

ΘΈΜΑ ΠΤΥΧΙΑΚΉΣ ΕΡΓΑΣΊΑΣ :

ΥΛΟΠΟΙΗΣΗ ΚΑΙ ΠΡΟΣΟΜΟΙΩΣΗ ΤΟΥ MICROBLAZE ΕΠΕΞΕΡΓΑΣΤΗ ΣΕ FPGA

ΑΡΙΘΜΌΣ ΜΗΤΡΏΟΥ :

2026202100134

ΟΝΟΜΑΤΕΠΏΝΥΜΟ :

ΑΝΑΣΤΑΣΙΟΣ ΠΟΥΤΑΧΊΔΗΣ

ΠΑΝΕΠΙΣΤΉΜΙΟ :

Πελοποννήσου.

ΤΜΉΜΑ :

Ψηφιακών Συστημάτων.

ΠΑΡΟΥΣΊΑΣΗ :

Πτυχιακής Εργασίας 20΄.

ΕΡΓΑΛΕΊΑ ΥΛΟΠΟΊΗΣΗΣ :

Vitis(SW DEVELOPER)2023.2

Click : Create Project > .



Vivado
ML Edition

Quick Start

[Create Project >](#)

[Open Project >](#)

[Open Example Project >](#)

Tasks

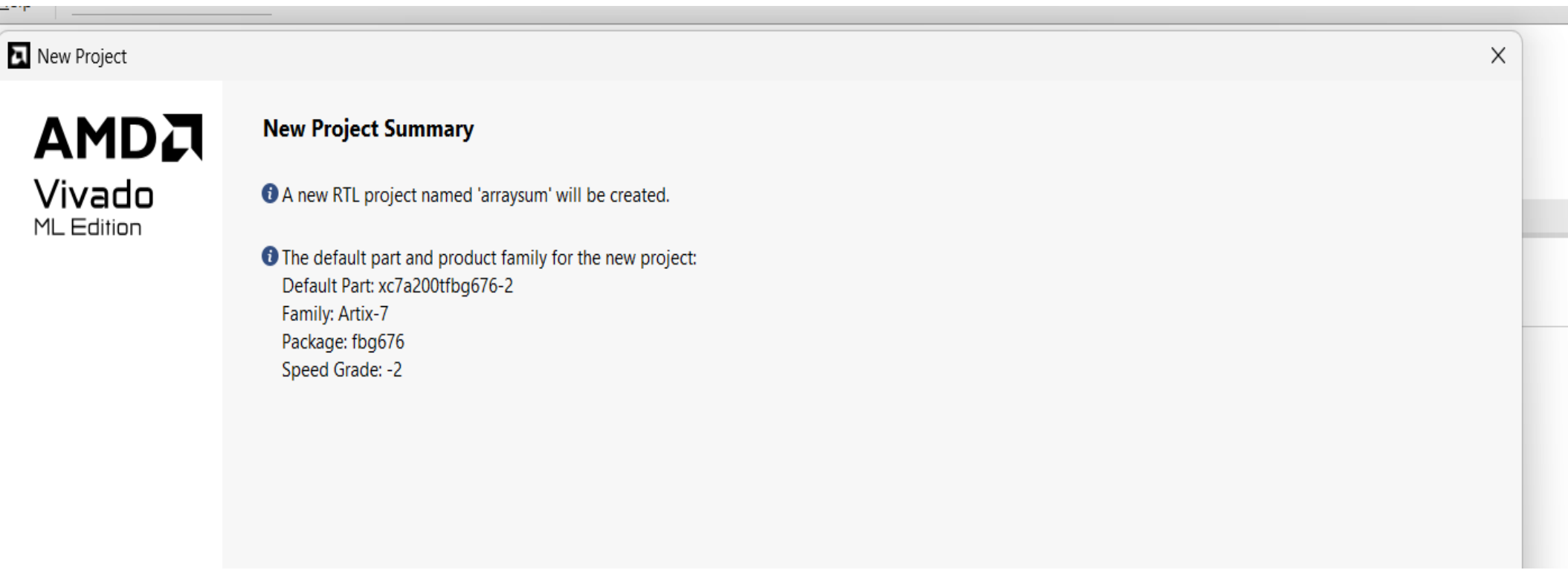
[Manage IP >](#)

[Open Hardware Manager >](#)

[Vivado Store >](#)

Project Name : arraysum

Xilinx Board Part : xc7a200tfbg676-2



Create Block Design : Add MicroBlaze and Run Block Automation.

The screenshot displays the Vivado 2023.2 IDE interface for a project named 'arraysum'. The main workspace is titled 'BLOCK DESIGN - design_1 *'. On the left, the 'Flow Navigator' pane shows the project structure with 'PROJECT MANAGER', 'IP INTEGRATOR', 'SIMULATION', 'RTL ANALYSIS', 'SYNTHESIS', and 'IMPLEMENTATION' sections. The 'IP INTEGRATOR' section is active, showing 'Create Block Design', 'Open Block Design', and 'Generate Block Design' options. The 'Sources' pane shows the 'design_1' project with a sub-entry 'microblaze_0 (MicroBlaze:11.0)'. The 'Properties' pane is empty, prompting the user to 'Select an object to see properties'. The 'Diagram' pane shows a block diagram of the 'microblaze_0' block, which is a 'MicroBlaze' core. The diagram includes input ports for 'INTERRUPT', 'DEBUG', 'Clk', and 'Reset', and output ports for 'DLMB' and 'ILMB'. A green banner at the top of the diagram pane states 'Designer Assistance available. [Run Block Automation](#)'. The 'Tcl Console' pane at the bottom shows the following commands and output:

```
Tcl Console x Messages Log Reports Design Runs
INFO: [IP_Flow 19-1704] No user IP repositories specified
INFO: [IP_Flow 19-2313] Loaded Vivado IP repository 'C:/Xilinx/Vivado/2023.2/data/ip'.
create_project: Time (s): cpu = 00:00:14 ; elapsed = 00:00:06 . Memory (MB): peak = 1418.355 ; gain = 0.000
set_property target_language VHDL [current_project]
create_bd_design "design_1"
Wrote : <C:/Users/znkr/Desktop/arraysum/arraysum.srcs/sources_1/bd/design_1/design_1.bd>
update_compile_order -fileset sources_1
startgroup
create_bd_cell -type ip -vlnv xilinx.com:ip:microblaze:11.0 microblaze_0
endgroup
```

The bottom of the console shows a prompt 'Type a Tcl command here'.

Set : Local Memory 128KB and Click OK.

(Improves Performance).

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.

Q

≡

≡

✓ ☒ All Automation (1 out of 1 selected)
✓ ☒ microblaze_0

Description

MicroBlaze connection automation generates local memory of selected size, and caches can be configured. MicroBlaze Debug Module, Peripheral AXI Interconnect, Interrupt Controller, a clock source, Processor System Reset are added and connected as needed. A preset MicroBlaze configuration can also be selected.

Information about the options can be found in the tooltips.

Options

Preset

None

Local Memory

128KB

Local Memory ECC

None

Cache Configuration

None

Debug Module

Debug Only

Peripheral AXI Port

Enabled

☐ Interrupt Controller

Clock Connection

New Clocking Wizard

?

OK

Cancel

Click : Run Connection Automation.

arraysum - [C:/Users/znkr/Desktop/arraysum/arraysum.xpr] - Vivado 2023.2

File Edit Flow Tools Reports Window Layout View Help Q Quick Access

Flow Navigator

- PROJECT MANAGER
 - Settings
 - Add Sources
 - Language Templates
 - IP Catalog
- IP INTEGRATOR
 - Create Block Design
 - Open Block Design
 - Generate Block Design
- SIMULATION
 - Run Simulation
- RTL ANALYSIS
 - Run Linter
 - Open Elaborated Design
- SYNTHESIS
 - Run Synthesis
 - Open Synthesized Design
- IMPLEMENTATION
 - Run Implementation
 - Open Implemented Design

BLOCK DESIGN - design_1*

Sources Design Signals

- design_1
 - Interface Connections
 - Nets
 - clk_wiz_1 (Clocking Wizard:6.0)
 - mdm_1 (MicroBlaze Debug Module (MDM):3.2)
 - microblaze_0 (MicroBlaze:11.0)

Block Properties

clk_wiz_1

Name: clk_wiz_1

General Properties IP

Diagram Address Editor

Designer Assistance available. Run Connection Automation

The diagram illustrates the hardware configuration for a MicroBlaze system. It includes a MicroBlaze core (microblaze_0) connected to a MicroBlaze Debug Module (MDM) (mdm_1). The MDM is connected to the MicroBlaze core's INTERRUPT, DEBUG, and SYS_Rst signals. The MicroBlaze core is also connected to a local memory block (microblaze_0_local_memory) via DLMB, ILMB, and LMB signals. A Clocking Wizard (clk_wiz_1) is connected to the MicroBlaze core's CLK_IN1_D and CLK_OUT1 signals. A Processor System Reset (rst_clk_wiz_1_100M) is connected to the MicroBlaze core's slowest_sync_clk, ext_reset_in, aux_reset_in, mb_debug_sys_rst, dcm_locked, mb_reset, bus_struct_reset[00], peripheral_reset[00], interconnect_aresetn[00], and peripheral_aresetn[00] signals.

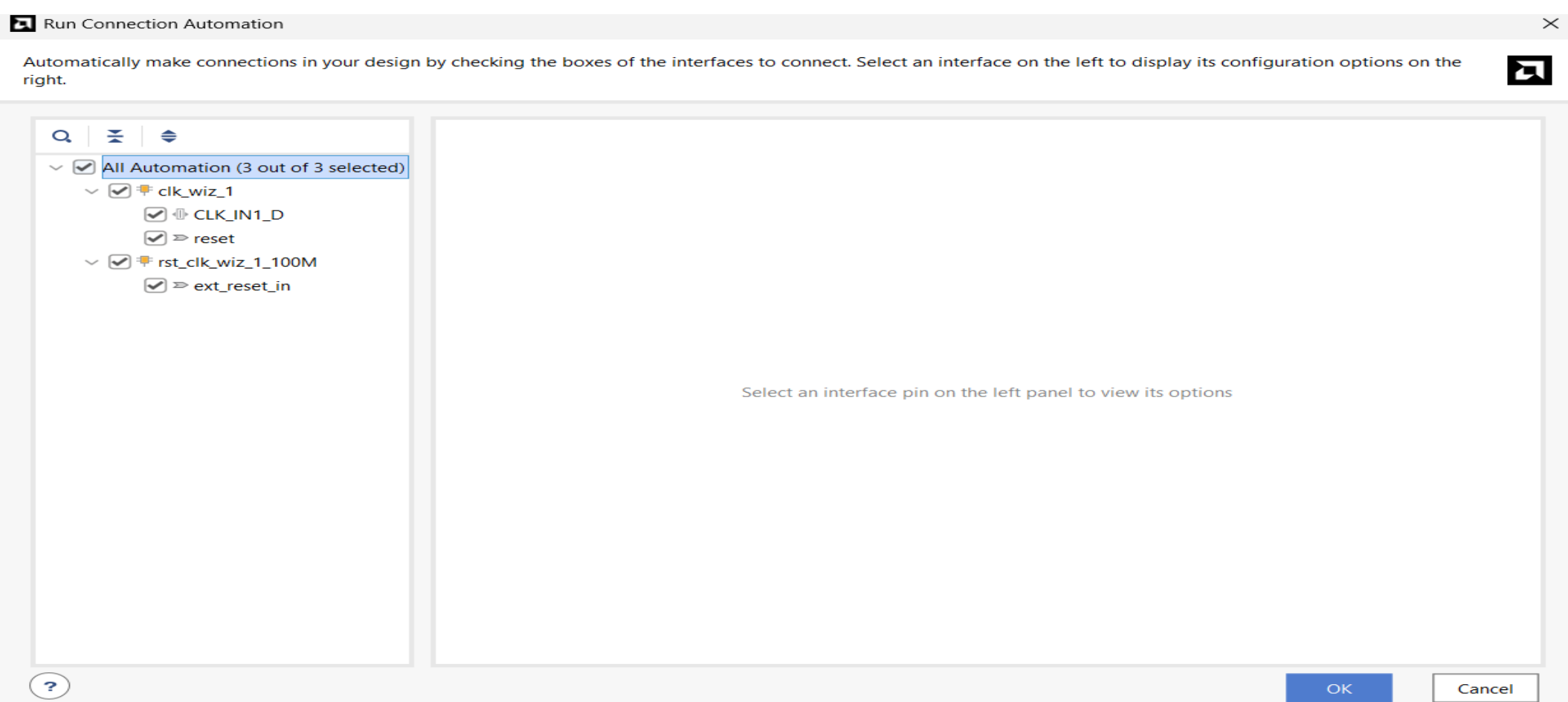
Tcl Console

Messages Log Reports Design Runs

```
update_compile_order -fileset sources_1
startgroup
create_bd_cell -type ip -vlnv xilinx.com:ip:microblaze:11.0 microblaze_0
endgroup
apply_bd_automation -rule xilinx.com:bd_rule:microblaze -config { axi_intc {0} axi_periph {Enabled} cache {None} clk {New Clocking Wizard} cores {1} debug_module {Debug On}
INFO: [Device 21-403] Loading part xc7a200tffbg676-2
create_bd_cell: Time (s): cpu = 00:00:10 ; elapsed = 00:00:22 . Memory (MB): peak = 1907.121 ; gain = 446.688
WARNING: [BD 5-700] No address spaces matched 'get_bd_addr_spaces /microblaze_0'
apply_bd_automation: Time (s): cpu = 00:00:22 ; elapsed = 00:00:34 . Memory (MB): peak = 1907.121 ; gain = 488.766
regenerate_bd_layout
```

Type a Tcl command here

Select : All(3 out of 3 selected) and Click OK.



Add BRAM and Run Connection Automation.

(BRAM : boosts system performance by minimizing the delay in data access).

arraysum - [C:/Users/znkr/Desktop/arraysum/arraysum.xpr] - Vivado 2023.2

File Edit Flow Tools Reports Window Layout View Help Q Quick Access

Flow Navigator

- PROJECT MANAGER
 - Settings
 - Add Sources
 - Language Templates
 - IP Catalog
- IP INTEGRATOR
 - Create Block Design
 - Open Block Design
 - Generate Block Design
- SIMULATION
 - Run Simulation
- RTL ANALYSIS
 - Run Linter
 - Open Elaborated Design
- SYNTHESIS
 - Run Synthesis
 - Open Synthesized Design
- IMPLEMENTATION
 - Run Implementation
 - Open Implemented Design

BLOCK DESIGN - design_1*

Sources Design Signals

design_1

- External Interfaces
- Interface Connections
- Ports
- Nets

Block Properties

clk_wiz_1

Name: clk_wiz_1

General Properties IP

Diagram Address Editor

Designer Assistance available. [Run Connection Automation](#)

diff_clock_rtl_0

reset_rtl_0

axi_bram_ctrl_0

clk_wiz_1

mdm_1

microblaze_0

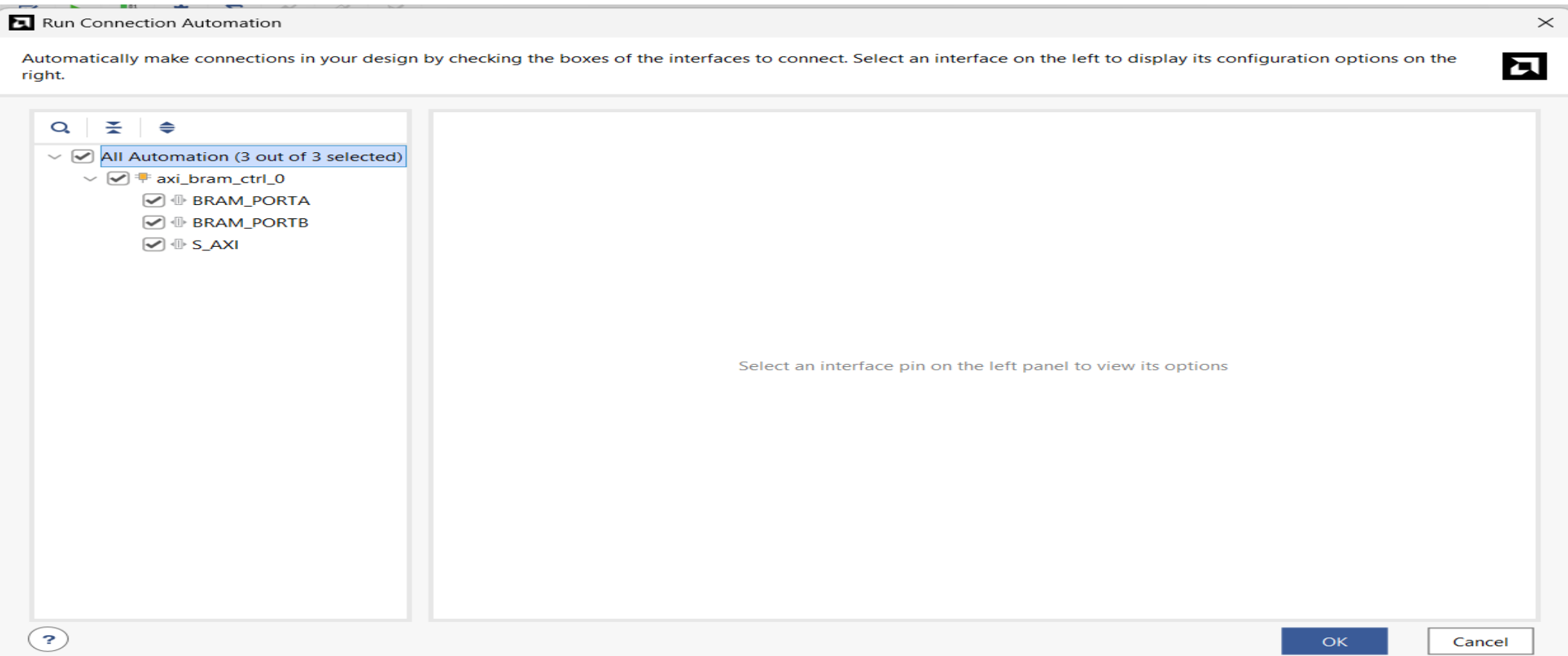
Tcl Console

Messages Log Reports Design Runs

```
INFO: [BoardRule 102-7] set_property CONFIG.POLARITY ACTIVE_LOW /reset_rtl_0
INFO: [BoardRule 102-15] connect_bd_net /reset_rtl_0 /rst_clk_wiz_1_100M/ext_reset_in
INFO: [BoardRule 102-19] set_property CONFIG.POLARITY ACTIVE_LOW /reset_rtl_0
endgroup
regenerate_bd_layout
startgroup
create_bd_cell -type ip -vlnv xilinx.com:ip:axi_bram_ctrl:4.1 axi_bram_ctrl_0
INFO: [xilinx.com:ip:axi_bram_ctrl:4.1-2] design_1_axi_bram_ctrl_0_0: In IP Integrator, please note that memory depth value gets calculated based on the Data Width of the
INFO: [xilinx.com:ip:axi_bram_ctrl:4.1-1] design_1_axi_bram_ctrl_0_0: In IP Integrator, The Maximum address range supported is 2G. Selecting the address range more than 2G
endgroup
```

Type a Tcl command here

Select : All(3 out of 3 selected) and Click OK.



Add UART and Set : Baud Rate 115200.

*Baud Rate:
Symbols/Second

(My testbench is written for this baud rate).

(UART : MicroBlaze can send and receive serial data with external devices).

Re-customize IP

AXI Uartlite (2.0)

Documentation IP Location

☐ Show disabled ports

☐ S_AXI
s_axi_aclk
s_axi_aresetn

UART
interrupt

Component Name axi_uartlite_0

AXI CLK Frequency 100 [10-300]MHz

Baud Rate 115200

Data Bits 8 [5 - 8]

Parity

☒ No Parity ☐ Odd ☐ Even

OK Cancel

Click : Run Connection Automation.

arraysum - [C:/Users/znkr/Desktop/arraysum/arraysum.xpr] - Vivado 2023.2

File Edit Flow Tools Reports Window Layout View Help Q Quick Access

Ready
Default Layout

Flow Navigator

- PROJECT MANAGER
 - Settings
 - Add Sources
 - Language Templates
 - IP Catalog
- IP INTEGRATOR
 - Create Block Design
 - Open Block Design
 - Generate Block Design
- SIMULATION
 - Run Simulation
- RTL ANALYSIS
 - Run Linter
 - Open Elaborated Design
- SYNTHESIS
 - Run Synthesis
 - Open Synthesized Design
- IMPLEMENTATION
 - Run Implementation
 - Open Implemented Design

BLOCK DESIGN - design_1*

Sources Design Signals

design_1

- External Interfaces
- Interface Connections
- Ports
- Nets

Block Properties

axi_uartlite_0

Name: axi_uartlite_0

General Properties IP

Diagram Address Editor

Designer Assistance available. [Run Connection Automation](#)

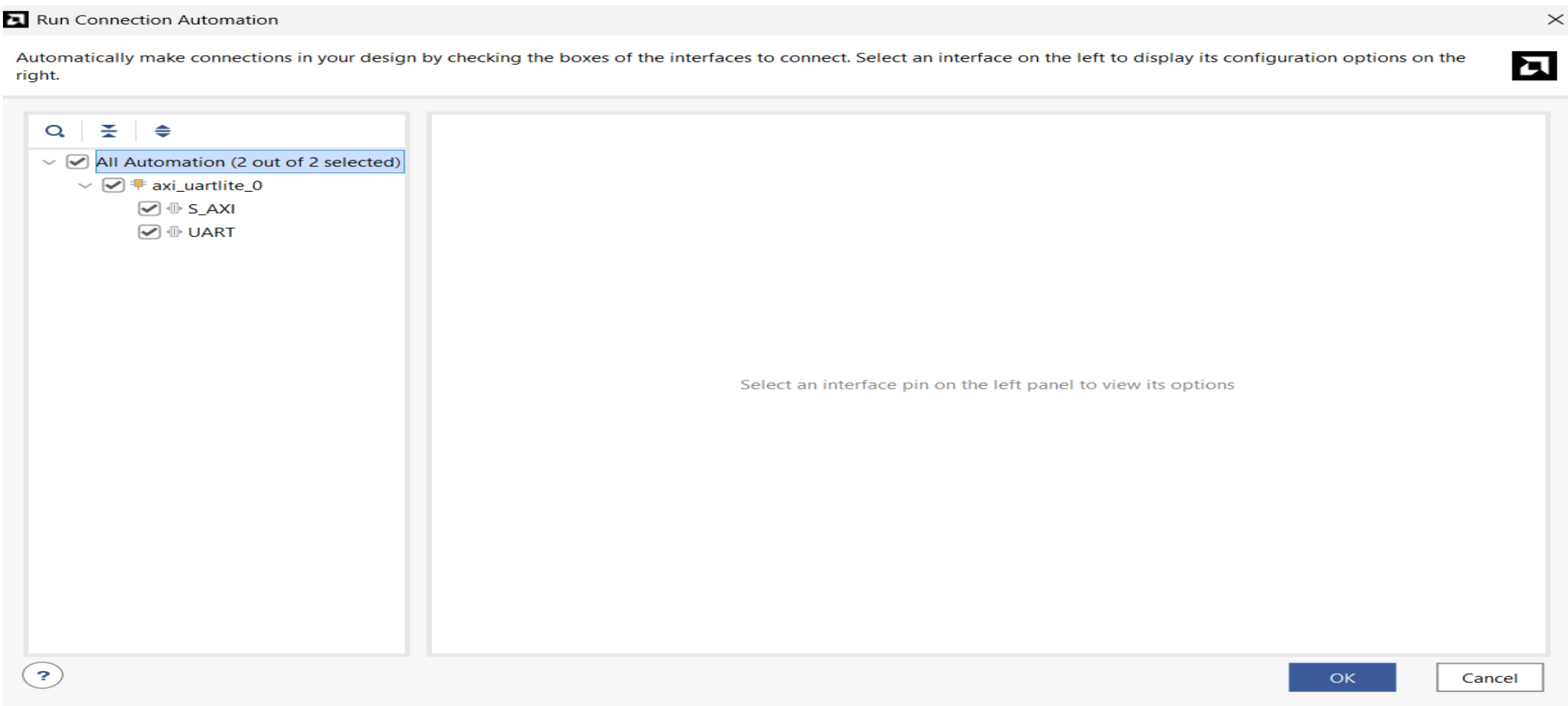
Diagram showing the connection of axi_uartlite_0 to microblaze_0 via mdm_1 (MicroBlaze Debug Module (MDM)).

Tcl Console

```
apply_bd_automation -rule xilinx.com:bd_rule:bram_cntlr -config {BRAM "Auto" } [get_bd_intf_pins axi_bram_ctrl_0/BRAM_PORTB]
apply_bd_automation -rule xilinx.com:bd_rule:axi4 -config { Clk_master {/clk_wiz_1/clk_out1 (100 MHz)} Clk_slave {Auto} Clk_xbar {Auto} Master {/microblaze_0 (Periph)} Slave segment '/axi_bram_ctrl_0/S_AXI/Mem0' is being assigned into address space '/microblaze_0/Data' at <0xc000_0000 [ 8K ]>.
endgroup
regenerate_bd_layout
startgroup
create_bd_cell -type ip -vlnv xilinx.com:ip:axi_uartlite:2.0 axi_uartlite_0
endgroup
set_property location {2 392 81} [get_bd_cells axi_uartlite_0]
set_property CONFIG.C_BAUDRATE {115200} [get_bd_cells axi_uartlite_0]
```

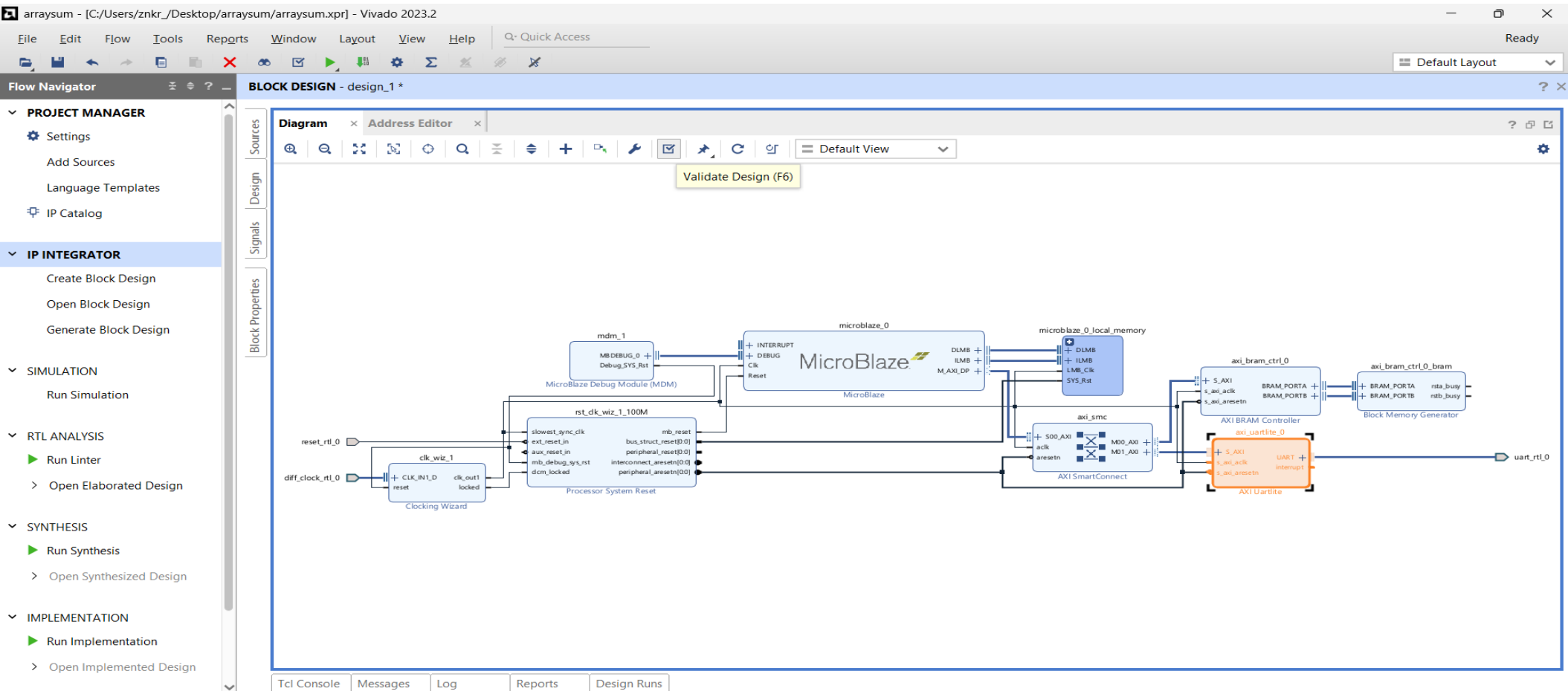
Type a Tcl command here

Select : All(3 out of 3 selected) and Click OK.



Click : Validate Design (F6).

(Given : The Form Of The Block Design).



Validate and display errors and critical warnings in this design

Click : OK. If The Validation Is Successful.
(This window will open).



Validate Design



Validation successful. There are no errors or critical warnings in this design.

OK

Click : Generate Output Products...

(For synthesizing the design and implementing it to test the system).

arraysum - [C:/Users/znkr/Desktop/arraysum/arraysum.xpr] - Vivado 2023.2

File Edit Flow Tools Reports Window Layout View Help Q Quick Access

Flow Navigator

- PROJECT MANAGER
 - Settings
 - Add Sources
 - Language Templates
 - IP Catalog
- IP INTEGRATOR
 - Create Block Design
 - Open Block Design
 - Generate Block Design
- SIMULATION
 - Run Simulation
- RTL ANALYSIS
 - Run Linter
 - Open Elaborated Design
- SYNTHESIS
 - Run Synthesis
 - Open Synthesized Design
- IMPLEMENTATION
 - Run Implementation
 - Open Implemented Design

BLOCK DESIGN - design_1*

Sources x Design Signals

- Design Sources (1)
- Constraints
- Simulation Sources (1)
 - sim_1 (1)
- Utility Sources

Source File Properties

Diagram x Address Editor x

Default View

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...

Copy Constraints Set...

Edit Constraints Sets...

Edit Simulation Sets...

Associate ELF Files...

Hierarchy IP Sources Libraries

Tcl Console Messages Log

Diagram

mdm_1

MBDEBUG_0 + Debug.SYS_Rst

MicroBlaze Debug Module (MDM)

microblaze_0

MicroBlaze

microblaze_0_local_r

axi_smc

AXI SmartConn

clk_wiz_1

CLK_IN1_D reset

clk_out1 locked

Clkgen Wizard

rst_clk_wiz_1_100M

slowest_sync_clk

ext_reset_in

aux_reset_in

mb_debug_sys_rst

dcm_locked

mb_reset

bus_struct_reset(D0)

peripheral_reset(D0)

interconnect_aresetn(D0)

peripheral_aresetn(D0)

Processor System Reset

INTERRUPT + DEBUG

DLMB + ILMB + M_AXI_DP

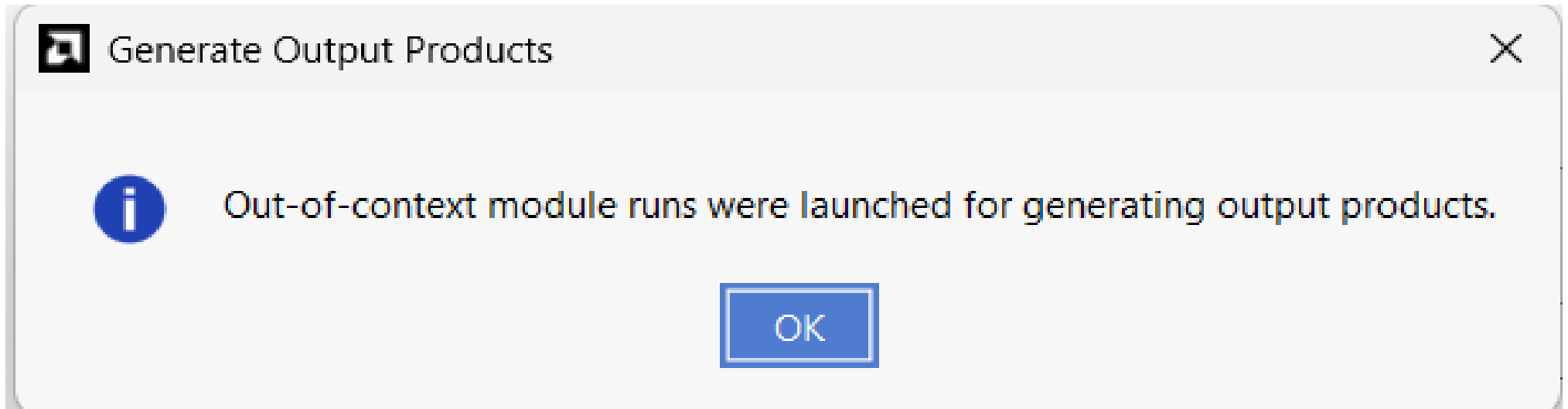
500_Axi

ack

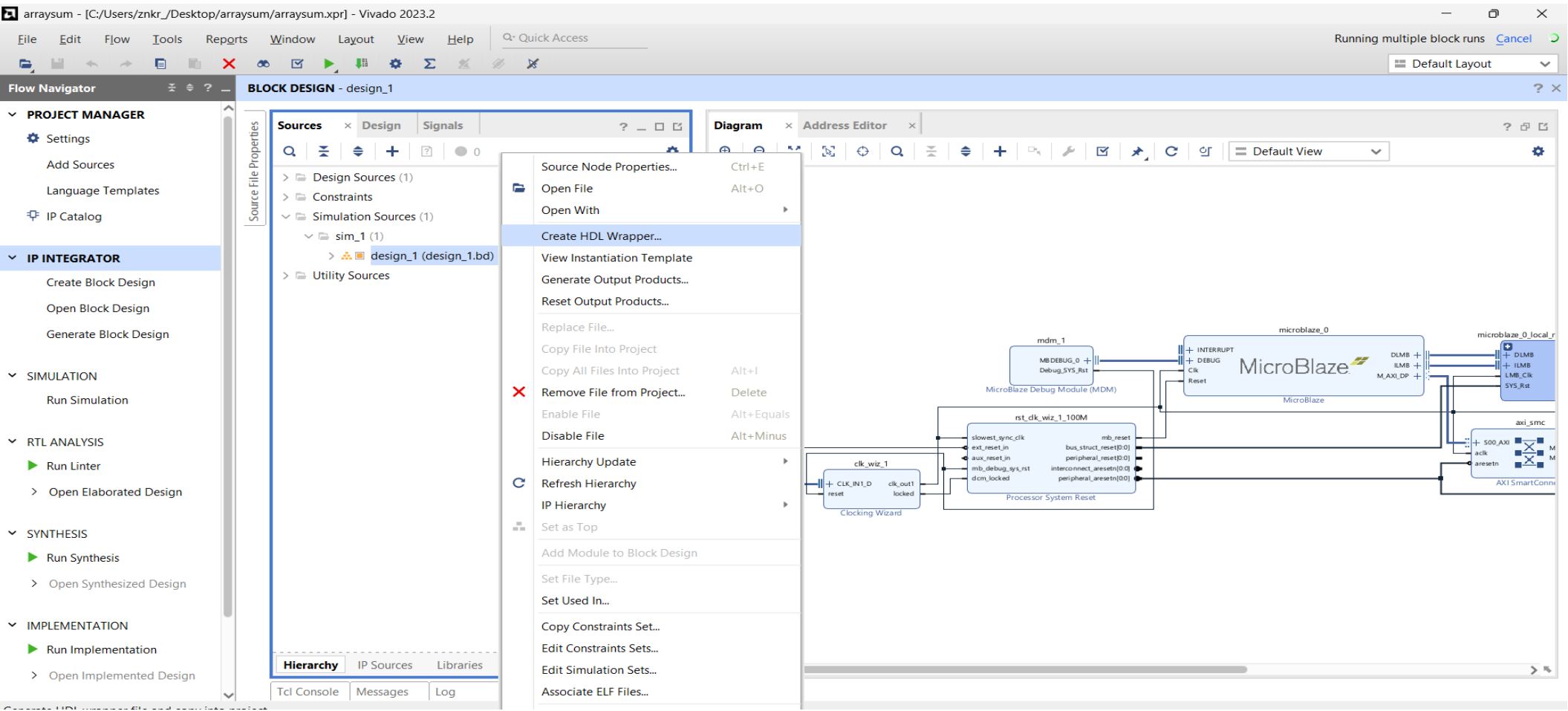
aresetn

***Click : OK. If the synthesis and implementation
are successful.***

(This window will open).

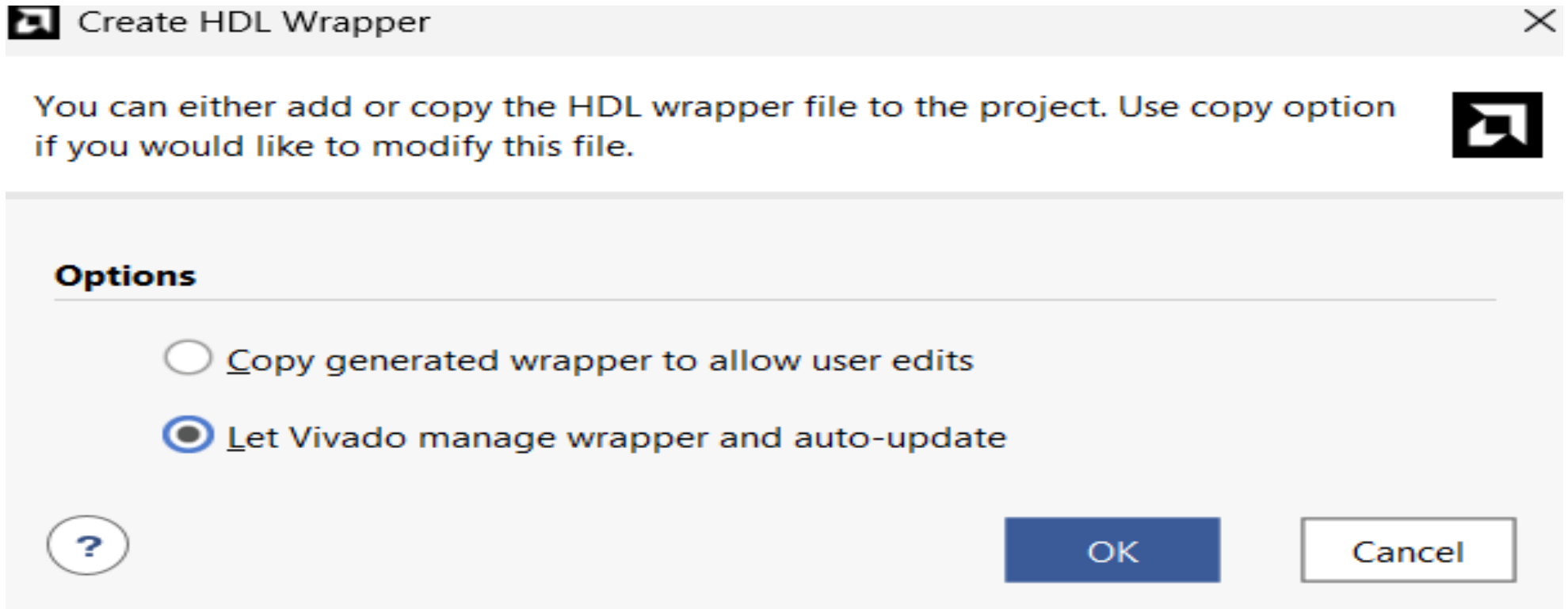


Click : Create HDL Wrapper...
(Manages the design within the FPGA).



Click : OK.

(Let Vivado manage wrapper and auto-update).



HDL Wrapper Generated.

(With name : design_1_wrapper.vhd).

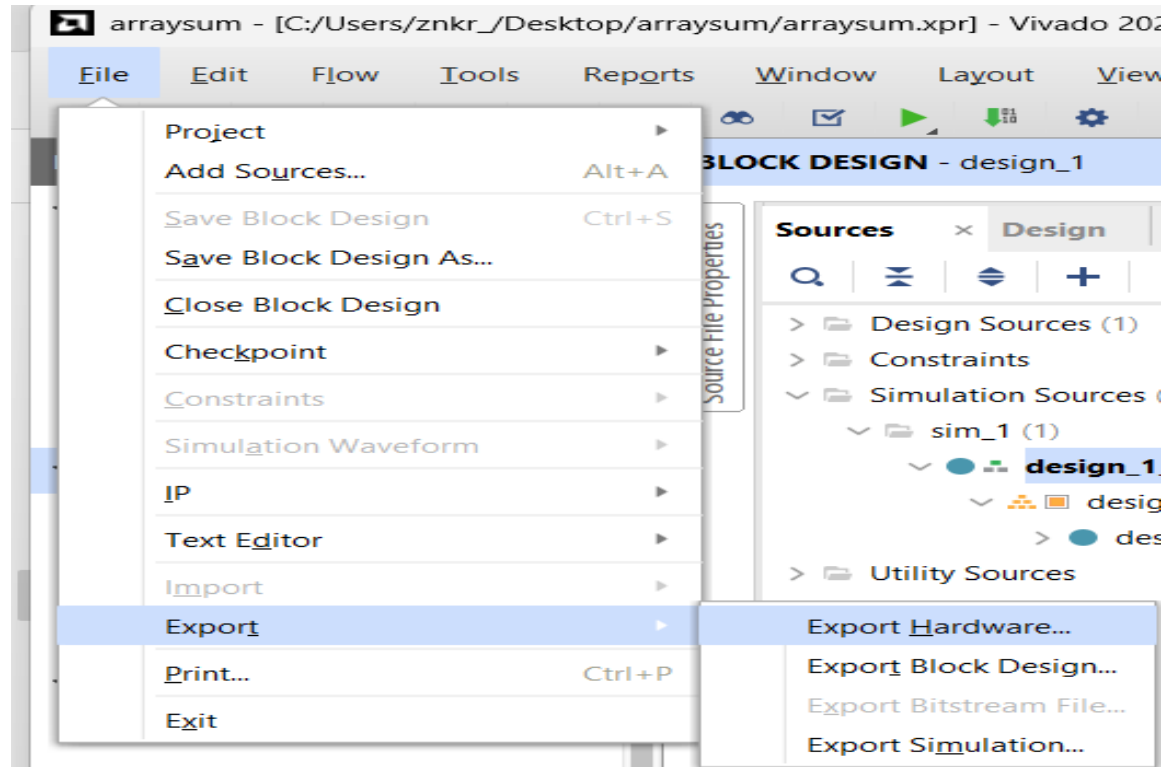
The screenshot displays the Vivado 2023.2 IDE interface. The top menu bar includes File, Edit, Flow, Tools, Reports, Window, Layout, View, and Help. The toolbar contains various icons for file operations, simulation, and design management. The left sidebar shows the Project Manager with sections for Settings, Add Sources, Language Templates, IP Catalog, IP Integrator, SIMULATION, RTL ANALYSIS, SYNTHESIS, and IMPLEMENTATION. The central Sources panel shows the Design Sources (1) and Simulation Sources (1) for the project design_1. The right pane shows the Code Editor with the generated HDL wrapper file, design_1_wrapper.vhd. The code includes copyright information, tool version details, and the VHDL entity and architecture definitions for the design_1_wrapper.

```
--Copyright 1986-2022 Xilinx, Inc. All Rights Reserved.
--Copyright 2022-2023 Advanced Micro Devices, Inc. All Rights Reserved.
-----
--Tool Version: Vivado v.2023.2 (win64) Build 4029153 Fri Oct 13 20:14:34 MDT 2023
--Date       : Fri Dec 27 16:52:20 2024
--Host       : rev_sharped running 64-bit major release (build 9200)
--Command    : generate_target design_1_wrapper.bd
--Design     : design_1_wrapper
--Purpose    : IP block netlist
-----
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
library UNISIM;
use UNISIM.VCOMPONENTS.ALL;
entity design_1_wrapper is
    port (
        diff_clock_rtl_0_clk_n : in STD_LOGIC;
        diff_clock_rtl_0_clk_p : in STD_LOGIC;
        reset_rtl_0 : in STD_LOGIC;
        uart_rtl_0_rxd : in STD_LOGIC;
        uart_rtl_0_txd : out STD_LOGIC
    );
end design_1_wrapper;

architecture STRUCTURE of design_1_wrapper is
    component design_1 is
        port (
            diff_clock_rtl_0_clk_n : in STD_LOGIC;
            diff_clock_rtl_0_clk_p : in STD_LOGIC;
            reset_rtl_0 : in STD_LOGIC;
            uart_rtl_0_rxd : in STD_LOGIC;
            uart_rtl_0_txd : out STD_LOGIC
        );
    end component design_1;
end STRUCTURE;
```


Click : Export Hardware...


(implementing the design on an FPGA).



Click : Next > .

(pre-synthesis ensures that the hardware is ready for software development).

 Export Hardware Platform ×

Output 

Set the platform properties to inform downstream tools of the intended use of the target platform's hardware design.

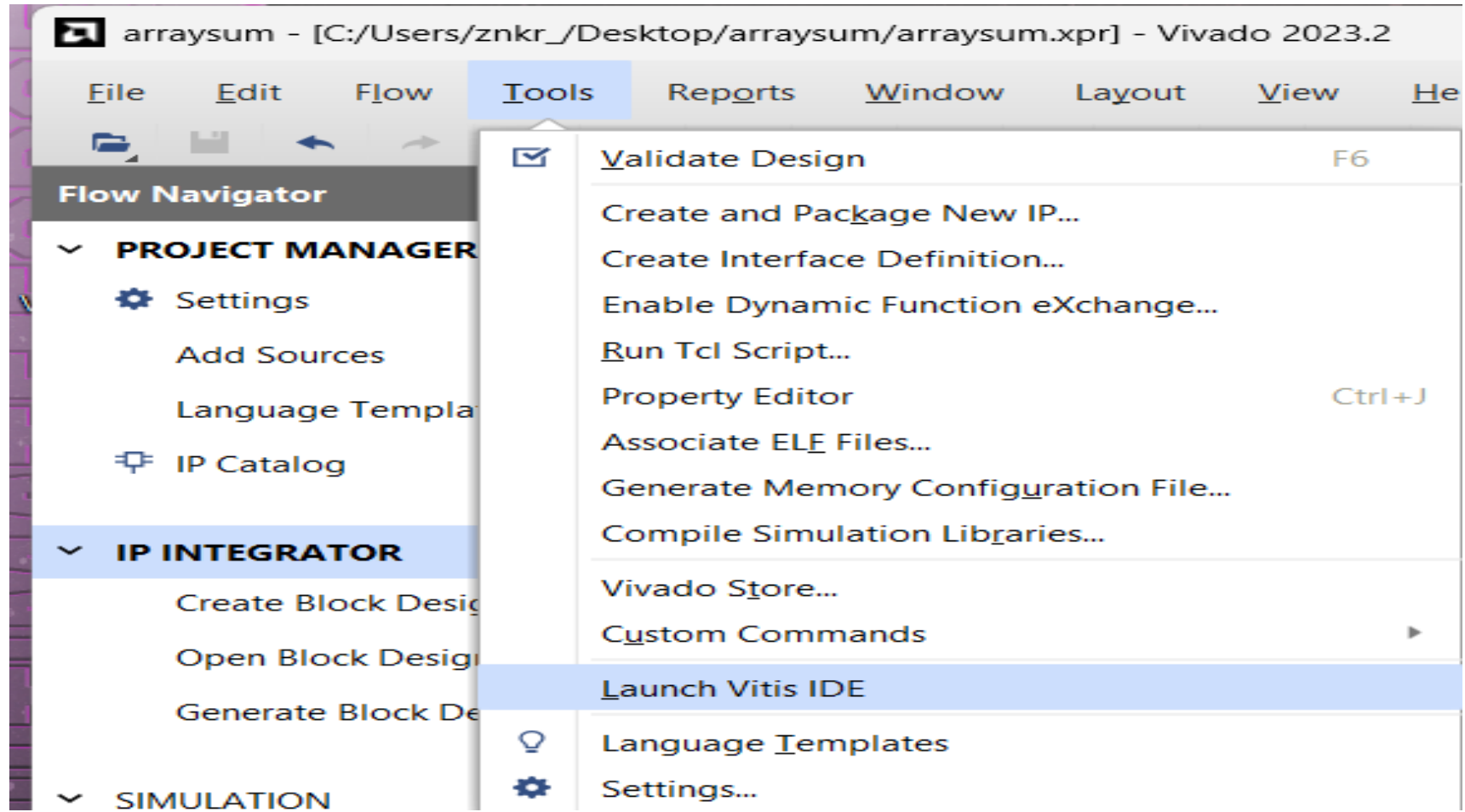
☒ **Pre-synthesis**
This platform includes a hardware specification for downstream software tools.

☐ **Include bitstream**
This platform includes the complete hardware implementation and bitstream, in addition to the hardware specification for software tools.

< Back Next > Finish Cancel

Click : Launch Vitis IDE .

(Make sure .xsa file is exported successfully).



Open the *project workspace* and create a platform.

(Platform : based on the .xsa file).

The screenshot displays the Vitis Unified IDE 2023.2 interface. On the left, the 'VITIS COMPONENTS' tree shows the project structure, including 'platform' (Platform) and 'vitis-comp.json'. The 'platform' component is selected, and its 'Settings' are visible. The 'FLOW' section at the bottom left shows the 'Component' dropdown set to 'platform' and a 'Build' button.

The main editor area shows the 'platform' component selected in the 'vitis-comp.json' file. The 'Platform - platform' configuration panel is visible on the right, showing the 'Name' as 'platform', the 'Hardware Specification' as 'design_1_wrapper.xsa', and a 'Description' field.

The bottom panel shows the 'OUTPUT' and 'PROBLEMS' tabs. The 'OUTPUT' tab is active, displaying the following log messages:

```
17:29:20 INFO : -- Build files have been written to: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/
17:29:20 INFO : Successfully created Domain at C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp
17:29:20 INFO : Successfully Generated Domain C:\Users\znkr\OneDrive\Desktop\arraysum\platform\microblaze_0\standalone_microblaze_0\bsp
17:29:20 INFO : Domain standalone_microblaze_0 added successfully.
17:29:20 INFO : Platform creation finished successfully.
17:29:20 INFO : Platform Quick Build initiated.
17:29:20 INFO : Generating Export directory
17:29:20 INFO : Generated platform metadata for creating application(s) based on platform platform.
```

The bottom right corner shows a notification: 'Created platform: platform'.

Click : Build.

(Build : To build the platform).

The screenshot displays the Vitis Unified IDE 2023.2 interface. The top menu bar includes File, Edit, Selection, View, Go, Terminal, and Help. The left sidebar shows the 'VITIS COMPONENTS' tree with 'ARRAYSUM' expanded, containing 'platform' (Platform) and 'Settings'. The 'platform' component is selected, showing its 'vitis-comp.json' file. The main editor area displays the 'Platform - platform' configuration page, which includes fields for Name (platform), Hardware Specification (design_1_wrapper.xsa, Switch XSA, Hardware Specification), and Description. The 'Build' button is visible in the bottom left corner of the main editor area. The bottom status bar shows the output of the build process, indicating that the platform was generated successfully.

Platform - platform

Name: platform

Hardware Specification: design_1_wrapper.xsa [Switch XSA](#) [Hardware Specification](#)

Description:

Build

OUTPUT **PROBLEMS**

Vitis Server platform

```
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/include/./include/xil_types.h
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/include/./include/xil_util.h
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/include/./include/xinterrupt_wrap.h
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/include/./include/xmem_config.h
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/include/./include/xmicroblaze.h
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/include/./include/xmicroblaze_config.h
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/include/./include/xplatform_info.h
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/include/./include/xstatus.h
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/include/./include/xuartlite.h
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/include/./include/xuartlite_i.h
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/include/./include/xuartlite_l.h
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/lib/libxilstandalone.a
-- Installing: C:/Users/znkr/OneDrive/Desktop/arraysum/platform/microblaze_0/standalone_microblaze_0/bsp/lib/libxiltimer.a
Generating Export directory
Platform Build Finished successfully.
[12/27/2024, 5:30:19 PM]: Generate platform platform with id '82885633-1520-42c3-a4d7-da1cf7ef6b45' ended.
```

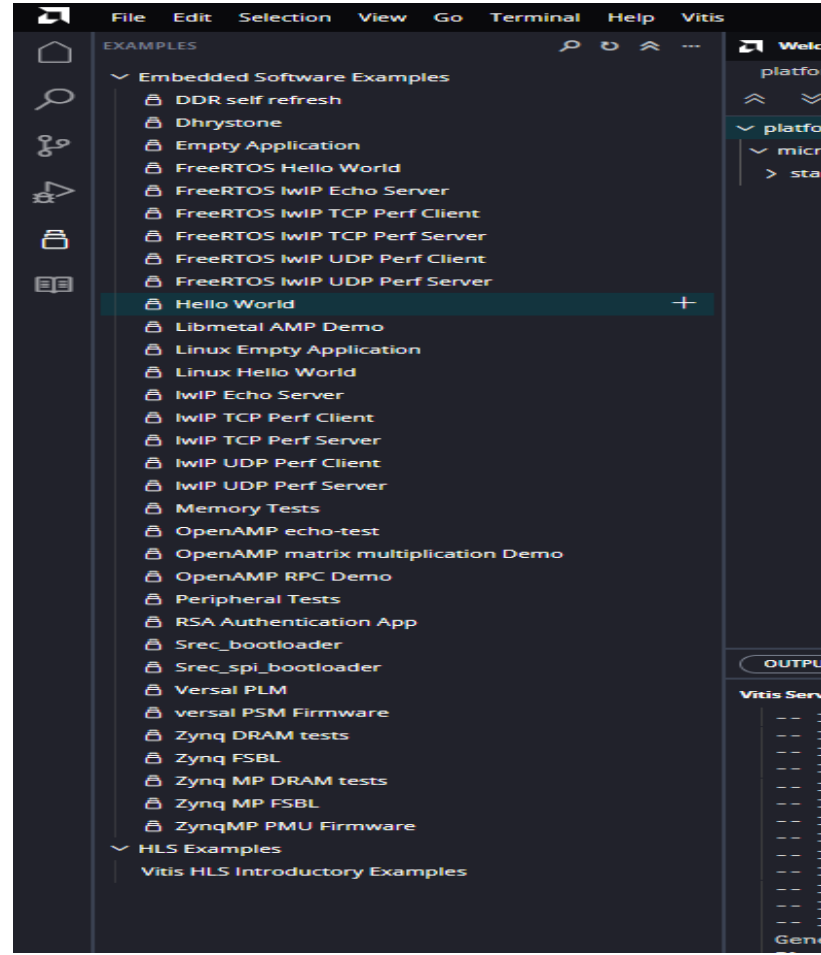
platform

- Vitis Server
- Git
- GitLens

Generate platform platform finished successfully.

Select : Hello World Application From Templates.

(Renamed: Array sum sdk).



Program : helloworld.c

(Calculates the sum of an array of nine floats, stores the result in BRAM and prints the sum via UART).

The screenshot displays the Vitis Unified IDE 2023.2 interface. The left sidebar shows the project structure for 'ARRAYSUM', including 'Array_sum_sdk' (Application) and 'platform' (Platform). The central code editor shows the 'helloworld.c' file with the following code:

```
1 #include <stdio.h> // Standard Input/Output library for printf
2 #include "platform.h" // Used for platform initialization and cleanup
3 #include "xil_printf.h" // Provides functions for printing via UART
4 #include "xil_io.h" // Provides functions for hardware register read/write (BRAM, IP)
5 #include "xparameters.h" // Contains base addresses for hardware resources (BRAM, IP)
6
7 int main()
8 {
9     init_platform(); // Initializes the platform (e.g., UART, STDIN/STDOUT)
10
11     char buff[11]; // Buffer to store the result as a string (up to 10 characters + null terminator)
12
13     // Define a static array of 9 float values
14     static float arr[9] = {1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0};
15
16     // Union for converting between unsigned int and float
17     union {
18         unsigned int a; // 32-bit unsigned integer
19         float b; // Float value
20     } itof;
21
22     float sum = 0.0; // Variable to store the sum of the array elements
23
24     // **Write array values to BRAM and calculate the sum**
25     for (int i = 0; i < 9; i++) {
26         itof.b = arr[i]; // Copy the current float element into the union
27         // Write the value (converted to unsigned int) to the BRAM address
28         Xil_Out32(XPAR_AXI_BRAM_0_BASEADDRESS + (4 * i), itof.a);
29         sum += arr[i]; // Calculate the sum of the elements
30     }
31
32     itof.b = sum; // Store the result (sum) in the union
33
34     //Print the result via UART:
35     sprintf(buff, "%f", itof.b); //Convert the result (float) to a string and store it in buff
36     xil_printf("Sum = %s\n\r", buff); // Print the result string via UART
37
38     //MY_OUTPUT_SIGNAL FROM THIS SIMULATION: charbuffer[1:20] : WROTE AS A STRING AND AFTER SOME SECONDS PRINTED RESULT IN TCL CONSOLE!
39
40     cleanup_platform(); // Cleanup the platform
41     return 0;
42 }
43
44 //1. This program CALCULATES the SUM of an ARRAY OF FLOATS.
45 //2. Stores the VALUES and RESULT in BRAM.
46 //3. Prints the SUM via UART.
```

The bottom status bar indicates 'No problems have been detected in the workspace so far.'

Click : Build.

(Build : To build the Application and generate ELF File).

The screenshot displays the Vitis Unified IDE interface for the 'Array_sum_sdk' application. The left sidebar shows the project structure with 'Array_sum_sdk' selected. The main editor window shows the 'helloworld.c' source file with the following code:

```
1 #include <stdio.h> // Standard Input/Output library for printf
2 #include "platform.h" // Used for platform initialization and cleanup
3 #include "xil_printf.h" // Provides functions for printing via UART
4 #include "xil_io.h" // Provides functions for hardware register read/write (BRAM, IP)
5 #include "xparameters.h" // Contains base addresses for hardware resources (BRAM, IP)
6
7 int main()
8 {
9     init_platform(); // Initializes the platform (e.g., UART, STDOUT)
10
11     char buff[11]; // Buffer to store the result as a string (up to 10 characters + null terminator)
12
13     // Define a static array of 9 float values
14     static float arr[9] = {1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0};
15
16     // Union for converting between unsigned int and float
17     union {
18         unsigned int a; // 32-bit unsigned integer
19         float b; // Float value
20     } itof;
21
22     float sum = 0.0; // Variable to store the sum of the array elements
23
24     // ... (rest of the code is obscured by the output window)
```

The bottom panel shows the 'OUTPUT' window with the following build log:

```
-- Generating done
-- Build files have been written to: C:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/build
C:/Xilinx/Vitis/2023.2/tps/win64/cmake-3.24.2/bin/cmake.exe -SC:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/src -BC:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/build
C:/Xilinx/Vitis/2023.2/tps/win64/cmake-3.24.2/bin/cmake.exe -E cmake_progress_start C:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/build
make -f CMakeFiles/Makefile2 all
make[1]: Entering directory 'c:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/build'
make -f CMakeFiles/Array_sum_sdk.elf.dir/build.make CMakeFiles/Array_sum_sdk.elf.dir/depend
make[2]: Entering directory 'c:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/build'
make[2]: Leaving directory 'c:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/build'
make -f CMakeFiles/Array_sum_sdk.elf.dir/build.make CMakeFiles/Array_sum_sdk.elf.dir/build
make[2]: Entering directory 'c:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/build'
[ 33%] Building C object CMakeFiles/Array_sum_sdk.elf.dir/helloworld.c.obj
C:/Xilinx/Vitis/2023.2/gnu/microblaze/nt/bin/mb-gcc.exe -isystem C:/Users/znkr/_OneDrive/Desktop/arraysum/platform/export/platform/sw/standalone
[ 66%] Building C object CMakeFiles/Array_sum_sdk.elf.dir/platform.c.obj
C:/Xilinx/Vitis/2023.2/gnu/microblaze/nt/bin/mb-gcc.exe -isystem C:/Users/znkr/_OneDrive/Desktop/arraysum/platform/export/platform/sw/standalone
[100%] Linking C executable Array_sum_sdk.elf
C:/Xilinx/Vitis/2023.2/gnu/microblaze/nt/bin/mb-gcc.exe -O2 -mlittle-endian -mxl-soft-mul -mcpu-v11.0 -DSDT -MMD -MP -specs=C:/Users/znkr/_OneDrive/Desktop/arraysum/platform/export/platform/sw/standalone
mb-size --format=berkeley Array_sum_sdk.elf
text      data      bss      dec      hex filename
95040     2104     2144     99288     18348 Array_sum_sdk.elf
mb-size --format=berkeley Array_sum_sdk.elf > C:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/build/Array_sum_sdk.elf.size
make[2]: Leaving directory 'c:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/build'
[100%] Built target Array_sum_sdk.elf
make[1]: Leaving directory 'c:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/build'
C:/Xilinx/Vitis/2023.2/tps/win64/cmake-3.24.2/bin/cmake.exe -E cmake_progress_start C:/Users/znkr/_OneDrive/Desktop/arraysum/Array_sum_sdk/build
Build Finished successfully
[12/27/2024, 6:00:04 PM]: Build for Array_sum_sdk:: with id '65b1a4b7-e5b6-449f-9d99-b44222e703c9' ended.
```

The right sidebar shows the 'Array_sum_sdk' project structure with the following components:

- Array_sum_sdk
 - Vitis Server
 - clangd
 - Git
 - GitLens
 - HLS Pragma Language Server
 - platform

Right-click : Block design and associate ELF files.

(Apply the ELF file from SDK only to the simulation sources).

arraysum - [C:/Users/znkr_/OneDrive/Desktop/arraysum/arraysum.xpr] - Vivado 2023.2

File Edit Flow Tools Reports Window Layout View Help Quick Access

Flow Navigator PROJECT MANAGER - arraysum

PROJECT MANAGER

- Settings
- Add Sources
- Language Templates
- IP Catalog
- IP INTEGRATOR
 - Create Block Design
 - Open Block Design
 - Generate Block Design
- SIMULATION
 - Run Simulation
- RTL ANALYSIS
 - Run Linter
 - Open Elaborated Design
- SYNTHESIS
 - Run Synthesis
 - Open Synthesized Design
- IMPLEMENTATION
 - Run Implementation
 - Open Implemented Design

Sources

- Simulation Sources (2)
 - sim_1 (2)
 - design_1_wrapper (STRUCTURE) (design_1_wrapper.vhd) (1)
 - design_1_i : design_1 (design_1.bd) (1)
 - design_1 (STRUCTURE) (design_1.vhd) (9)

Source File Properties

design_1.bd

☒ Enabled

Location: C:/Users/znkr_/OneDrive/Desktop/arraysum/arraysum.srcs/sources_1/bd/design_1

General Properties

Tcl Console Messages Log Reports **Design Runs**

Name	Constraints	Status	WNS	TNS	W
synth_1 (active)	constrs_1	Not started			
impl_1	constrs_1	Not started			
Out-of-Context Module Runs					
design_1		Submodule Runs Complete			

Associate ELF Files

Associate an ELF file with a processor instance (Address Map). ELF files are available after running generate on your embedded design sources. Only processors that are visible from the active top of the design will be shown.

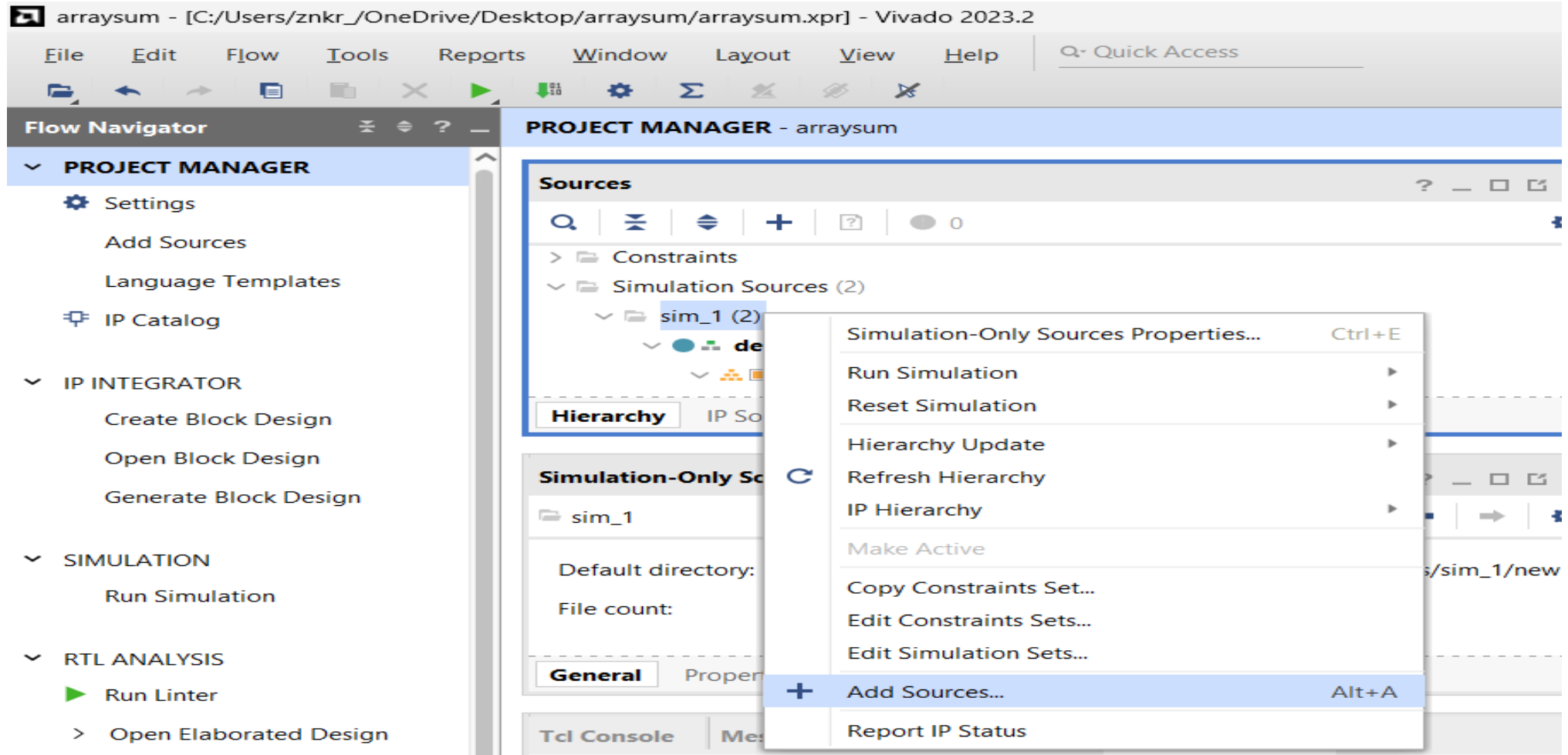
Processors/Address Maps	Associated ELF File
Design Sources <ul style="list-style-type: none">design_1<ul style="list-style-type: none">microblaze_0	mb_bootloop_le.elf
Simulation Sources <ul style="list-style-type: none">design_1<ul style="list-style-type: none">microblaze_0	Array_sum_sdkelf

OK Cancel

Not started
No errors or warnings

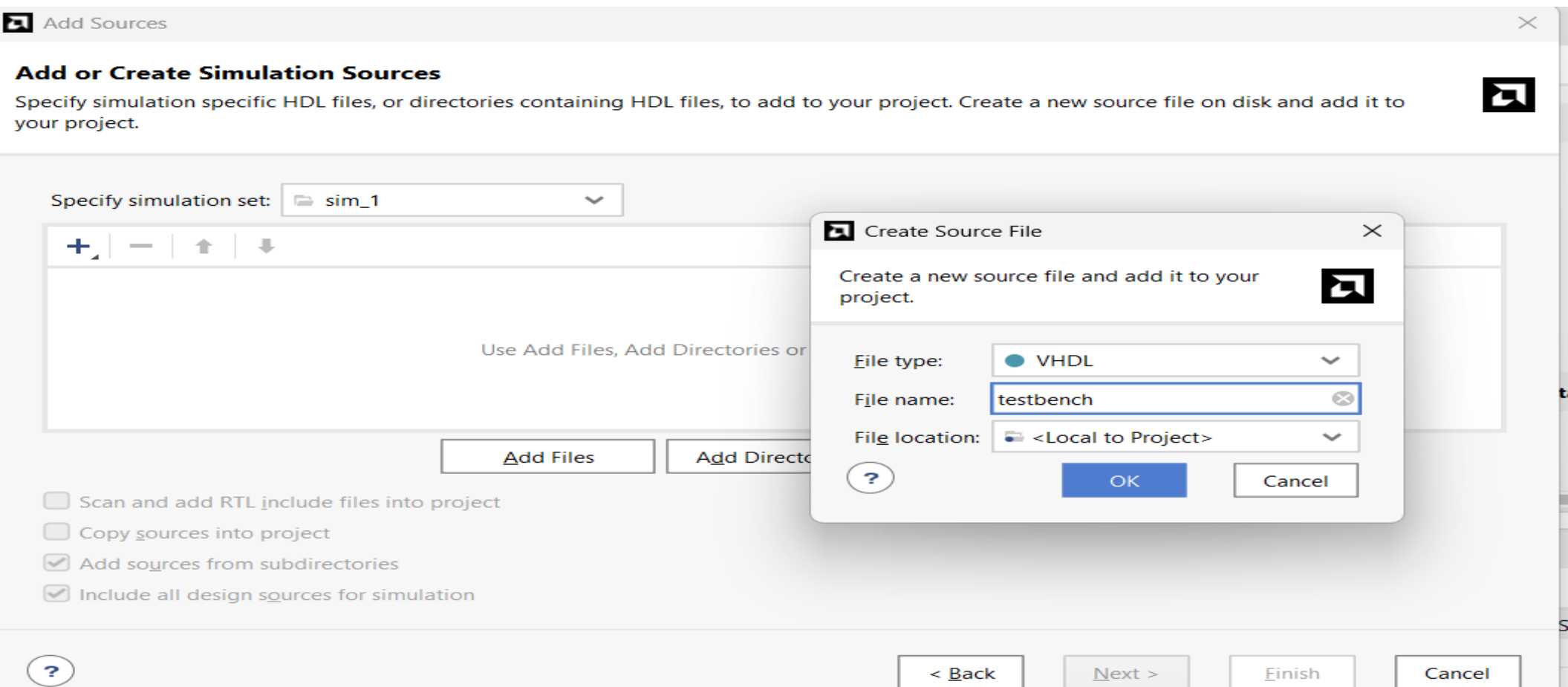
Associate ELF Files

Right-click : Simulation Sources and Add Sources...



Click : Create Source File.

(File name : testbench).



Click : Run Behavioral Simulation.

(The testbench contains VHDL code for UART communication and displaying data in the TCL Console).

The screenshot displays the Vivado 2023.2 IDE interface. The top menu bar includes File, Edit, Flow, Tools, Reports, Window, Layout, View, and Help. The main workspace is divided into several panes:

- Flow Navigator:** Shows the project hierarchy with sections for PROJECT MANAGER, IP INTEGRATOR, SIMULATION, RTL ANALYSIS, SYNTHESIS, and IMPLEMENTATION.
- PROJECT MANAGER - arraysum:** Displays the project structure, including Design Sources (1), Constraints, Simulation Sources (2), and Utility Sources.
- Sources:** Shows the list of sources, including design_1_wrapper (STRUCTURE) (design_1_wrapper.vhd) (1), Constraints, Simulation Sources (2), and Utility Sources.
- testbench.vhd:** Displays the VHDL code for the testbench, which includes UART communication logic and data display in the TCL Console.

The **Run Behavioral Simulation** menu is open, showing options for Run Behavioral Simulation, Run Post-Synthesis Functional Simulation, Run Post-Synthesis Timing Simulation, Run Post-Implementation Functional Simulation, and Run Post-Implementation Timing Simulation.

```
when "011" =>
    if counter = x"00000364" then -- middle of stop character
        -- report "received " & integer'image (to_integer (unsigned (data_received(9 downto 2)
        -- write (line1, character'val(to_integer(unsigned(data_received(9 downto 2))));
        -- writeline (output_file_stdout, line1);
        if (character_index = 19) then
            character_index (1) <= character'val(to_integer(unsigned(data_received(9 downto 2))));
            character_index <= 2;
            write (line1, character, left);
            writeline (output_file_stdout, line1);
        elsif (10 = to_integer(unsigned(data_received(9 downto 2)))) then
            character_index <= 1;
            write (line1, character, left);
            writeline (output_file_stdout, line1);
        else
            character(character_index) <= character'val(to_integer(unsigned(data_received(9
            character_index <= character_index + 1;
        end if;
        counter <= x"00000000";
        character_state <= "000";
    end if;
when others =>
    counter <= x"00000000";
    data_received <= "0000000000";
    number_of_bits_received <= x"00000000";
    character_state <= "000";
end case;
end if;
end if;
end process;
```

***Click : Run All (F3).
(Simulation launched).***

The screenshot displays the Xilinx Vivado IDE interface during a behavioral simulation. The top toolbar shows the 'Run All (F3)' button highlighted with a red box. The 'SIMULATION - Behavioral Simulation - Function testbench' window is active, showing the 'Scope' and 'Objects' panels. The 'Scope' panel lists the testbench hierarchy, and the 'Objects' panel lists the signals being monitored. The 'Tcl Console' at the bottom shows the simulation command and progress.

Scope Panel:

Name	Design Unit
testbench	testbench(my_testbench)
ut	design_1_wrapper(STRUCTURE)
gbl	gbl

Objects Panel:

Name	Value
clk	0
clk_n	1
rst_n	1
rst	0
rs232_uart_txd	1
rs232_uart_rxd	U
character_state[2:0]	0
counter[31:0]	00000000
number_of_bits_received[31:0]	00000000
data_received[9:0]	000
charbuffer[1:20]	"
charbuffer_index	1
output_file_stdout	

Tcl Console:

```
Copyright 2022-2023 Advanced Micro Devices, Inc. All Rights Reserved.  
Running: C:/Xilinx/Vivado/2023.2/bin/unwrapped/win64.o/xelab.exe --incr --debug typical --relax --mt 2 -L xil_defaultlib -L microblaze_v11_0_12 -L lmb_v10_v3_0_13 -L lmb_bram  
Using 2 slave threads.  
Starting static elaboration  
Pass Through NonSizing Optimizer
```

Run the simulation until there are no more events or until a Verilog '\$finish' or '\$stop'

Sim Time: 1 us

Click : TCL Console to see the result.

It may take some time;
please be patient.

(Sum = 45.000000).

OR Click: charbuffer[1:20] Signal in the waveform viewer.

The screenshot displays the Vivado IDE interface during a simulation. The left sidebar contains the Project Manager, IP Integrator, Simulation, RTL Analysis, Synthesis, and Implementation sections. The Simulation section is active, showing the 'Run Simulation' button. The main window is divided into two panes. The top pane, titled 'SIMULATION - Behavioral Simulation - Functional - sim_1 - testbench', shows a list of signals and their current values. The bottom pane, titled 'Tcl Console', shows the output of the simulation, including the sum of the data received.

Name	Value
clk	1
clk_n	0
rst_n	1
rst	0
rs232_uart_txd	1
rs232_uart_rxd	U
character_state[2:0]	2
counter[31:0]	0000008a
number_of_bits_received[31:0]	00000001
data_received[9:0]	214
charbuffer[1:20]	"Sum = 45.000000"
charbuffer_index	1

The Tcl Console output shows the following messages:

```
Block Memory Generator module testbench.uut.design_1_i.axi_bram_ctrl_0_bram.inst.\native_mem_mapped_module.blk_mem_gen_v8_4_7_inst is using a behavioral model for simulation
Block Memory Generator module testbench.uut.design_1_i.microblaze_0_local_memory.lmb_bram.inst.\native_mem_mapped_module.blk_mem_gen_v8_4_7_inst is using a behavioral
relaunch_sim: Time (s): cpu = 00:00:05 ; elapsed = 00:00:24 . Memory (MB): peak = 1695.969 ; gain = 0.000
run all
Sum = 45.000000
```

The bottom status bar indicates the simulation time: Sim Time: 1786145 ns.

Using TinyAES In SDK for Data Encryption | Decryption.

The screenshot displays the Vitis Unified IDE 2023.2 interface. The left sidebar shows the project structure for 'array_sum_sdk'. The main editor window shows the C source code for 'helloworld.c', which implements AES encryption and decryption using the TinyAES library. The code includes headers for Xilinx IP and TinyAES, defines a union for data storage, and uses Xilinx APIs for memory access and printing. The bottom panel shows the build output, indicating a successful build for the 'array_sum_sdk' target.

```
array_sum_sdk > src > C helloworld.c > main
31     itof.b = arr[i];
32     Xil_Out32(XPAR_AXI_BRAM_0_BASEADDRESS + (4 * i), itof.a); // Writing to BRAM.
33     sum += arr[i]; //Calculates Sum.
34 }
35
36 itof.b = sum; //Saving : Result : In : UNION
37 Xil_Out32(XPAR_AXI_BRAM_0_BASEADDRESS + (4 * 9), itof.a); //Write The sum to BRAM.
38
39
40 //PRINTING ORIGINAL : SUM RESULT :
41 sprintf(buff, "%f", itof.b); //CONVERT FLOAT of ITOF.B TO a STRING and store it in BUFF
42 xil_printf("Sum= %s\n\r", buff); //Print The formatted STRING stored in BUFF to the UART
43
44 //AES ENCRYPTION : SUM RESULT :
45 struct AES_ctx ctx; //AES : context
46 AES_init_ctx(&ctx, key); //Initialize AES context with the key
47 memcpy(data, &itof.b, sizeof(itof.b)); //Copy the float sum into the data buffer
48 AES_ECB_encrypt(&ctx, data); //Encrypt the data buffer
49 xil_printf("EncSum= "); //Print the encrypted SUM. (IN HEXDECIMAL)
50 for(int i=0; i<16; i++){
51     xil_printf("%02x ", data[i]); //Print each byte of the encrypted data
52 }
53 xil_printf("\n\r");
54
55 //AES DECRYPTION : SUM RESULT :
56 AES_ECB_decrypt(&ctx, data); //Decrypt the data buffer
57 memcpy(&itof.b, data, sizeof(itof.b)); //Copy the decrypted data back into the union (float)
58 sprintf(buff, "%f", itof.b); //CONVERT FLOAT of ITOF.B TO a STRING and store it in BUFF
59 xil_printf("DecSum= %s\n\r", buff); //Print the decrypted SUM.
60
61 cleanup_platform();
62 return 0;
63 }
```

OUTPUT

```
Vitis Server array_sum_sdk: X
make[2]: Leaving directory 'c:/Users/znkr/Downloads/arraysum/array_sum_sdk/build'
make -f CMakeFiles/array_sum_sdk.elf.dir/build.make CMakeFiles/array_sum_sdk.elf.dir/build
make[2]: Entering directory 'c:/Users/znkr/Downloads/arraysum/array_sum_sdk/build'
[ 25%] Building C object CMakeFiles/array_sum_sdk.elf.dir/helloworld.c.obj
C:/Xilinx/Vitis/2023.2/gnu/microblaze/nt/bin/mb-gcc.exe -isystem C:/Users/znkr/Downloads/arraysum/platform/export/platform/sw/standalone_micro
[ 50%] Linking C executable array_sum_sdk.elf
C:/Xilinx/Vitis/2023.2/gnu/microblaze/nt/bin/mb-gcc.exe -O2 -mlittle-endian -mxl-soft-mul -mcpu=v11.0 -DSDT -MMD -MP -specs=C:/Users/znkr/_Dow
mb-size --format=berkeley array_sum_sdk.elf
text      data      bss      dec      hex filename
103732     2104     2148     107984    1a5d0 array_sum_sdk.elf
mb-size --format=berkeley array_sum_sdk.elf > C:/Users/znkr/Downloads/arraysum/array_sum_sdk/build/array_sum_sdk.elf.size
make[2]: Leaving directory 'c:/Users/znkr/Downloads/arraysum/array_sum_sdk/build'
[100%] Built target array_sum_sdk.elf
make[1]: Leaving directory 'c:/Users/znkr/Downloads/arraysum/array_sum_sdk/build'
C:/Xilinx/Vitis/2023.2/tps/win64/cmake-3.24.2/bin/cmake.exe -E cmake_progress_start C:/Users/znkr/Downloads/arraysum/array_sum_sdk/build/CMakeF
Build Finished successfully
[1/6/2025, 6:27:45 PM]: Build for array_sum_sdk:: with id 'e7dbec4c-b83c-49dc-85d2-33744a77fda6' ended.
```

array_sum_sdk

- Vitis Server
- clangd
- Git
- GitLens
- HLS Pragma Language Server

Component: array_sum_sdk

Build

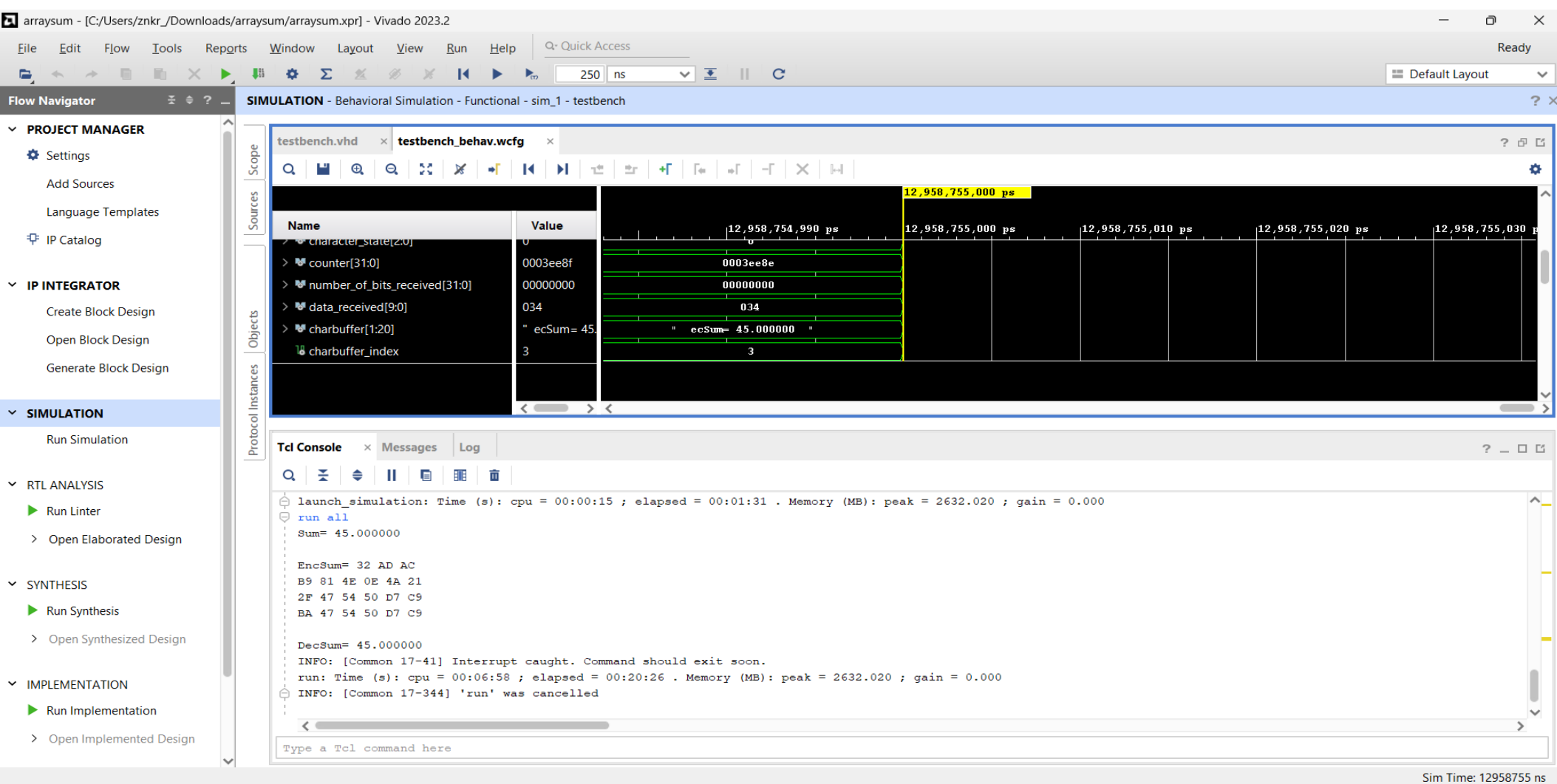
Run

Debug

clangd: idle

Ln 63, Col 2 LF UTF-8 Spaces: 4

Simulation Results: Original Sum | Encrypted Sum | Decrypted Sum



Thank you for your participation.