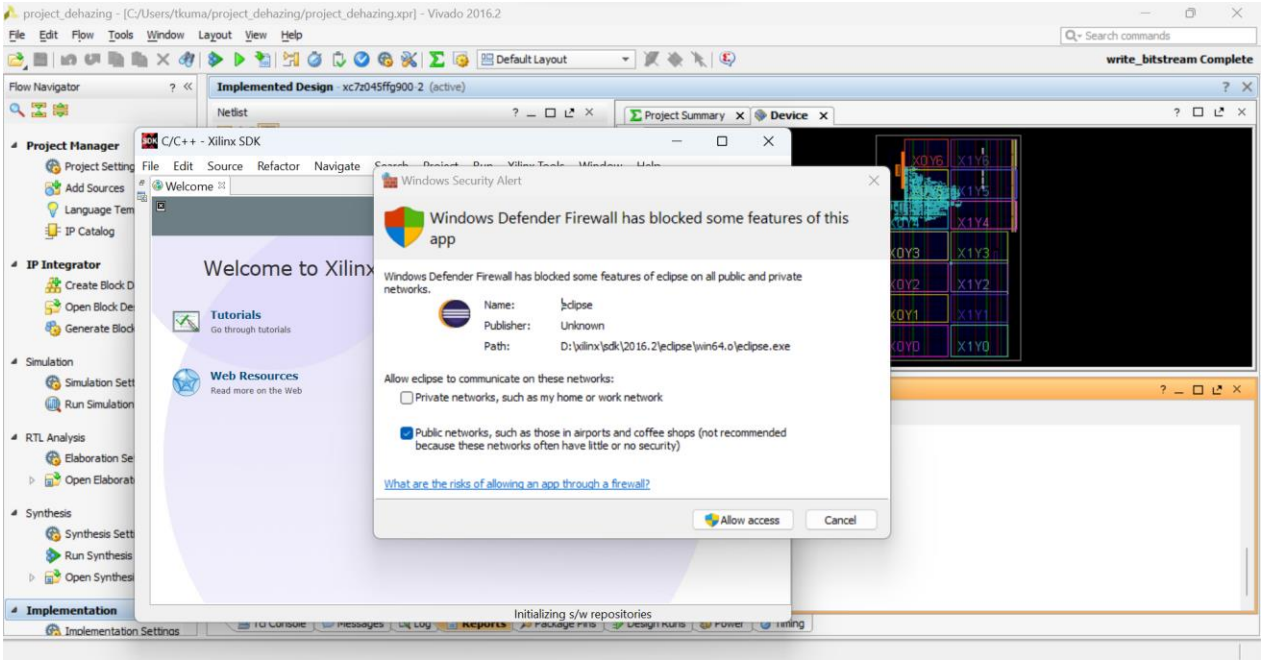

After generating the bitstream, export it to the hardware including the bitstream to dump it to the FPGA.



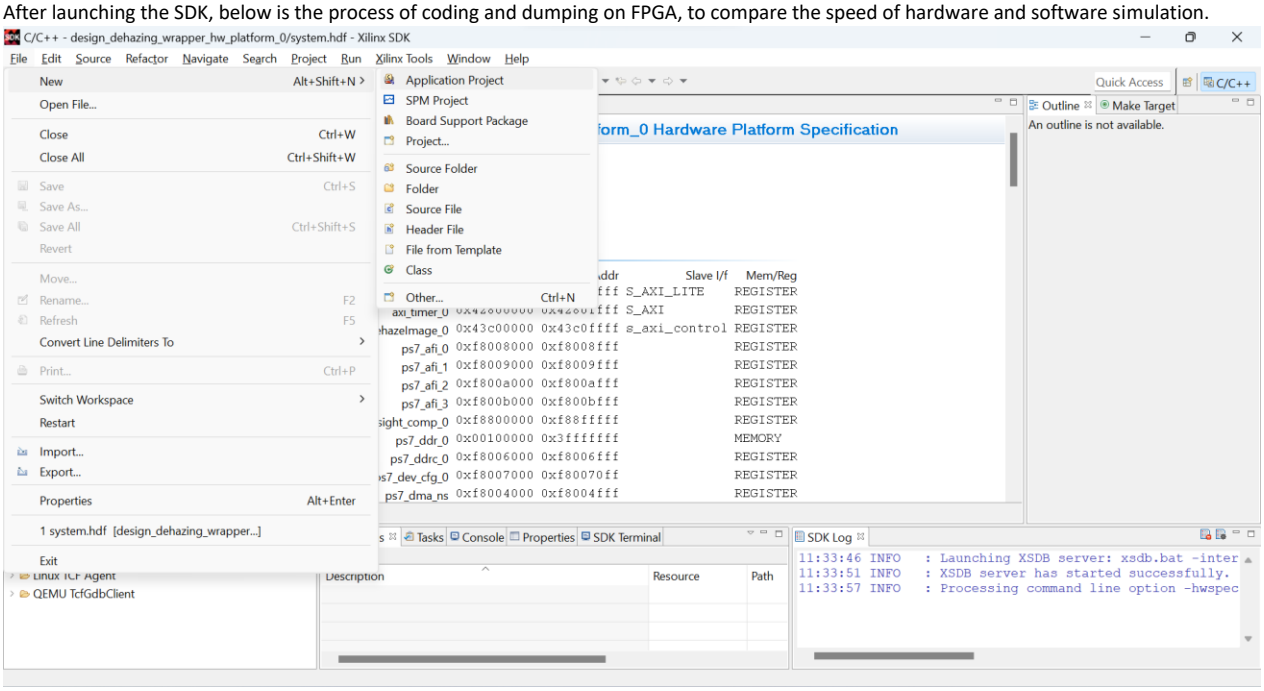

After exporting hardware, launch SDK in same location for dumping on FPGA

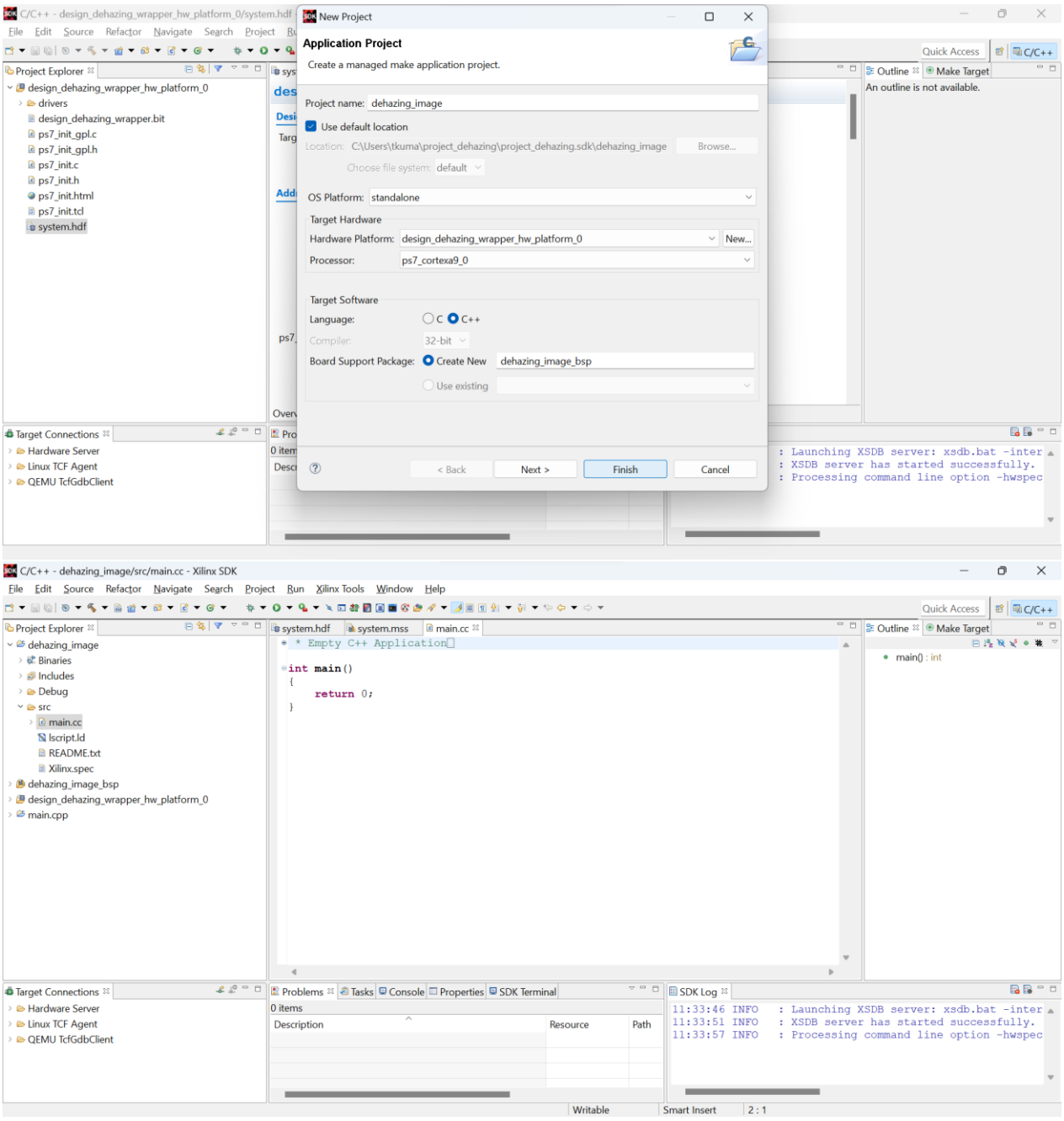

New Project...
Open Project...
Open Recent Project
Open Example Project...
Save Project As...
Write Project Tcl...
Archive Project...
Close Project
Save Constraints Ctrl+S
Save Constraints As...
Close Implemented Design
Open Checkpoint...
Open Recent Checkpoint
Write Checkpoint...
New IP Location...
Open IP Location...
Open Recent IP Location
New File...
Open File... Ctrl+O
Open Recent File
Open IP-XACT File...
Open Interactive Report...
Save All Files...
Add Sources... Alt+A
Open Source File... Ctrl+N
Import
Export
Launch SDK
Open Log File
Open Journal File
Print... Ctrl+P
Exit
write_bitstream Complete
Project Summary
Device
Name
Modified
Size
GUI Report
DRC Report
Methodology DRC Report
Power Report
Route Status Report
Timing Summary Report
Incremental Reuse Report
Clock Utilization Report
Post-Route Phys Opt Design
Write Bitstream
Vivado Implementation Log
WebTalk Report
Tcl Console
Messages
Log
Reports
Package Pins
Design Runs
Power
Timing


File Edit Flow Tools Window Layout View Help
write_bitstream Complete
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Implementation
Implementation Settings
Launch SDK
Launch software development tool.
Exported location: <Local to Project>
Workspace: <Local to Project>
OK
Cancel
Launch Hardware




After launching the SDK, below is the process of coding and dumping on FPGA, to compare the speed of hardware and software simulation.







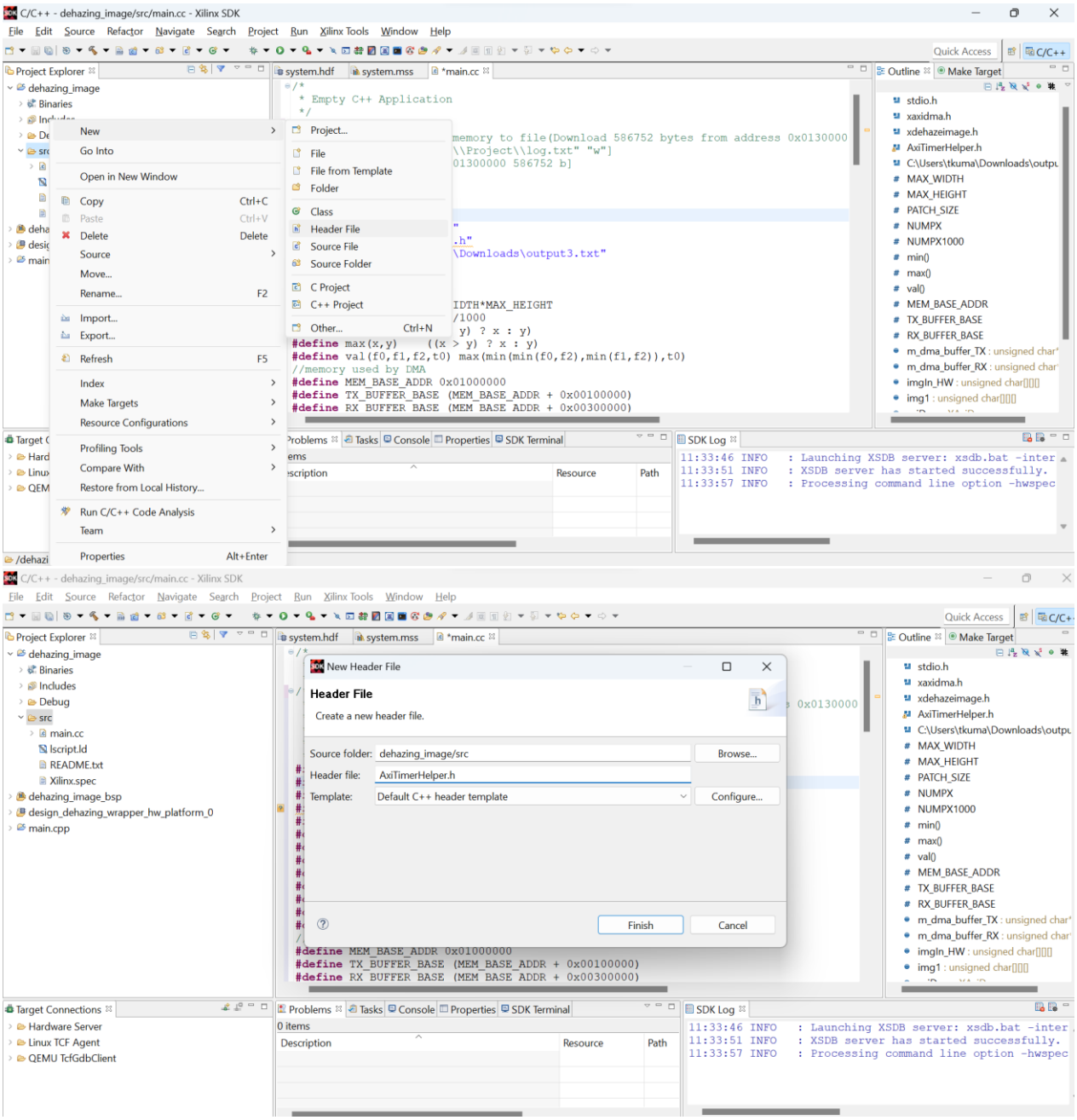


code for main.cc:

```
/*
 * Empty C++ Application
 */
/*
 * Commands to download memory to file(Download 586752 bytes from address 0x01300000 to file logresp.txt
 * set logfile [open "E:\\Project\\log.txt" "w"]
 * puts $logfile [mrd 0x01300000 586752 b]
 * close $logfile
 */
#include <stdio.h>
#include "xaxidma.h"
```

```
#include "xdehazeimage.h"
#include "AxiTimerHelper.h"
#include "C:\Users\tkuma\Downloads\output3.txt"
#define MAX_WIDTH 512
#define MAX_HEIGHT 384
#define PATCH_SIZE 15
#define NUMPX           MAX_WIDTH*MAX_HEIGHT
#define NUMPX1000  NUMPX/1000
#define min(x,y)   ((x < y) ? x : y)
#define max(x,y)        ((x > y) ? x : y)
#define val(f0,f1,f2,t0) max(min(min(f0,f2),min(f1,f2)),t0)
//memory used by DMA
#define MEM_BASE_ADDR 0x01000000
#define TX_BUFFER_BASE (MEM_BASE_ADDR + 0x00100000)
#define RX_BUFFER_BASE (MEM_BASE_ADDR + 0x00300000)
//get a pointer to the TX and RX dma buffer(configure DMA)
//the pointers are 8-bit memory but their addresses are 32 bit(u32)
unsigned char* m_dma_buffer_TX = (unsigned char*)TX_BUFFER_BASE;
unsigned char* m_dma_buffer_RX = (unsigned char*)RX_BUFFER_BASE;
unsigned char imgIn_HW[MAX_WIDTH][MAX_HEIGHT][3];
unsigned char img1[384][512][3];
XAxiDma axiDma;
int initDMA(){
        XAxiDma_Config *Cfgptr;
        Cfgptr = XAxiDma_LookupConfig(XPAR_AXI_DMA_0_DEVICE_ID);
        XAxiDma_CfgInitialize(&axiDma, Cfgptr);
        //Disable interrupts
        XAxiDma_IntrDisable(&axiDma,XAXIDMA_IRQ_ALL_MASK,XAXIDMA_DEVICE_TO_DMA);
        XAxiDma_IntrDisable(&axiDma,XAXIDMA_IRQ_ALL_MASK,XAXIDMA_DMA_TO_DEVICE);
        return XST_SUCCESS;
}
XDehazeimage dehazeImage;
int initdehaze(){
        int status;
        XDehazeimage_Config *dehazeImage_cfg;
        dehazeImage_cfg=XDehazeimage_LookupConfig(XPAR_DEHAZEIMAGE_0_DEVICE_ID);
        if(!dehazeImage_cfg)
        {
                printf("Error loading config for dehazeImage_cfg\n");
        }
        status = XDehazeimage_CfgInitialize(&dehazeImage, dehazeImage_cfg);
        if(status!=XST_SUCCESS)
        {
                printf("Error initializing for dehazeImage\n");
        }
        return status;
}
int main()
{
        printf("1.Debugging\n");
        initDMA();
        printf("2\n");
        initdehaze();
        printf("Doing dehaze on HW\n");
        //reading the image
        for(int idx=0;idx<MAX_HEIGHT;idx++)
        {
                for(int idx1=0;idx1<MAX_WIDTH;idx1++)
           {
                        for(int idx2=0;idx2<3;idx2++)
                        {
```

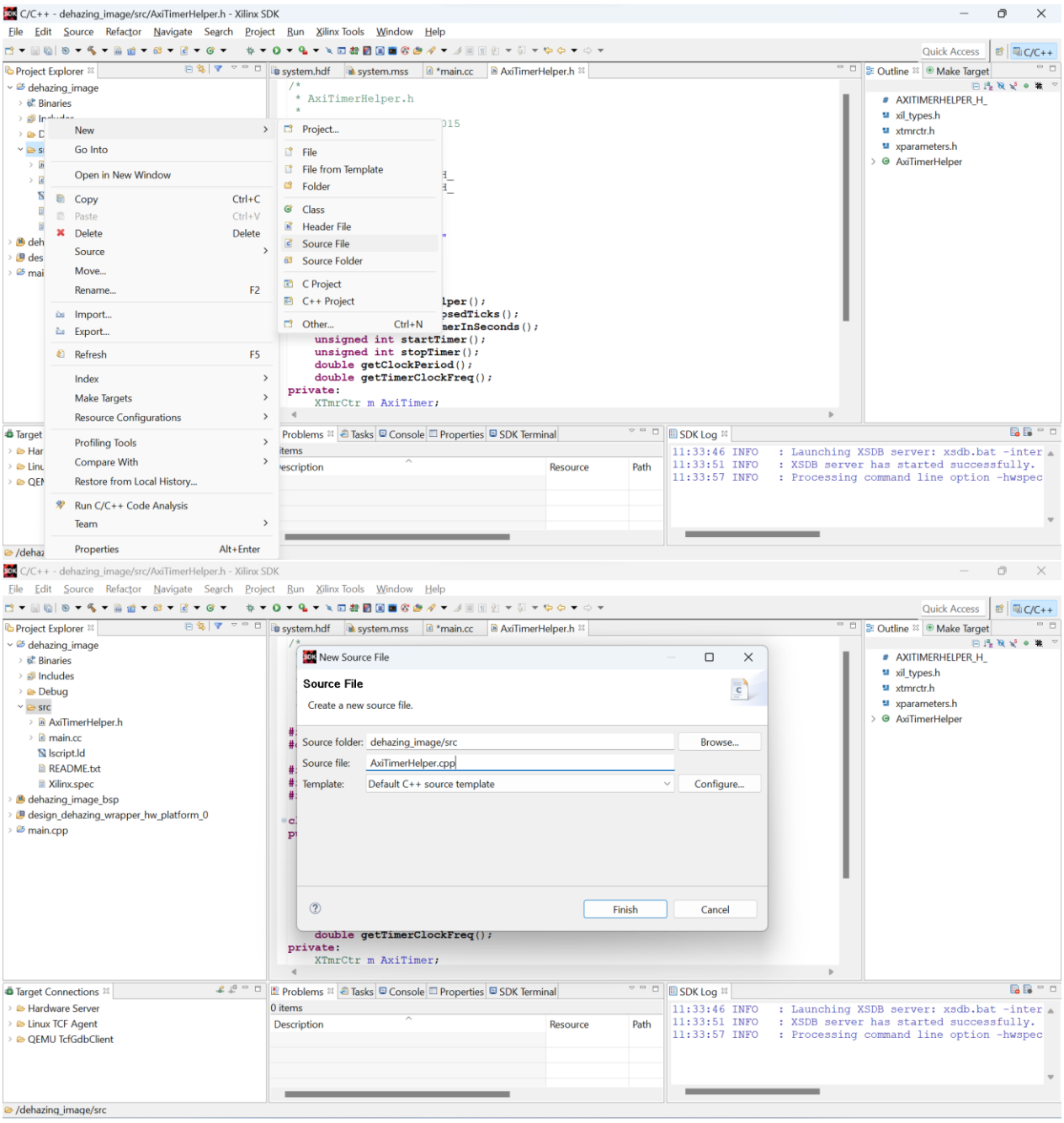
```
                                        imgIn_HW[idx][idx1][idx2]=img[idx][idx1][idx2];
                                }
                                }
        }
        printf("3\n");
        //imgIn_HW=cv::imread("C:\\Users\\tkuma\\Downloads\\haze.png");
        AxiTimerHelper axiTimer;
        printf("Starting......HW\n");
        //ask for dehazing
        XDehazeimage_Set_t(&dehazeImage,0.1);
        XDehazeimage_Set_w(&dehazeImage,0.75);
        printf("4\n");
        XDehazeimage_Start(&dehazeImage);
        printf("5\n");
        //do the dma transfer to push and get our image
        axiTimer.startTimer();
        printf("6\n");
        Xil_DCacheFlushRange((u32)imgIn_HW,MAX_HEIGHT*MAX_WIDTH*3*sizeof(unsigned char));
        printf("7\n");
        Xil_DCacheFlushRange((u32)m_dma_buffer_RX,MAX_HEIGHT*MAX_WIDTH*3*sizeof(unsigned char));
        printf("8\n");
        XAxiDma_SimpleTransfer(&axiDma,(u32)imgIn_HW,MAX_HEIGHT*MAX_WIDTH*3*sizeof(unsigned
char),XAXIDMA_DMA_TO_DEVICE);
        printf("9\n");
        XAxiDma_SimpleTransfer(&axiDma,(u32)m_dma_buffer_RX,MAX_HEIGHT*MAX_WIDTH*3*sizeof(unsigned
char),XAXIDMA_DEVICE_TO_DMA);
        printf("10\n");
        Xil_DCacheInvalidateRange((u32)m_dma_buffer_RX,MAX_HEIGHT*MAX_WIDTH*3*sizeof(unsigned char));
        printf("11\n");
        axiTimer.stopTimer();
        printf("12\n");
        double HW_elapsed = axiTimer.getElapsedTimerInSeconds();
        printf("HW execution time: %f sec\n",HW_elapsed);
        return 0;
}
```




Code for AxiTimerHelper.h:

```
/*
 * AxiTimerHelper.h
 *
 *  Created on: 06/07/2015
 *      Author: laraujo
 */

#ifndef AXITIMERHELPER_H_
#define AXITIMERHELPER_H_

#include "xil_types.h"
#include "xtmrctr.h"
#include "xparameters.h"
```

```
class AxiTimerHelper {
public:
        AxiTimerHelper();
        virtual ~AxiTimerHelper();
        unsigned int getElapsedTicks();
        double getElapsedTimerInSeconds();
        unsigned int startTimer();
        unsigned int stopTimer();
        double getClockPeriod();
        double getTimerClockFreq();
private:
        XTmrCtr m_AxiTimer;
        unsigned int m_tickCounter1;
        unsigned int m_tickCounter2;
        double m_clockPeriodSeconds;
        double m_timerClockFreq;
};

#endif /* AXITIMERHELPER_H_ */
```

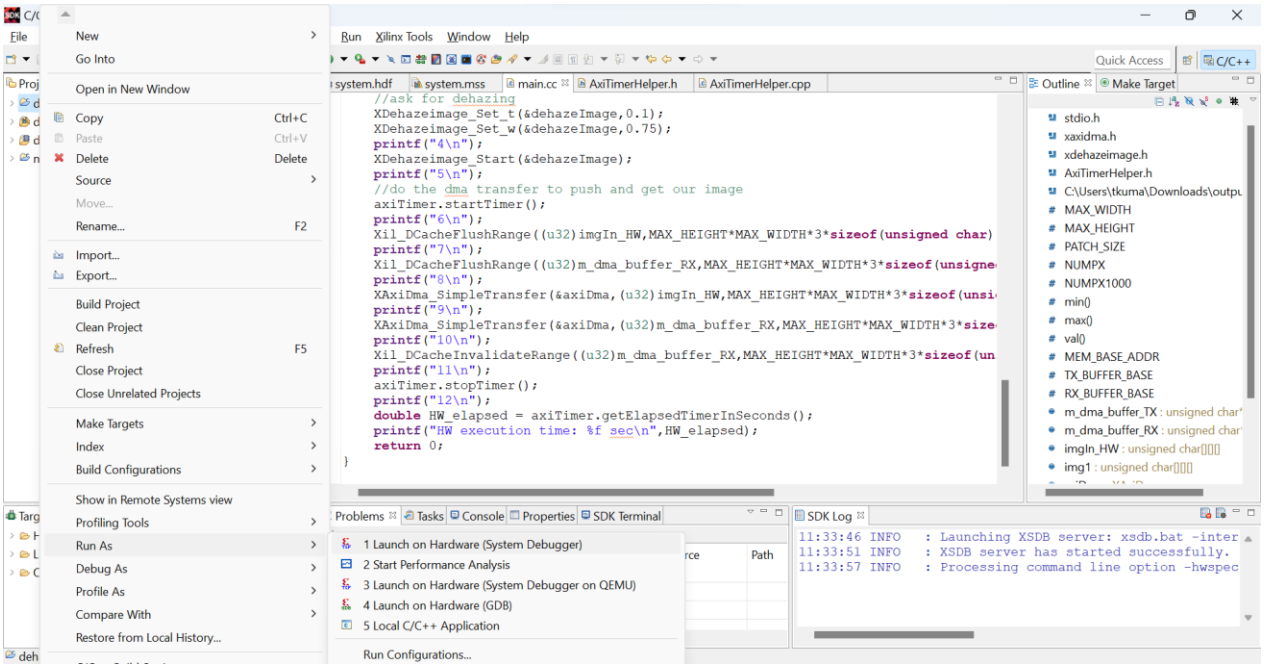



Code for AxiTimerHelper.cpp:

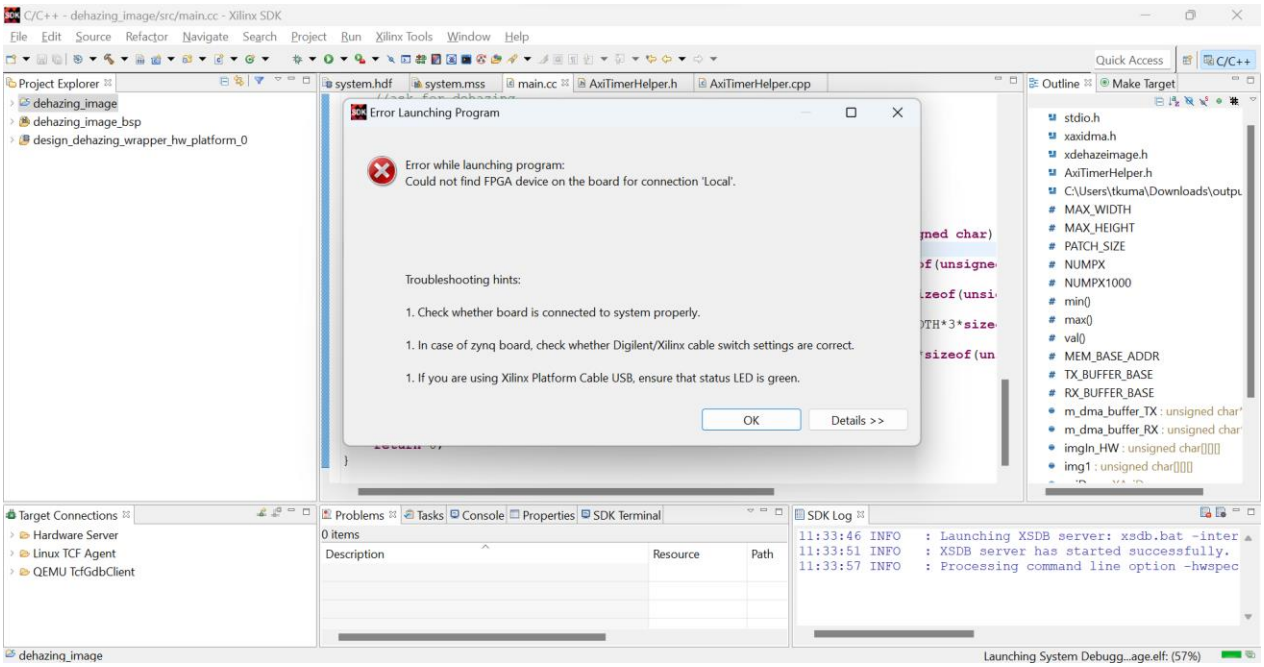
```
/*
 * AxiTimerHelper.cpp
 *
 *  Created on: 06/07/2015
 *      Author: laraujo
 */

#include "AxiTimerHelper.h"

AxiTimerHelper::AxiTimerHelper() {
		// Initialize the timer hardware...
		 XTmrCtr_Initialize(&m_AxiTimer, XPAR_TMRCTR_0_DEVICE_ID);
```

```
		//XTmrCtr_SetOptions(&m_AxiTimer, 0, XTC_AUTO_RELOAD_OPTION);

		// Get the clock period in seconds
		m_timerClockFreq = (double) XPAR_AXI_TIMER_0_CLOCK_FREQ_HZ;
		m_clockPeriodSeconds = (double)1/m_timerClockFreq;

}

AxiTimerHelper::~AxiTimerHelper() {
		// TODO Auto-generated destructor stub
}

unsigned int AxiTimerHelper::getElapsedTicks() {
		return m_tickCounter2 - m_tickCounter1;
}

unsigned int AxiTimerHelper::startTimer() {
		// Start timer 0 (There are two, but depends how you configured in vivado)
		XTmrCtr_Reset(&m_AxiTimer,0);
		m_tickCounter1 =  XTmrCtr_GetValue(&m_AxiTimer, 0);
		XTmrCtr_Start(&m_AxiTimer, 0);
		return m_tickCounter1;
}

unsigned int AxiTimerHelper::stopTimer() {
		XTmrCtr_Stop(&m_AxiTimer, 0);
		m_tickCounter2 =  XTmrCtr_GetValue(&m_AxiTimer, 0);
		return m_tickCounter2 - m_tickCounter1;
}

double AxiTimerHelper::getElapsedTimerInSeconds() {
		double elapsedTimeInSeconds = (double)(m_tickCounter2 - m_tickCounter1) * m_clockPeriodSeconds;
		return elapsedTimeInSeconds;
}

double AxiTimerHelper::getClockPeriod() {
		return m_clockPeriodSeconds;
}

double AxiTimerHelper::getTimerClockFreq() {
		return m_timerClockFreq;
}
```

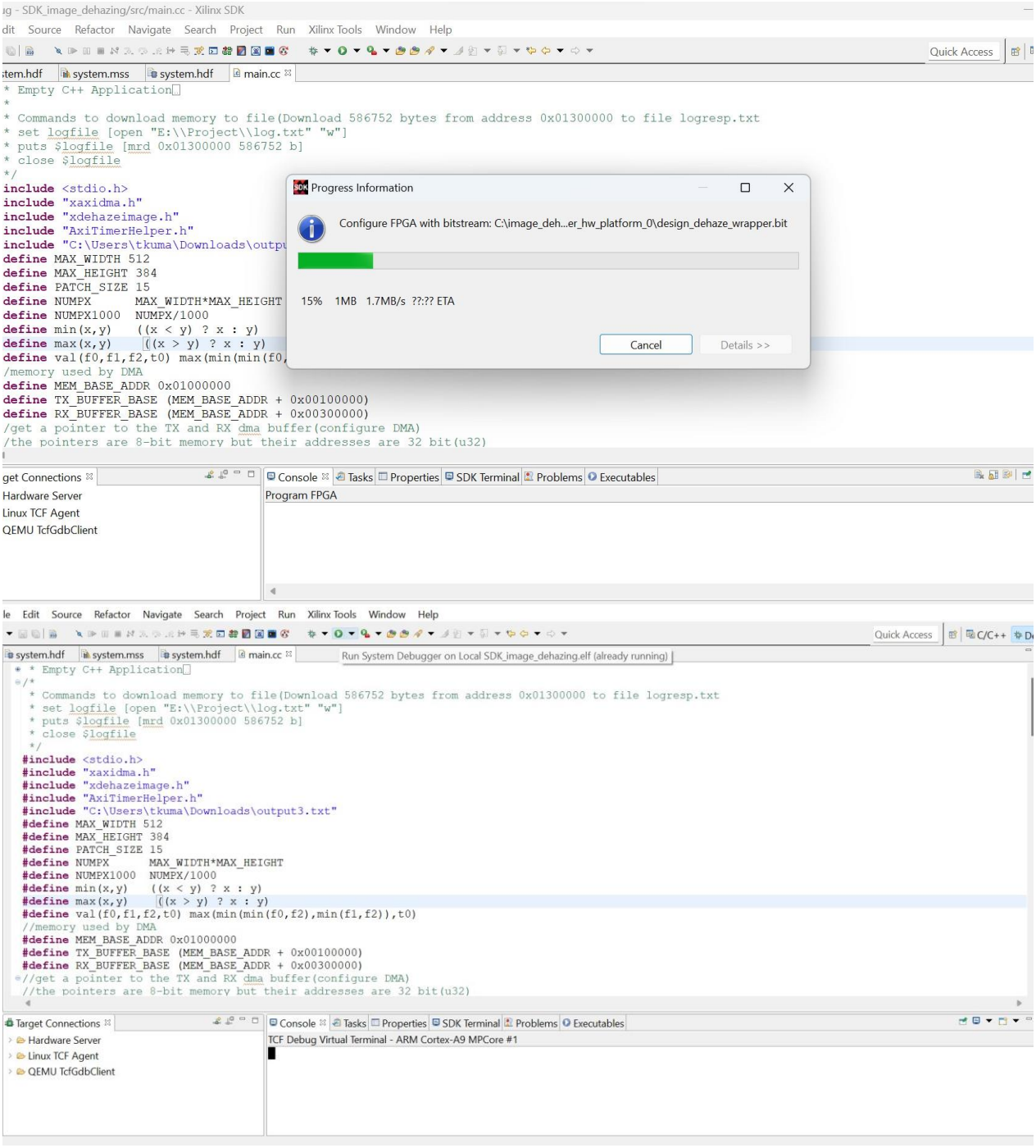
Click on Launch on Hardware to check the results




Below the error shows that FPGA is not connected

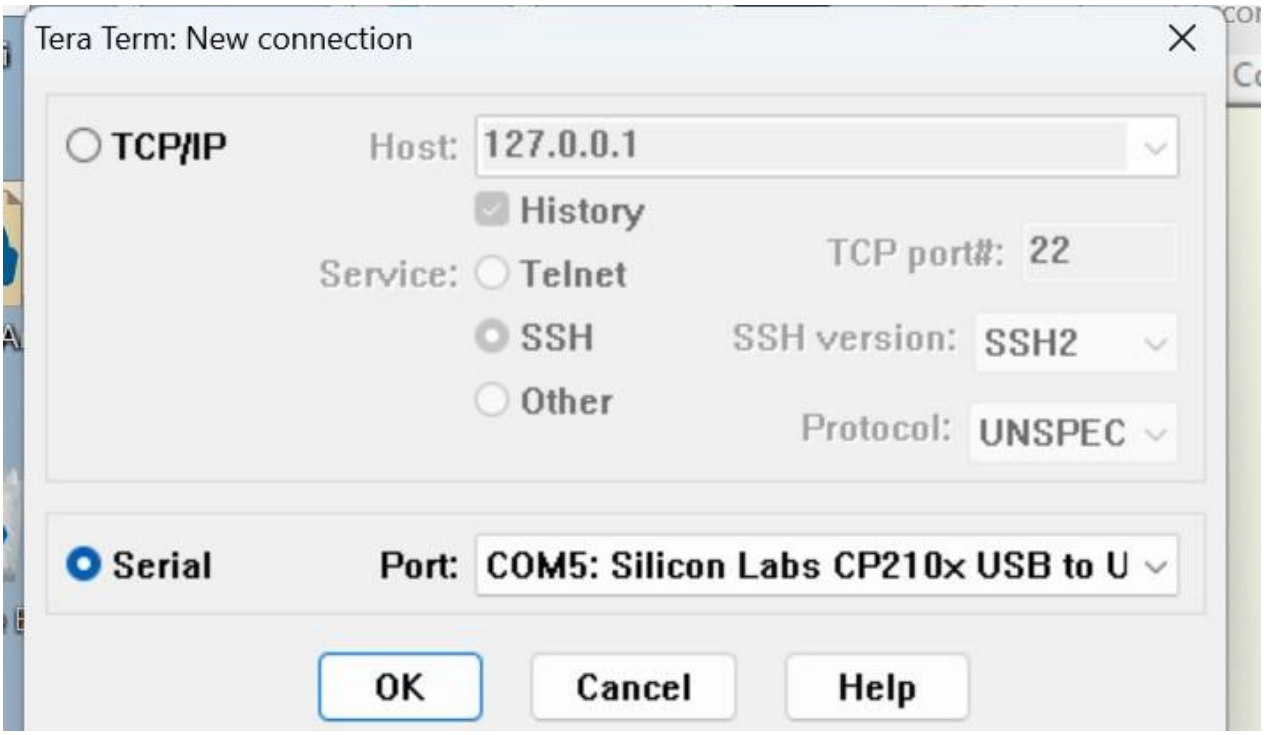


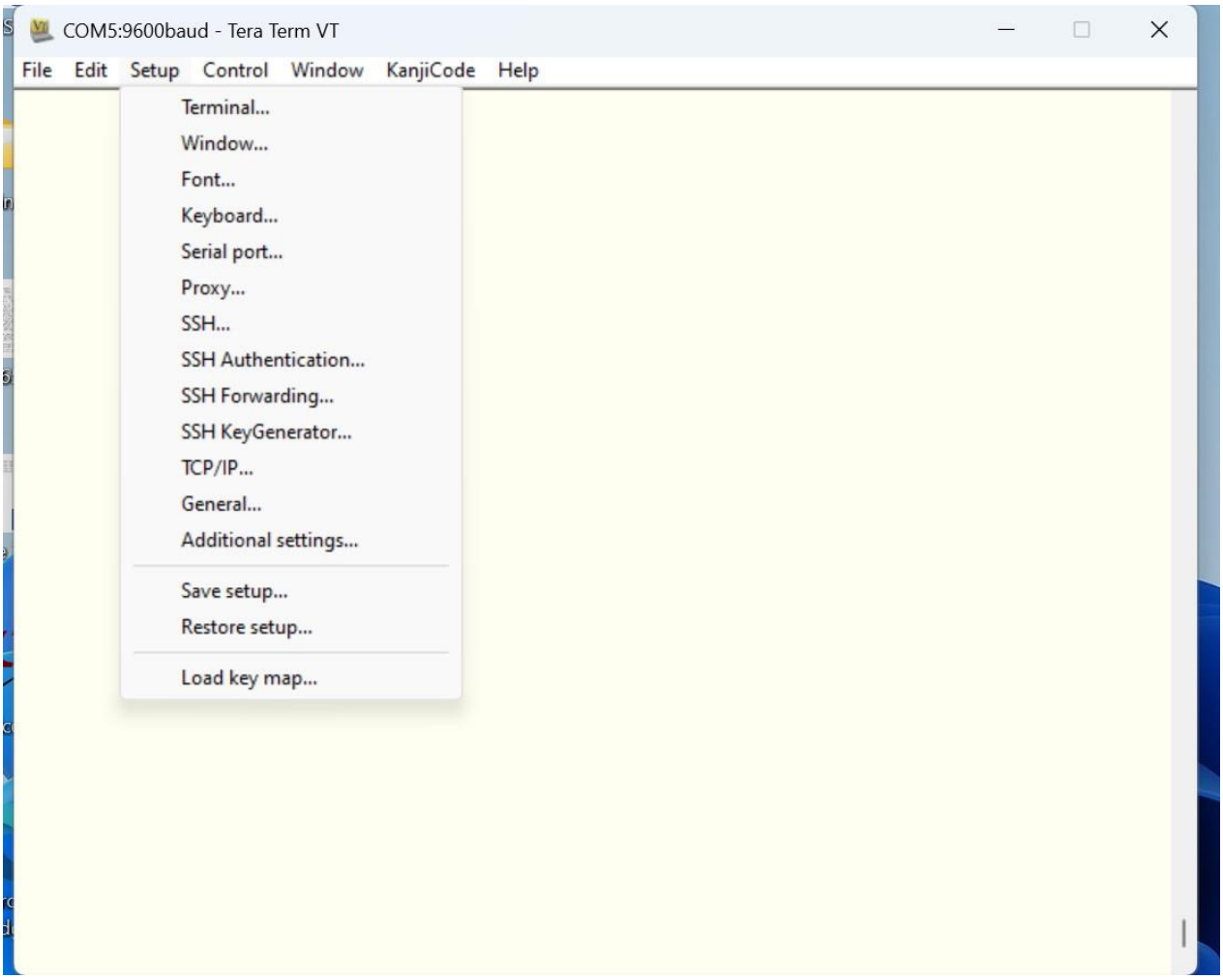

After connecting FPGA, below is the result

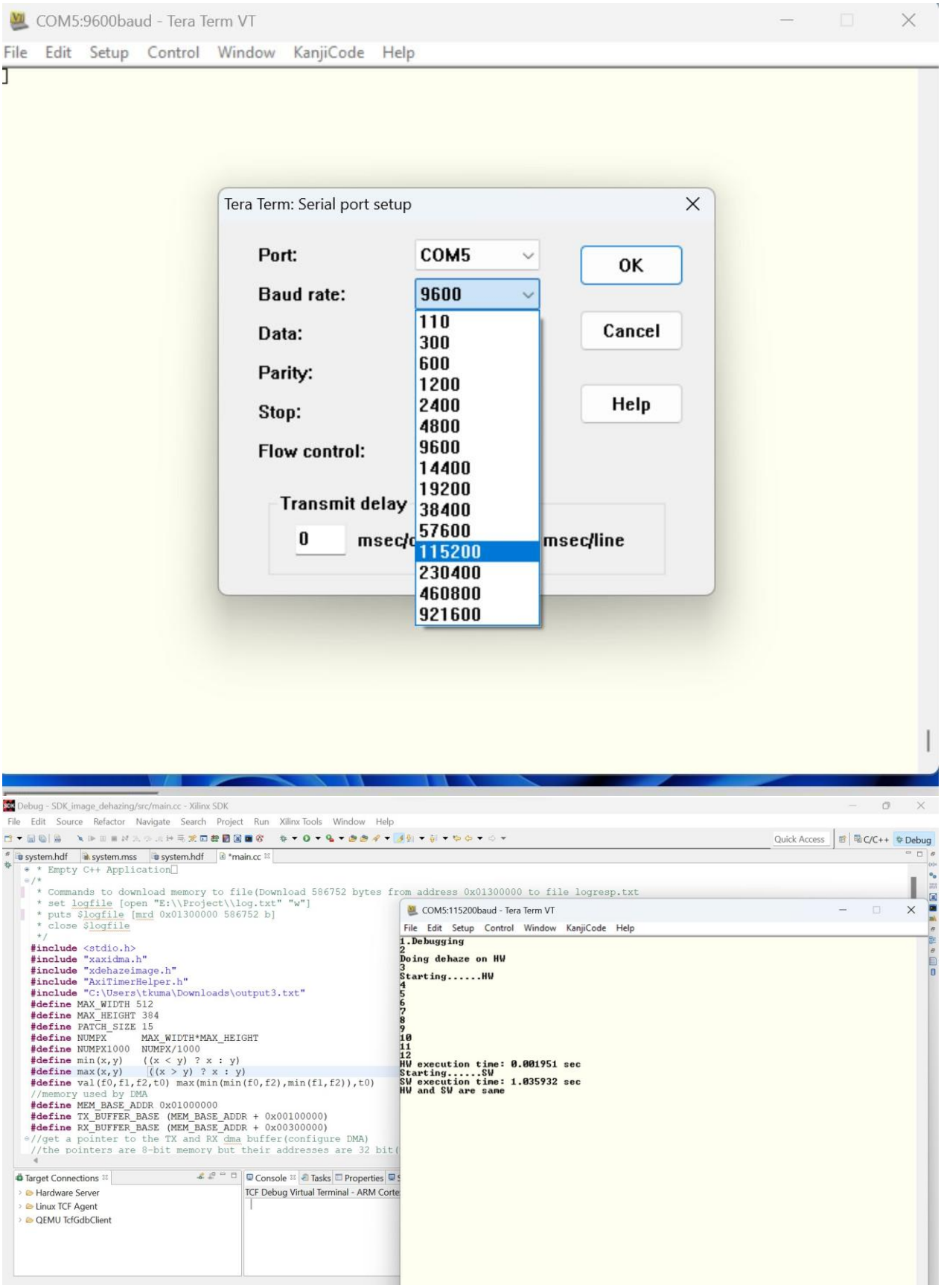



After launching on Hardware, Check the Tera term connection for results

Below is the link for drivers, which are to upgraded for connecting to teraterm:
https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers?tab=downloads


Tera Term: New connection
TCP/IP
Host: 127.0.0.1
History
Service: Telnet
TCP port#: 22
SSH
SSH version: SSH2
Other
Protocol: UNSPEC
Serial
Port: COM5: Silicon Labs CP210x USB to U
OK
Cancel
Help


COM5:9600baud - Tera Term VT
File Edit Setup Control Window KanjiCode Help
Terminal...
Window...
Font...
Keyboard...
Serial port...
Proxy...
SSH...
SSH Authentication...
SSH Forwarding...
SSH KeyGenerator...
TCP/IP...
General...
Additional settings...
Save setup...
Restore setup...
Load key map...


COM5:9600baud - Tera Term VT
File Edit Setup Control Window KanjiCode Help
Tera Term: Serial port setup
Port:
COM5
Baud rate:
9600
Data:
Parity:
Stop:
Flow control:
Transmit delay
0
msec/line
OK
Cancel
Help
110
300
600
1200
2400
4800
9600
14400
19200
38400
57600
115200
230400
460800
921600
Debug - SDK_image_dehazing/src/main.cc - Xilinx SDK
File Edit Source Refactor Navigate Search Project Run Xilinx Tools Window Help
system.hdf
system.mss
system.hdf
*main.cc
* Empty C++ Application
/*
* Commands to download memory to file(Download 586752 bytes from address 0x01300000 to file logresp.txt
* set logfile [open "E:\\Project\\log.txt" "w"]
* puts $logfile [mrd 0x01300000 586752 b]
* close $logfile
*/
#include <stdio.h>
#include "xaxidma.h"
#include "xdehazeimage.h"
#include "AxiTimerHelper.h"
#include "C:\Users\tkuma\Downloads\output3.txt"
#define MAX_WIDTH 512
#define MAX_HEIGHT 384
#define PATCH_SIZE 15
#define NUMPX MAX_WIDTH*MAX_HEIGHT
#define NUMPX1000 NUMPX/1000
#define min(x,y) ((x < y) ? x : y)
#define max(x,y) ((x > y) ? x : y)
#define val(f0,f1,f2,t0) max(min(min(f0,f2),min(f1,f2)),t0)
//memory used by DMA
#define MEM_BASE_ADDR 0x01000000
#define TX_BUFFER_BASE (MEM_BASE_ADDR + 0x00100000)
#define RX_BUFFER_BASE (MEM_BASE_ADDR + 0x00300000)
//get a pointer to the TX and RX dma buffer(configure DMA)
//the pointers are 8-bit memory but their addresses are 32 bit(
Target Connections
Hardware Server
Linux TCF Agent
QEMU TcfGdbClient
Console
Tasks
Properties
TCF Debug Virtual Terminal - ARM Corte
COM5:115200baud - Tera Term VT
File Edit Setup Control Window KanjiCode Help
1.Debugging
2
Doing dehaze on HW
3
Starting.....HW
4
5
6
7
8
9
10
11
12
HW execution time: 0.001951 sec
Starting.....SW
SW execution time: 1.035932 sec
HW and SW are same