Vitis_HLS Getting Start

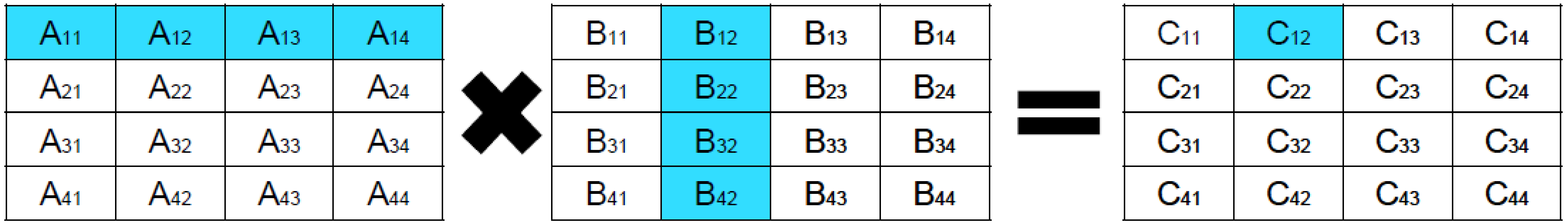
AMD

together we advance_

Matrix Operations

Matrix Operations

C[4][4] = A[4][4] * B[4][4]



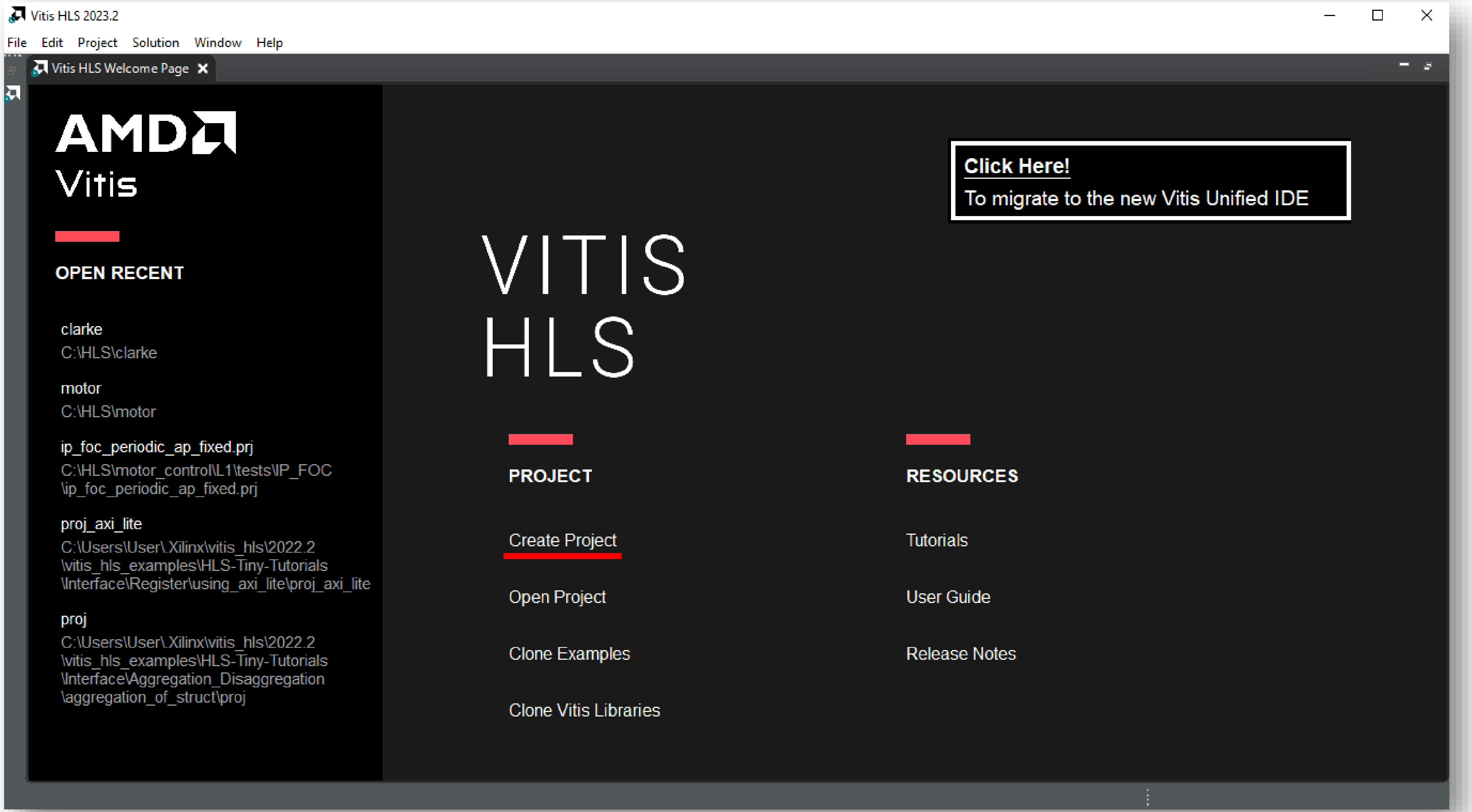
C11 = A11 * B11 + A12 * B21 + A13 * B31 + A41 * B41

C12 = A11 * B12 + A12 * B22 + A13 * B32 + A41 * B42

C13 = A11 * B13 + A12 * B23 + A13 * B33 + A41 * B43

C14 = A11 * B14 + A12 * B24 + A13 * B34 + A41 * B44

Vitis_HLS Design Flow



OPEN RECENT

clarke
C:\HLS\clarke

motor
C:\HLS\motor

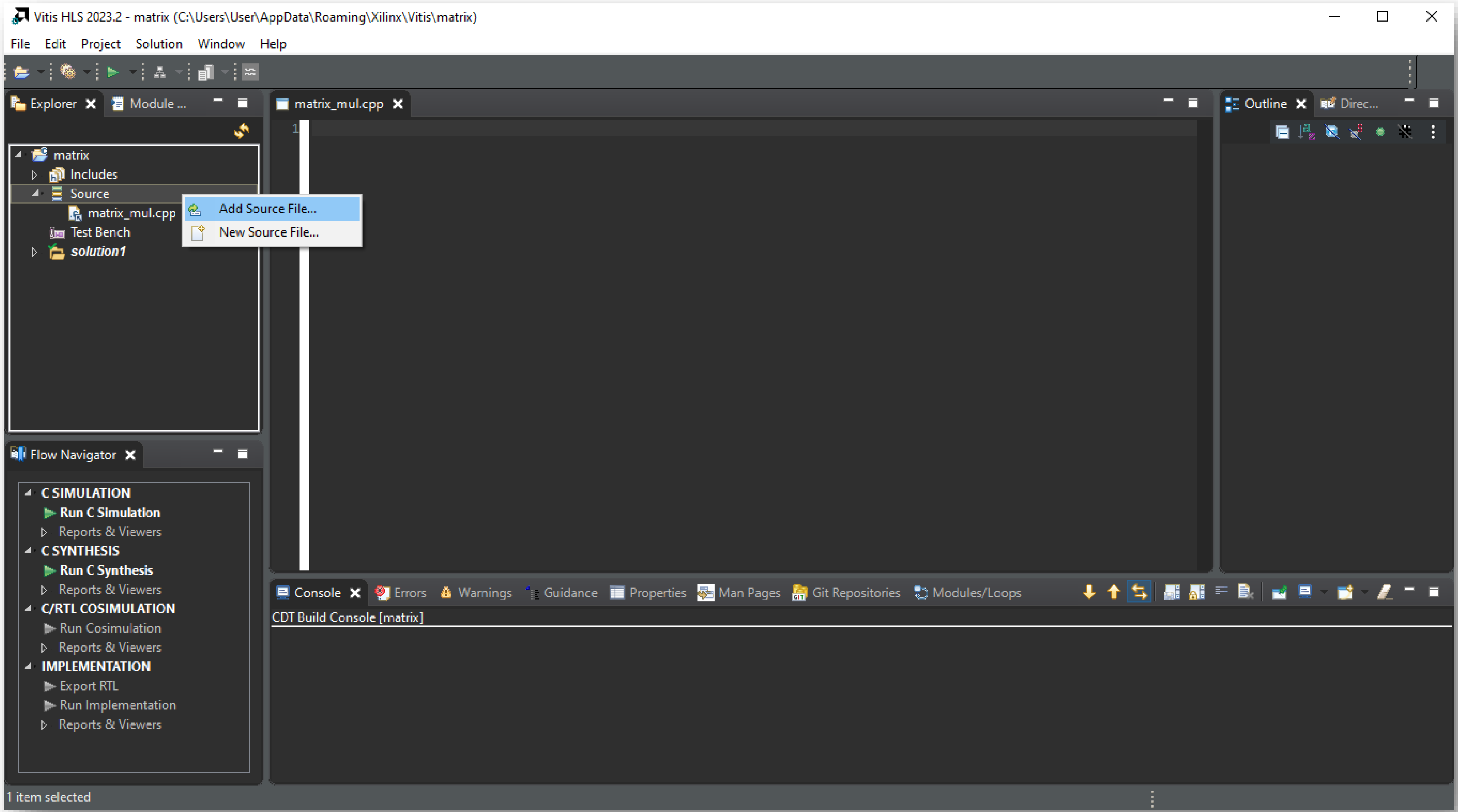
ip_foc_periodic_ap_fixed.prj
C:\HLS\motor_control\L1\tests\IP_FOC\ip_foc_periodic_ap_fixed.prj

proj_axi_lite
C:\Users\User\.Xilinx\vitis_hls\2022.2\vitis_hls_examples\HLS-Tiny-Tutorials\Interface\Register\using_axi_lite\proj_axi_lite

proj
C:\Users\User\.Xilinx\vitis_hls\2022.2\vitis_hls_examples\HLS-Tiny-Tutorials\Interface\Aggregation_Disaggregation\aggregation_of_struct\proj

[Public]

Add Source File



AMD
together we advance_

matrix_mul.h

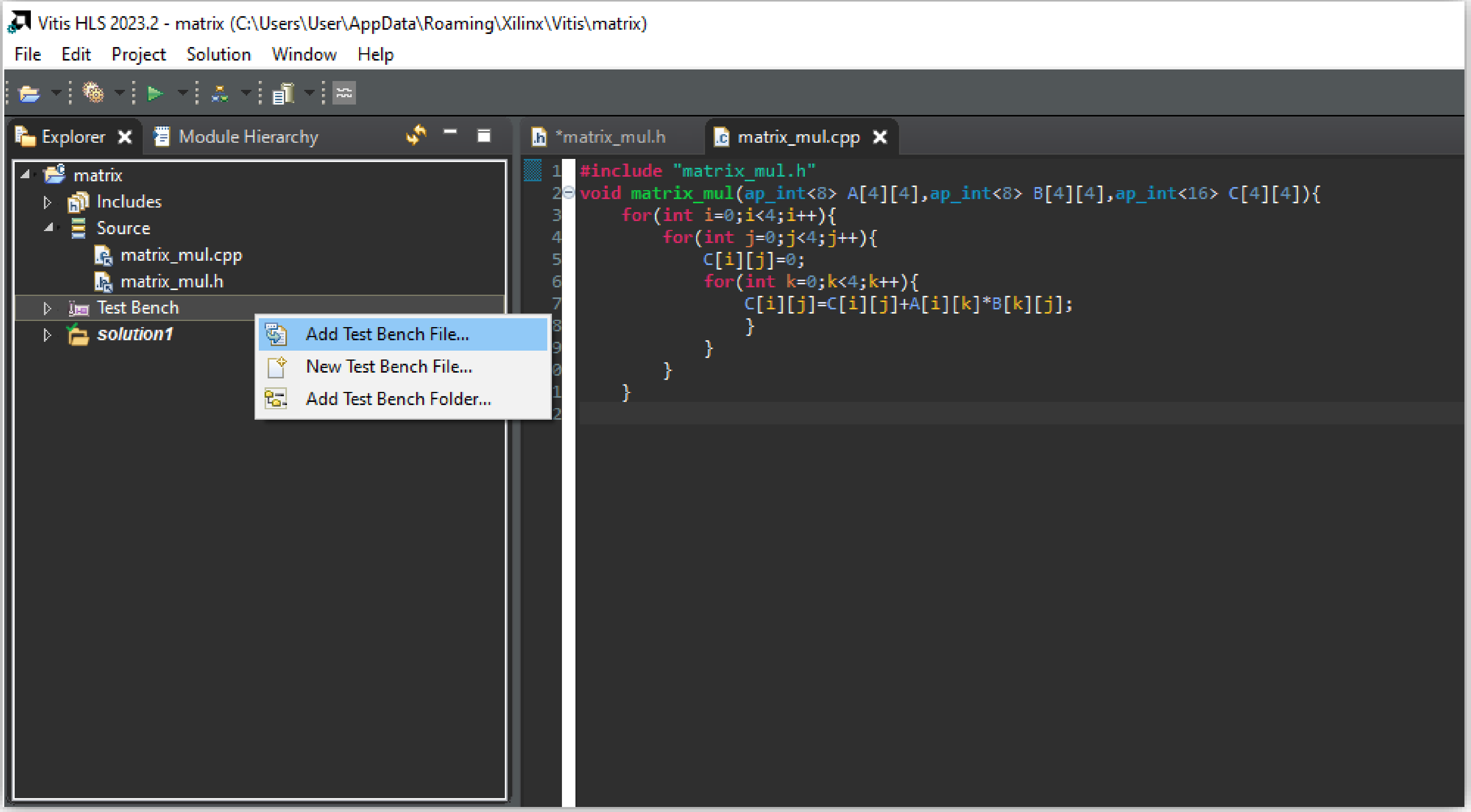
```
#ifndef __MATRIX_MUL__
#define __MATRIX_MUL__

//Custom precision
#include "ap_fixed.h"

//Matrix Declaration
void matrix_mul(ap_int<8> A[4][4],ap_int<8> B[4][4],ap_int<16> C[4][4]);
//ap_type<n bits> matrix_name[row][column]

#endif
```

matrix_mul.cpp

```cpp
#include "matrix_mul.h"
void matrix_mul(ap_int<8> A[4][4],ap_int<8> B[4][4],ap_int<16> C[4][4]){
        for(int i=0;i<4;i++){
                for(int j=0;j<4;j++){
                        C[i][j]=0;
                        for(int k=0;k<4;k++){
                                C[i][j]=C[i][j]+A[i][k]*B[k][j];
                        }
                }
        }
}
```

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix} \times \begin{bmatrix} B_{11} & B_{12} & B_{13} & B_{14} \\ B_{21} & B_{22} & B_{23} & B_{24} \\ B_{31} & B_{32} & B_{33} & B_{34} \\ B_{41} & B_{42} & B_{43} & B_{44} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & C_{34} \\ C_{41} & C_{42} & C_{43} & C_{44} \end{bmatrix}$$

Add Test Bench

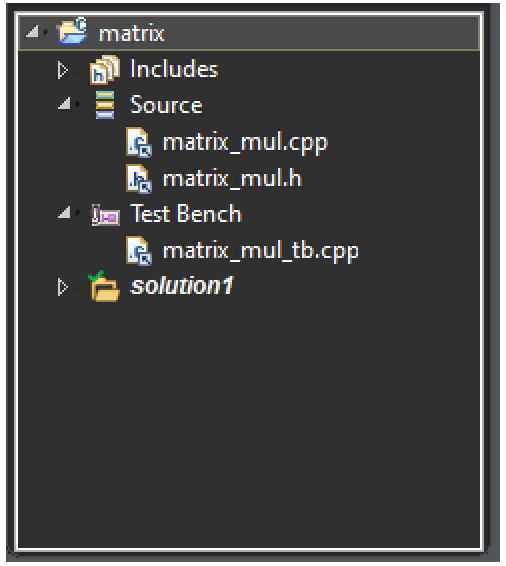


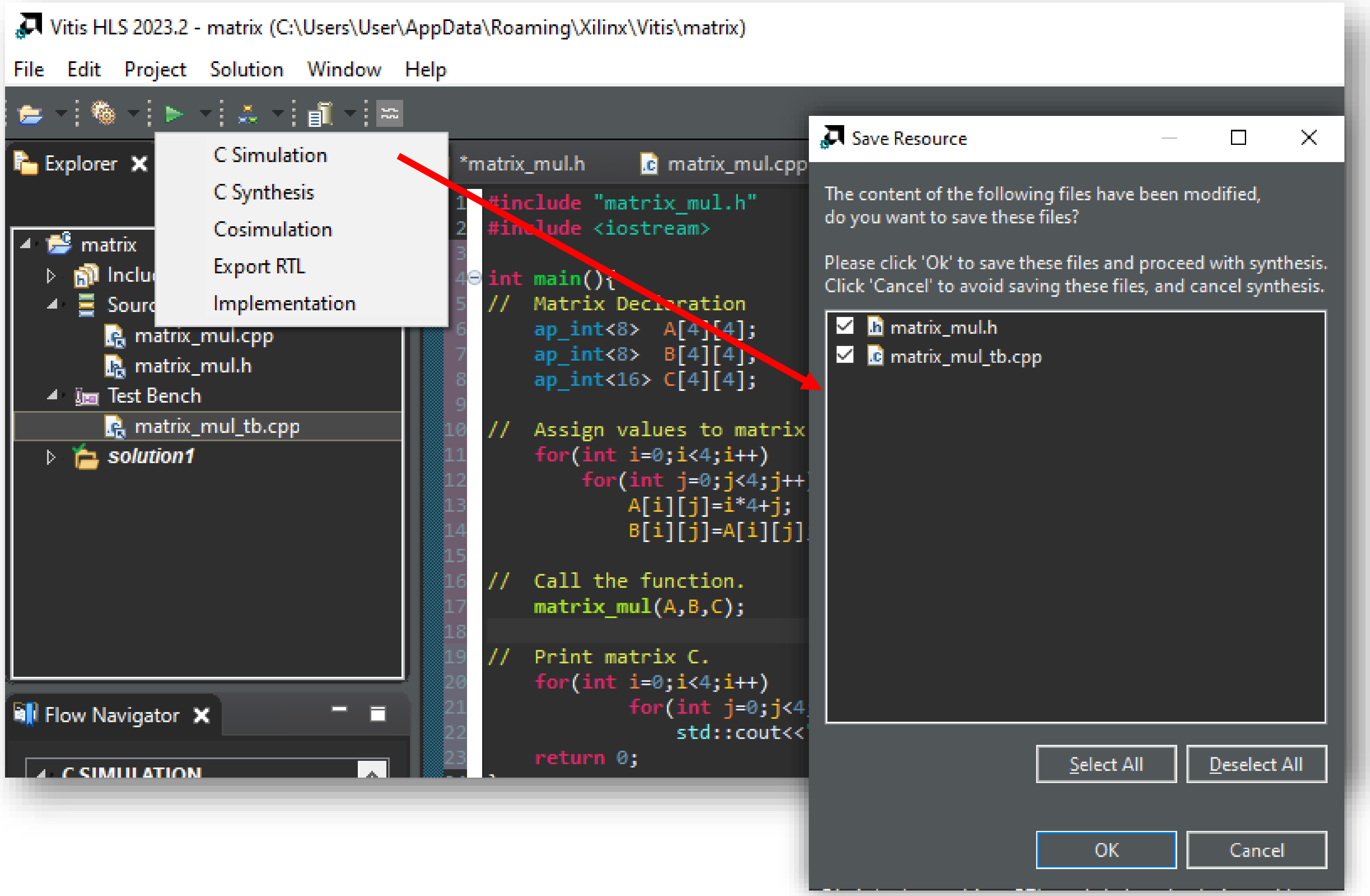
```
#include "matrix_mul.h"
void matrix_mul(ap_int<8> A[4][4],ap_int<8> B[4][4],ap_int<16> C[4][4]){
    for(int i=0;i<4;i++){
        for(int j=0;j<4;j++){
            C[i][j]=0;
            for(int k=0;k<4;k++){
                C[i][j]=C[i][j]+A[i][k]*B[k][j];
            }
        }
    }
}
```

matrix_mul_tb.cpp

```
#include "matrix_mul.h"
#include <iostream>
int main(){
//Matrix Declaration
       ap_int<8>  A[4][4];
       ap_int<8>  B[4][4];
       ap_int<16> C[4][4];
//Assign values to matrix A and matrix B
       for(int i=0;i<4;i++)
              for(int j=0;j<4;j++){
                     A[i][j]=i*4+j;
                     B[i][j]=A[i][j];}
//Call the function.
       matrix_mul(A,B,C);
//Print matrix C.
       for(int i=0;i<4;i++)
              for(int j=0;j<4;j++)
                     std::cout<<"C["<<i<<","<<j<<"]="<<C[i][j]<<std::endl;
return 0;
}
```

Run Flow (C Simulation)




Vitis HLS 2023.2 - matrix (C:\Users\User\AppData\Roaming\Xilinx\Vitis\matrix)

```
#include "matrix_mul.h"
#include <iostream>

int main(){
//  Matrix Declaration
    ap_int<8>  A[4][4];
    ap_int<8>  B[4][4];
    ap_int<16> C[4][4];

//  Assign values to matrix
    for(int i=0;i<4;i++)
        for(int j=0;j<4;j++
            A[i][j]=i*4+j;
            B[i][j]=A[i][j]

//  Call the function.
    matrix_mul(A,B,C);

//  Print matrix C.
    for(int i=0;i<4;i++)
            for(int j=0;j<4
                std::cout<<
    return 0;
```

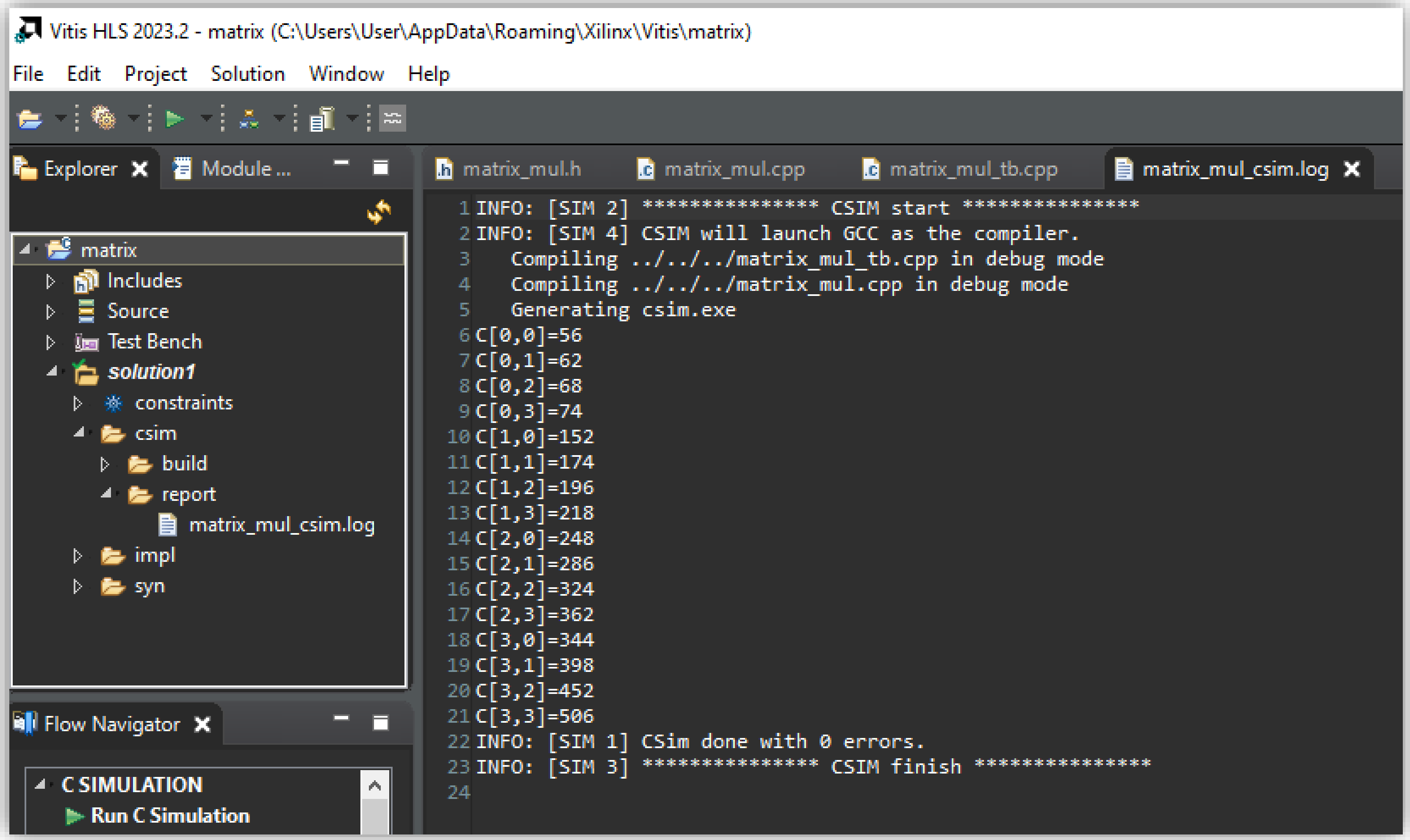
Save Resource

The content of the following files have been modified, do you want to save these files?

Please click 'Ok' to save these files and proceed with synthesis.
Click 'Cancel' to avoid saving these files, and cancel synthesis.

- ✓ matrix_mul.h
- ✓ matrix_mul_tb.cpp

C Simulation Result

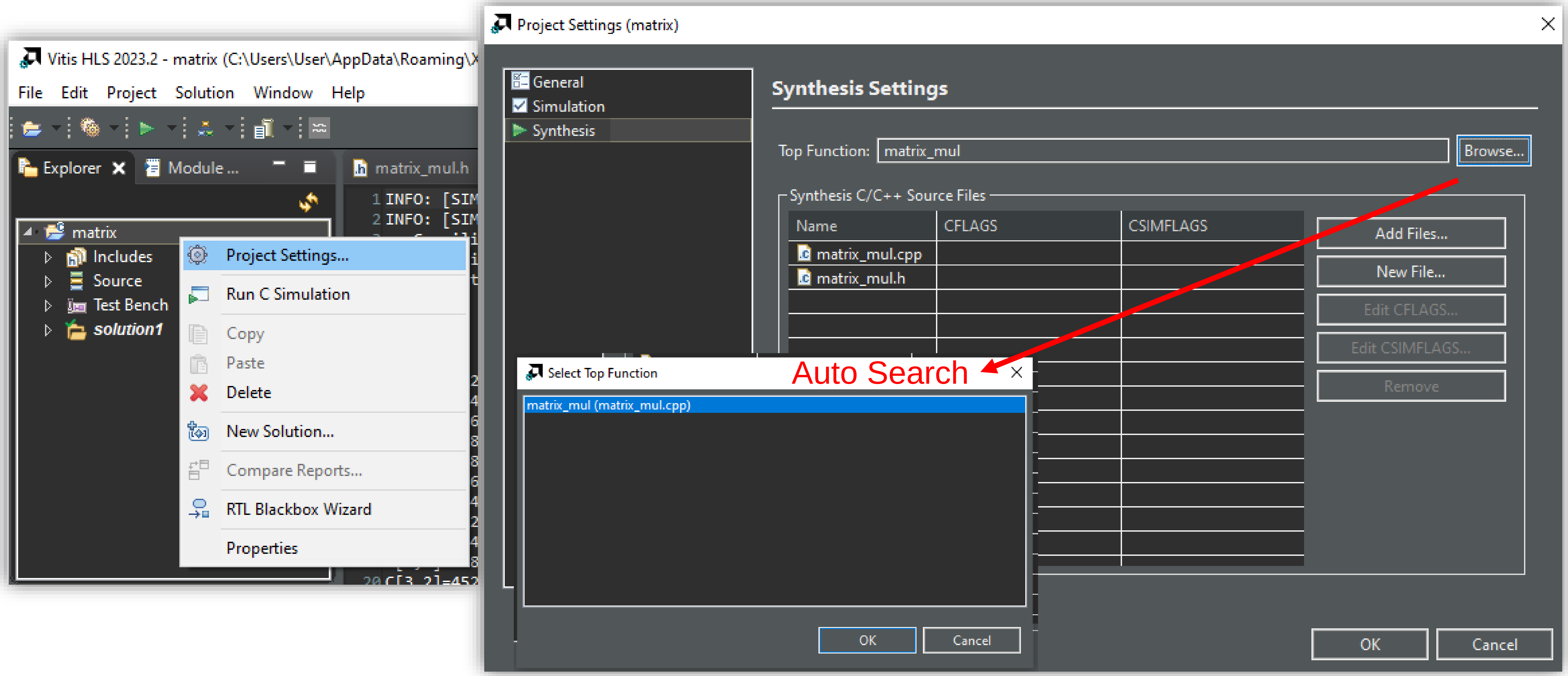


```
INFO: [SIM 2] *************** CSIM start ***************
INFO: [SIM 4] CSIM will launch GCC as the compiler.
   Compiling ../../../matrix_mul_tb.cpp in debug mode
   Compiling ../../../matrix_mul.cpp in debug mode
   Generating csim.exe
C[0,0]=56
C[0,1]=62
C[0,2]=68
C[0,3]=74
C[1,0]=152
C[1,1]=174
C[1,2]=196
C[1,3]=218
C[2,0]=248
C[2,1]=286
C[2,2]=324
C[2,3]=362
C[3,0]=344
C[3,1]=398
C[3,2]=452
C[3,3]=506
INFO: [SIM 1] CSim done with 0 errors.
INFO: [SIM 3] *************** CSIM finish ***************
```

AMD
together we advance_

Run Flow (C Synthesis)

Convert the C code to RTL code.



```
1 INFO: [SIM
2 INFO: [SIM
```

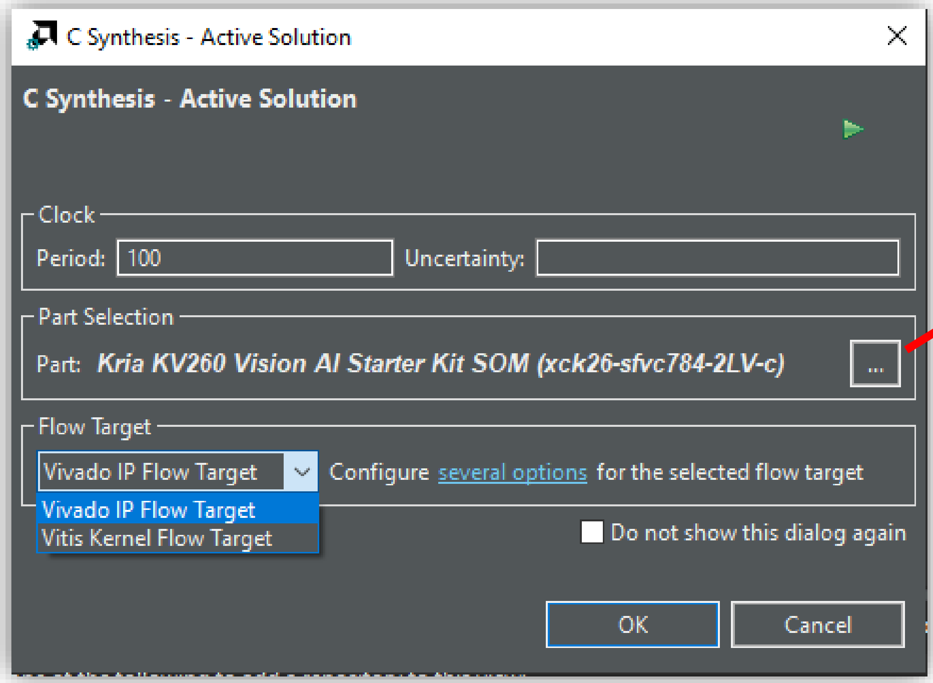
Top Function: matrix_mul

Name	CFLAGS	CSIMFLAGS
matrix_mul.cpp		
matrix_mul.h		

matrix_mul (matrix_mul.cpp)

[Public]

Run Flow (C Simulation)



C Synthesis - Active Solution

C Synthesis - Active Solution

Clock

Period: 100 Uncertainty:

Part Selection

Part: *Kria KV260 Vision AI Starter Kit SOM (xck26-sfvc784-2LV-c)*

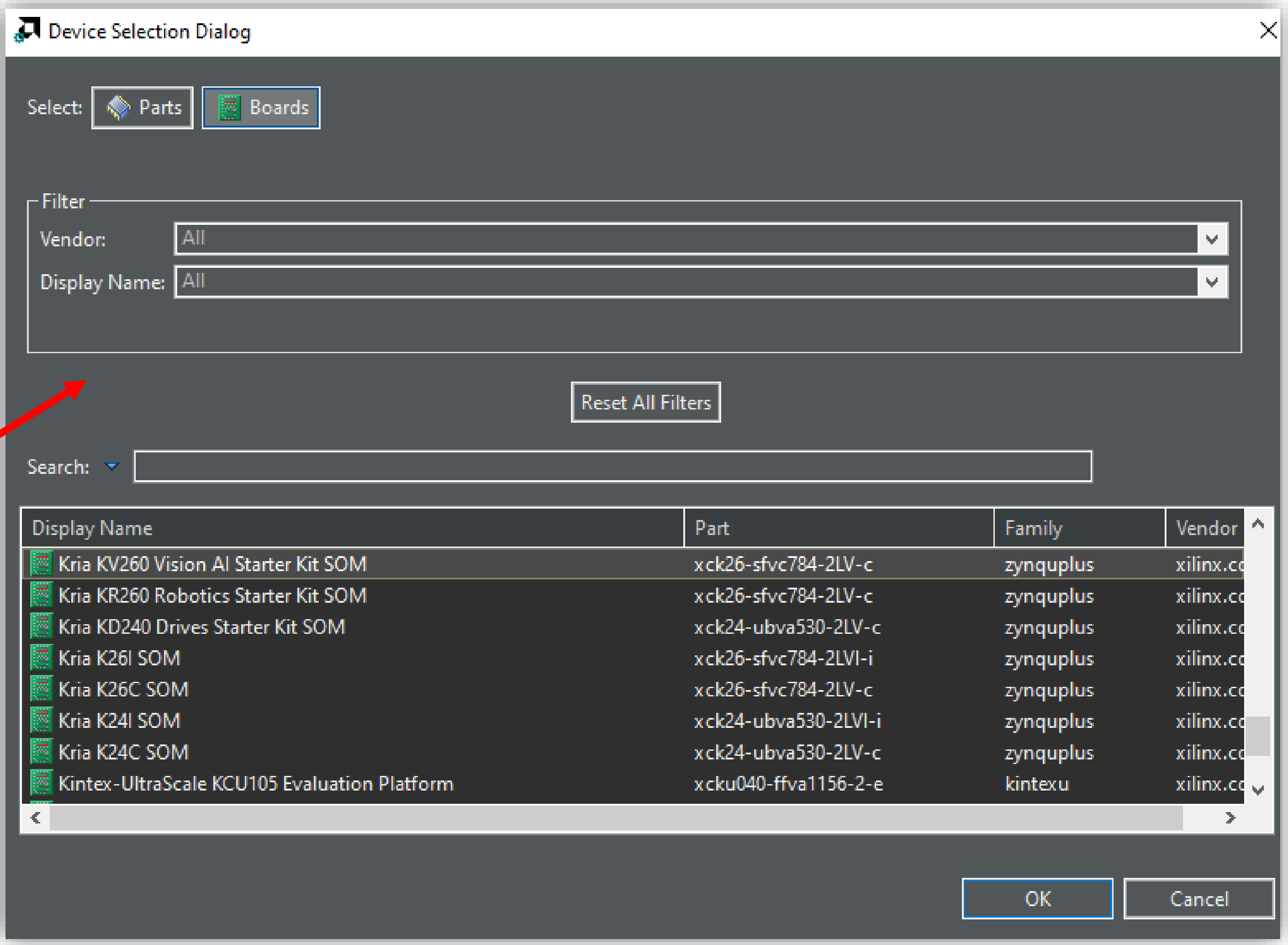
Flow Target

Vivado IP Flow Target — Configure several options for the selected flow target

- Vivado IP Flow Target
- Vitis Kernel Flow Target

☐ Do not show this dialog again

OK Cancel



Device Selection Dialog

Select: Parts Boards

Filter

Vendor: All

Display Name: All

Reset All Filters

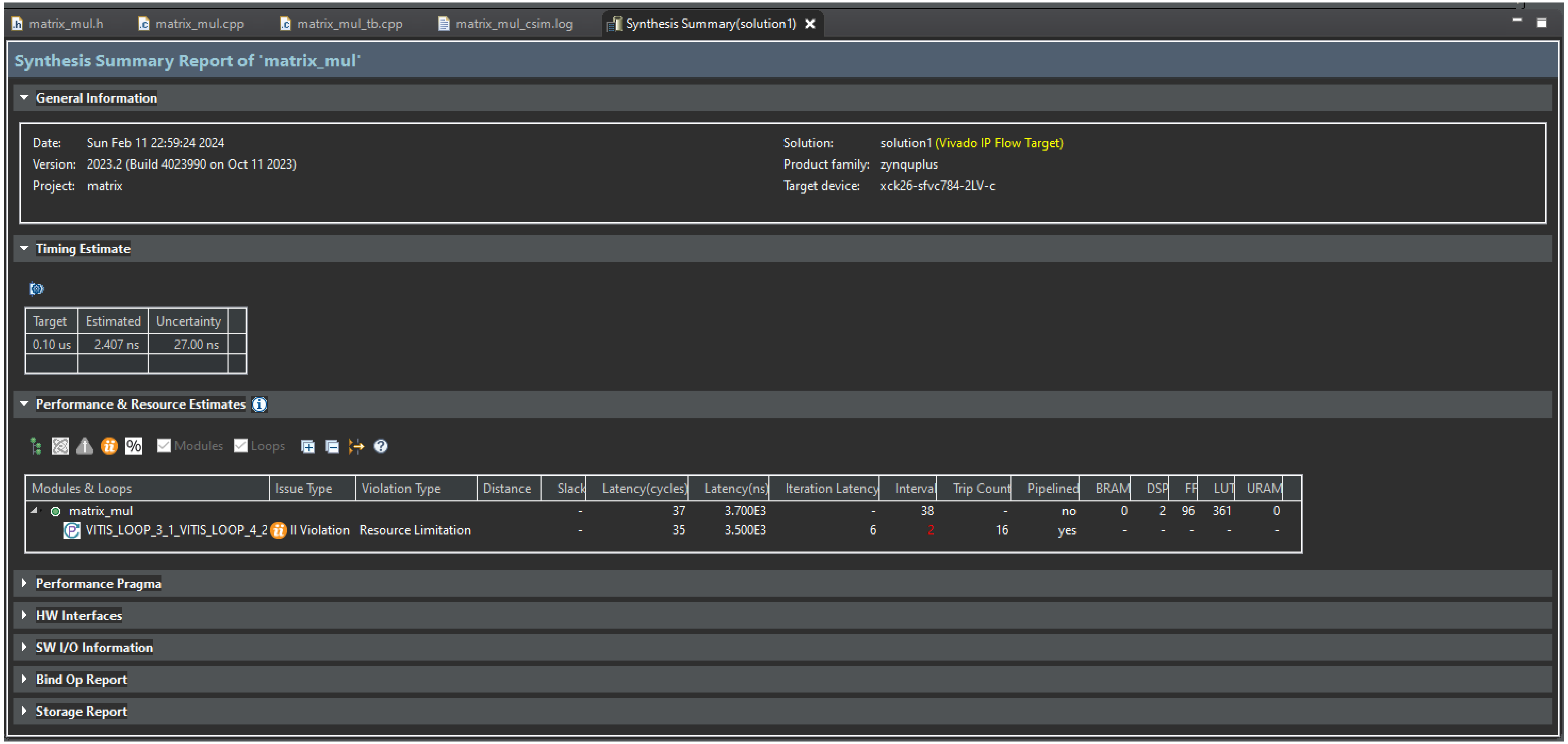
Search:

Display Name	Part	Family	Vendor
Kria KV260 Vision AI Starter Kit SOM	xck26-sfvc784-2LV-c	zynquplus	xilinx.c
Kria KR260 Robotics Starter Kit SOM	xck26-sfvc784-2LV-c	zynquplus	xilinx.c
Kria KD240 Drives Starter Kit SOM	xck24-ubva530-2LV-c	zynquplus	xilinx.c
Kria K26I SOM	xck26-sfvc784-2LVI-i	zynquplus	xilinx.c
Kria K26C SOM	xck26-sfvc784-2LV-c	zynquplus	xilinx.c
Kria K24I SOM	xck24-ubva530-2LVI-i	zynquplus	xilinx.c
Kria K24C SOM	xck24-ubva530-2LV-c	zynquplus	xilinx.c
Kintex-UltraScale KCU105 Evaluation Platform	xcku040-ffva1156-2-e	kintexu	xilinx.c

OK Cancel

AMD
together we advance_

Synthesis Summary Report



matrix_mul.h | matrix_mul.cpp | matrix_mul_tb.cpp | matrix_mul_csim.log | Synthesis Summary(solution1)

Synthesis Summary Report of 'matrix_mul'

General Information

Date:	Sun Feb 11 22:59:24 2024	Solution:	solution1 (Vivado IP Flow Target)
Version:	2023.2 (Build 4023990 on Oct 11 2023)	Product family:	zynquplus
Project:	matrix	Target device:	xck26-sfvc784-2LV-c

Timing Estimate

Target	Estimated	Uncertainty
0.10 us	2.407 ns	27.00 ns

Performance & Resource Estimates

Modules Loops

Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
matrix_mul				-	37	3.700E3	-	38	-	no	0	2	96	361	0
VITIS_LOOP_3_1_VITIS_LOOP_4_2	II Violation	Resource Limitation		-	35	3.500E3	6	2	16	yes	-	-	-	-	-

Performance Pragma

HW Interfaces

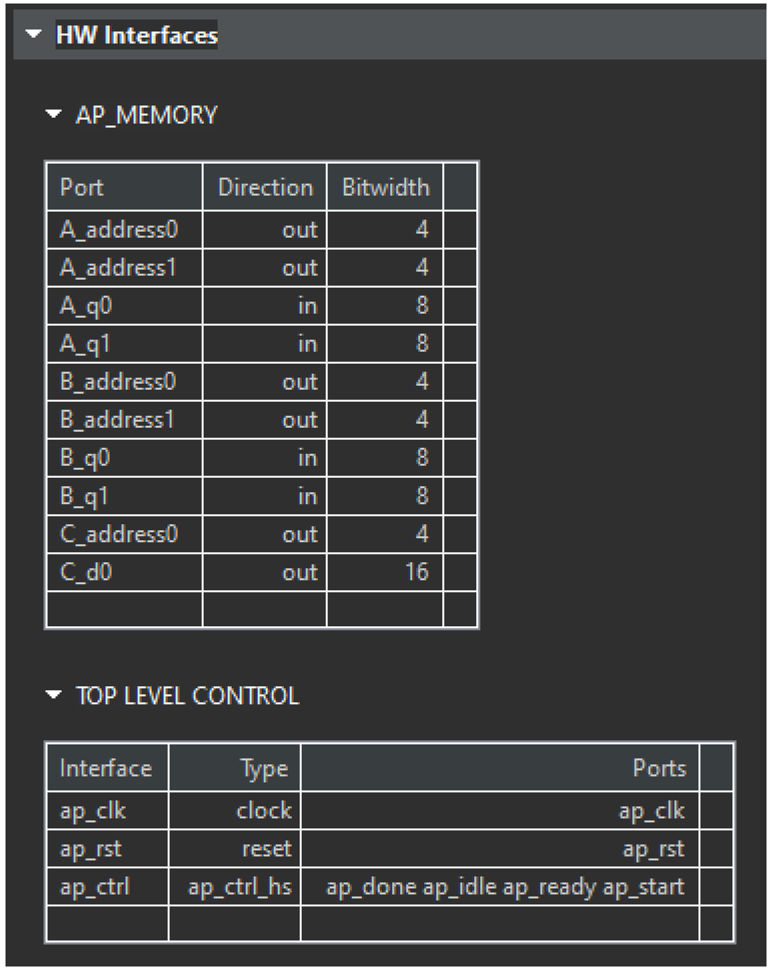
SW I/O Information

Bind Op Report

Storage Report

AMD
together we advance_

Synthesis Summary Report



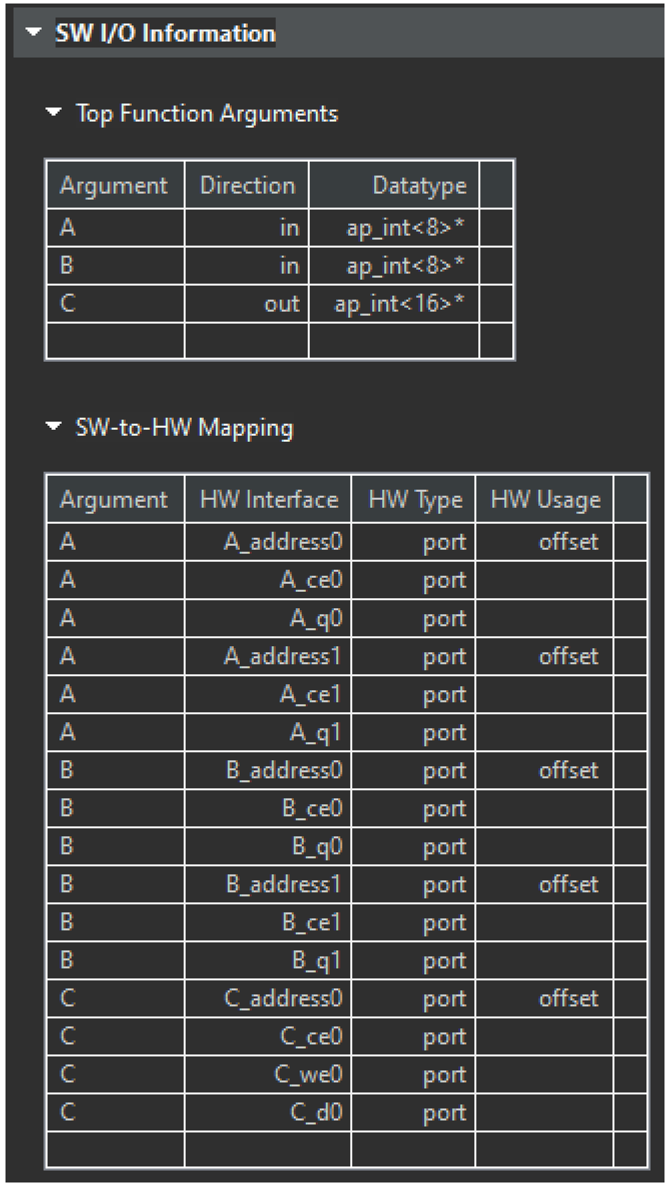
HW Interfaces

AP_MEMORY

Port	Direction	Bitwidth
A_address0	out	4
A_address1	out	4
A_q0	in	8
A_q1	in	8
B_address0	out	4
B_address1	out	4
B_q0	in	8
B_q1	in	8
C_address0	out	4
C_d0	out	16

TOP LEVEL CONTROL

Interface	Type	Ports
ap_clk	clock	ap_clk
ap_rst	reset	ap_rst
ap_ctrl	ap_ctrl_hs	ap_done ap_idle ap_ready ap_start



SW I/O Information

Top Function Arguments

Argument	Direction	Datatype
A	in	ap_int<8>*
B	in	ap_int<8>*
C	out	ap_int<16>*

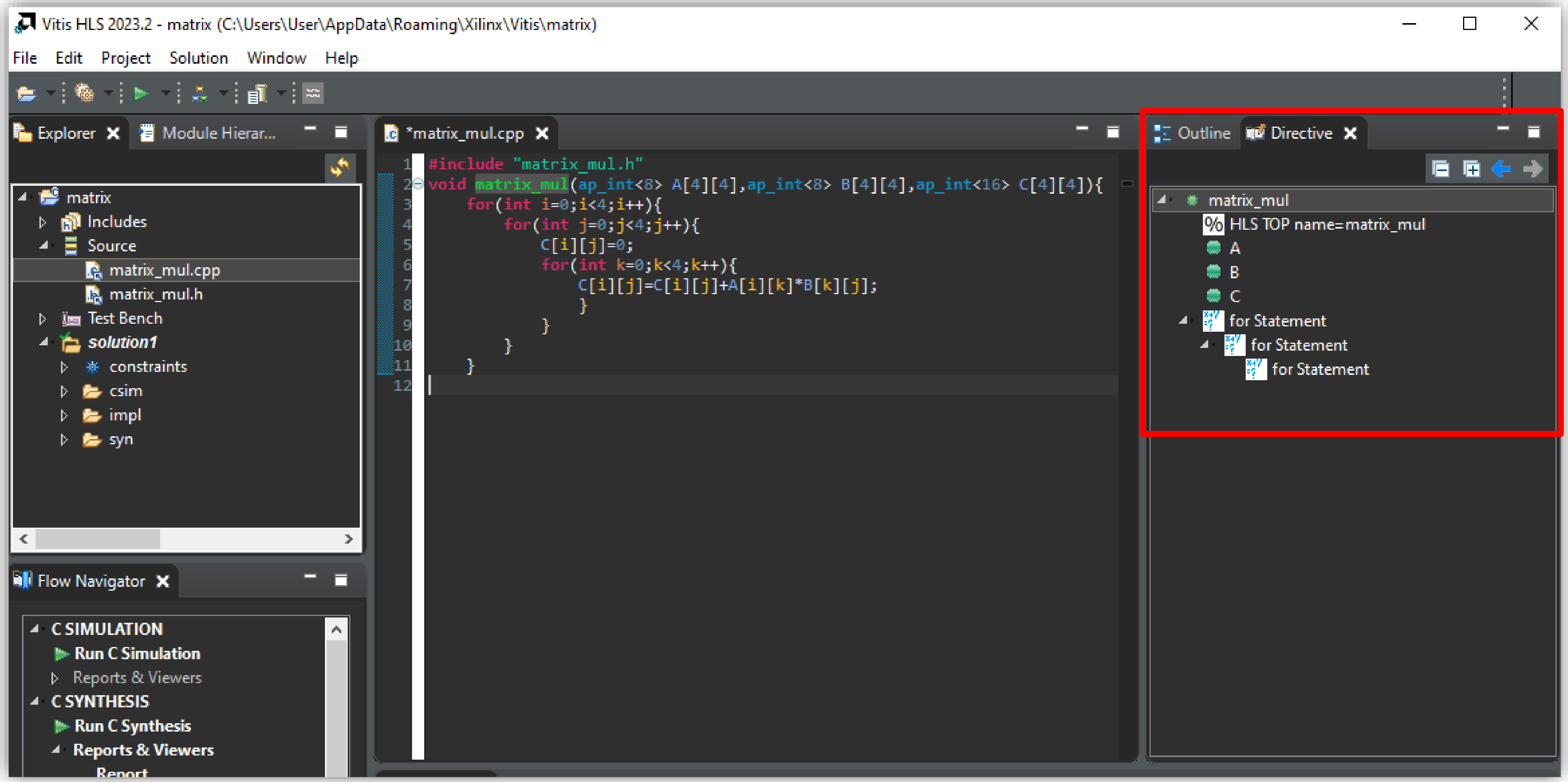
SW-to-HW Mapping

Argument	HW Interface	HW Type	HW Usage
A	A_address0	port	offset
A	A_ce0	port	
A	A_q0	port	
A	A_address1	port	offset
A	A_ce1	port	
A	A_q1	port	
B	B_address0	port	offset
B	B_ce0	port	
B	B_q0	port	
B	B_address1	port	offset
B	B_ce1	port	
B	B_q1	port	
C	C_address0	port	offset
C	C_ce0	port	
C	C_we0	port	
C	C_d0	port	

AMD
together we advance_

HLS Optimization

HLS will automatically optimize based on the set directives.

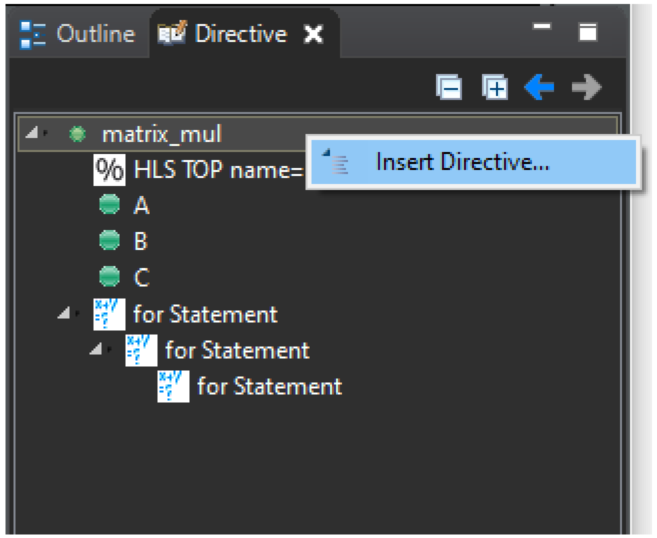


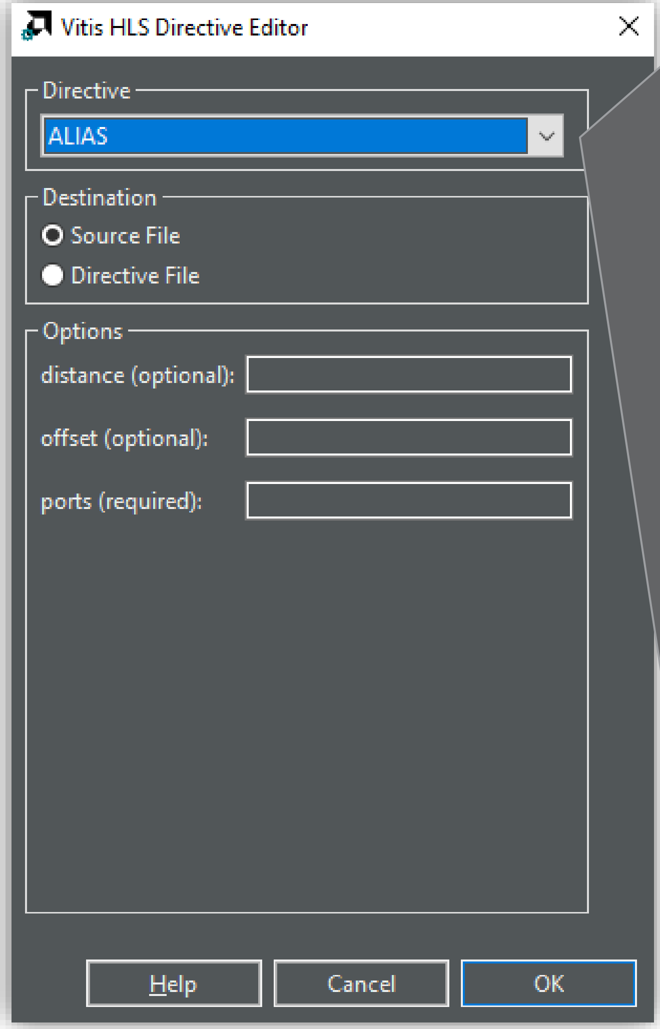
Vitis HLS 2023.2 - matrix (C:\Users\User\AppData\Roaming\Xilinx\Vitis\matrix)

```
#include "matrix_mul.h"
void matrix_mul(ap_int<8> A[4][4],ap_int<8> B[4][4],ap_int<16> C[4][4]){
    for(int i=0;i<4;i++){
        for(int j=0;j<4;j++){
            C[i][j]=0;
            for(int k=0;k<4;k++){
                C[i][j]=C[i][j]+A[i][k]*B[k][j];
            }
        }
    }
}
```

```
matrix_mul
    HLS TOP name=matrix_mul
    A
    B
    C
    for Statement
        for Statement
            for Statement
```

Insert Directive

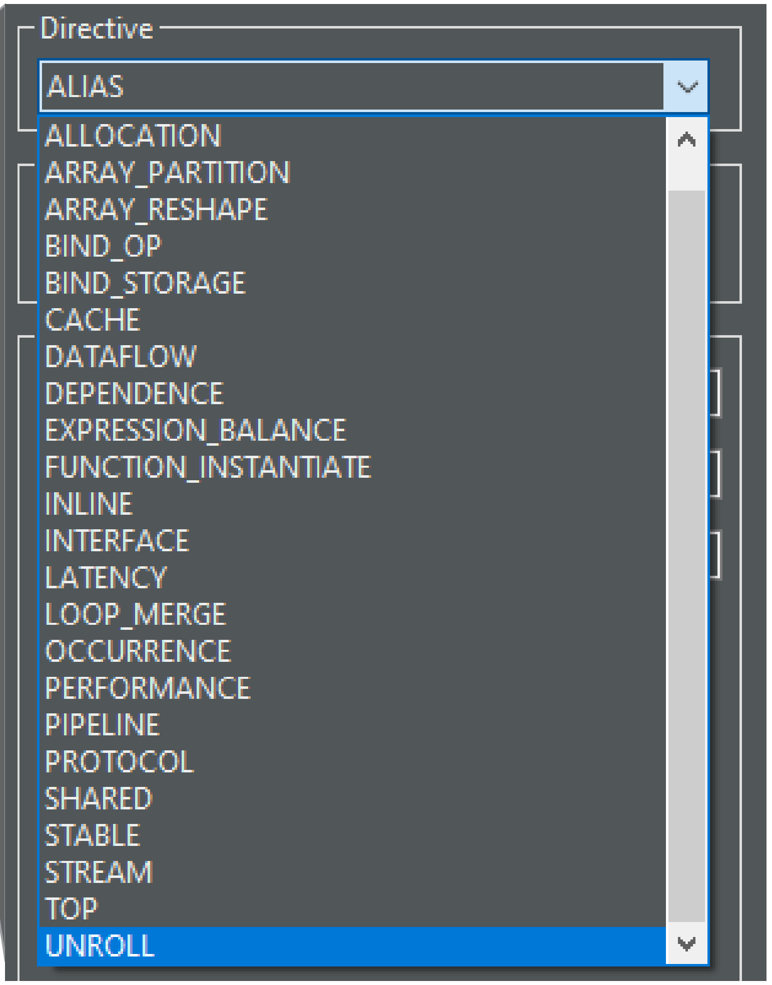



Vitis HLS Directive Editor

Directive: ALIAS

Destination
Source File
Directive File

Options
distance (optional):
offset (optional):
ports (required):

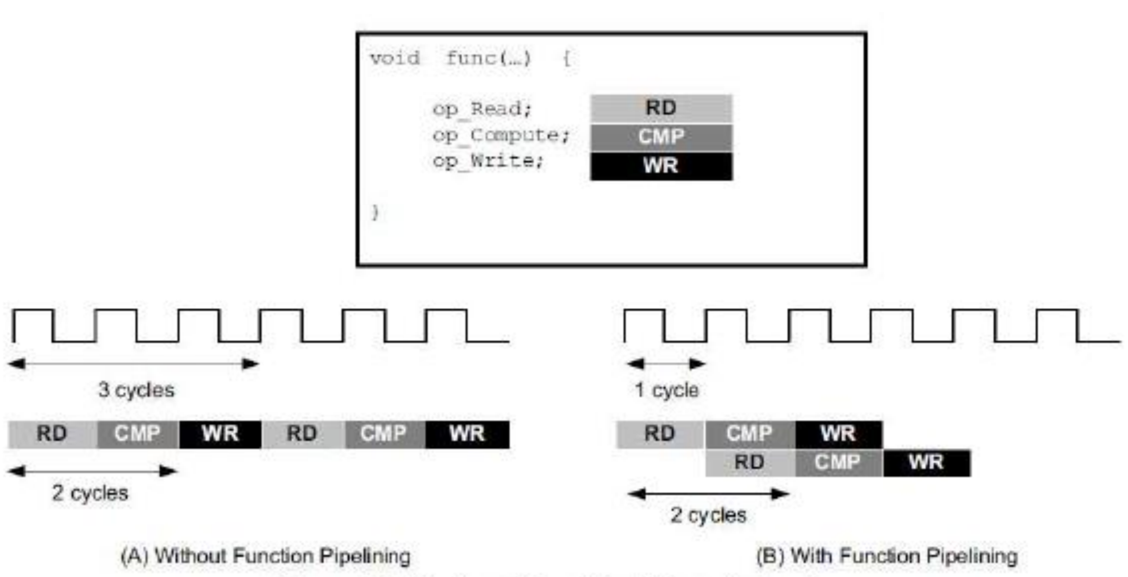

Directive: ALIAS

ALLOCATION
ARRAY_PARTITION
ARRAY_RESHAPE
BIND_OP
BIND_STORAGE
CACHE
DATAFLOW
DEPENDENCE
EXPRESSION_BALANCE
FUNCTION_INSTANTIATE
INLINE
INTERFACE
LATENCY
LOOP_MERGE
OCCURRENCE
PERFORMANCE
PIPELINE
PROTOCOL
SHARED
STABLE
STREAM
TOP
UNROLL

Pipelining

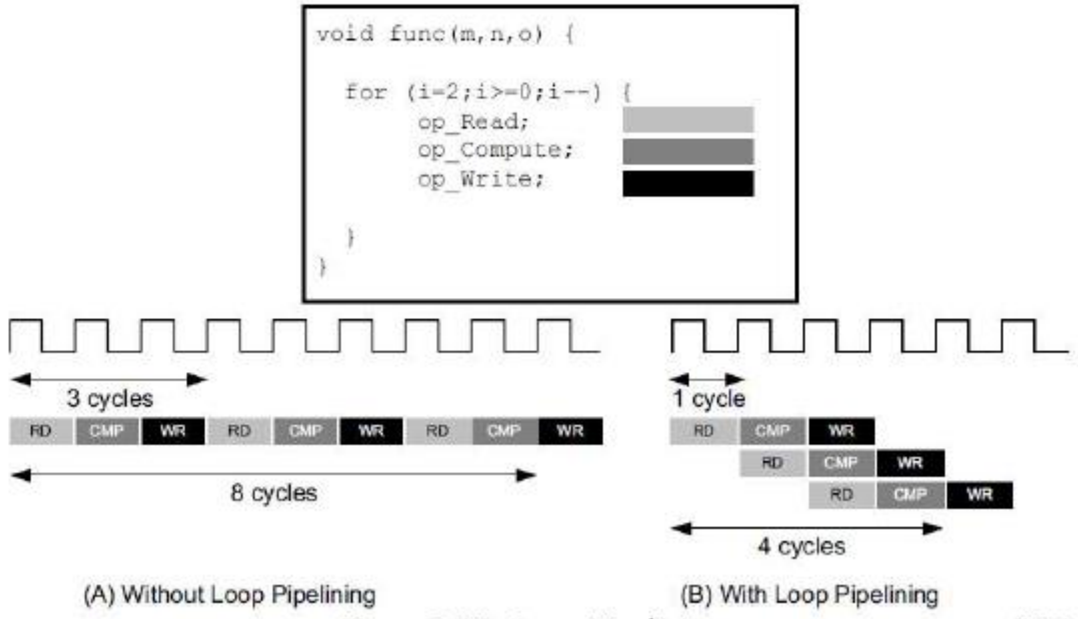
Pipelining doesn't need to wait for the completion of the previous one; all steps can proceed independently. This is useful for functions and loops.

As shown in the example below, using pipeline optimization instructions can change a loop with a cycle of 8 clocks (II = 3) to a cycle of 4 clocks.



```
void  func(…)  {

      op_Read;
      op_Compute;
      op_Write;

}
```

Figure 1-50 : Function Pipelining Behavior



```
void func(m,n,o) {

  for (i=2;i>=0;i--) {
       op_Read;
       op_Compute;
       op_Write;

  }
}
```

Figure 1-51 : Loop Pipelining

AMD
together we advance_

Pipelining

Rewinding pipelined loops

The loop can be a top-level loop in a function or in an area optimized using DATAFLOW optimization.

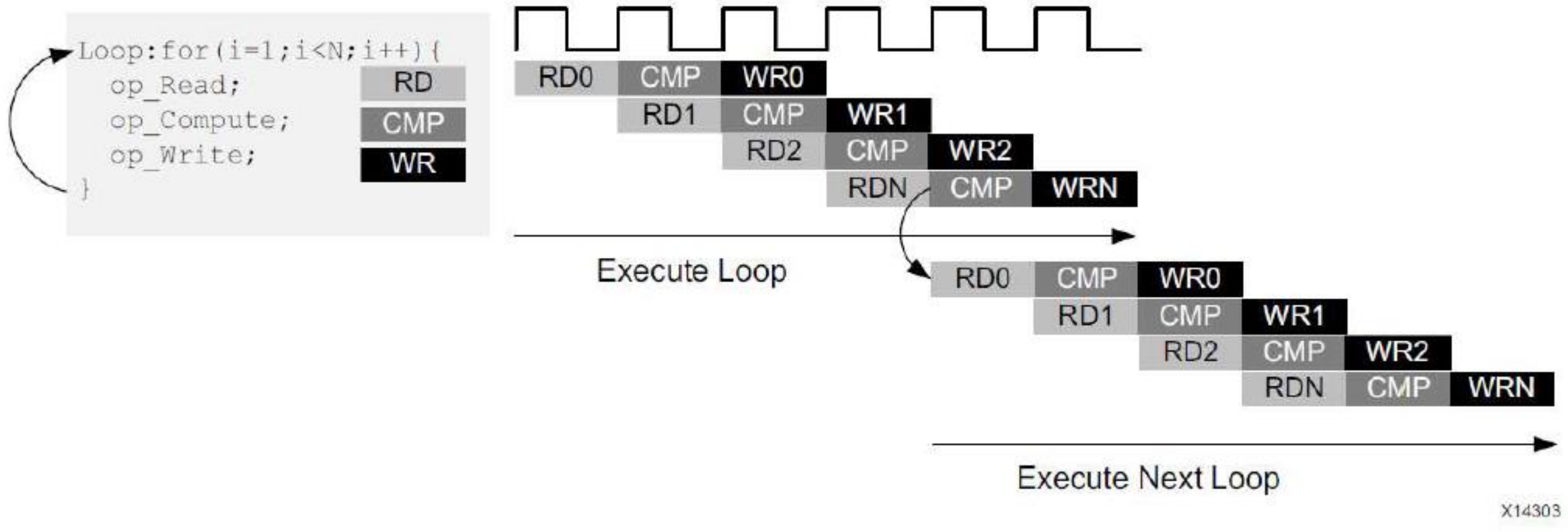

```
Loop:for(i=1;i<N;i++){
  op_Read;
  op_Compute;
  op_Write;
}
```

Figure 1-63: **Loop Pipelining with Rewind Option**

Unroll

Rolled loop

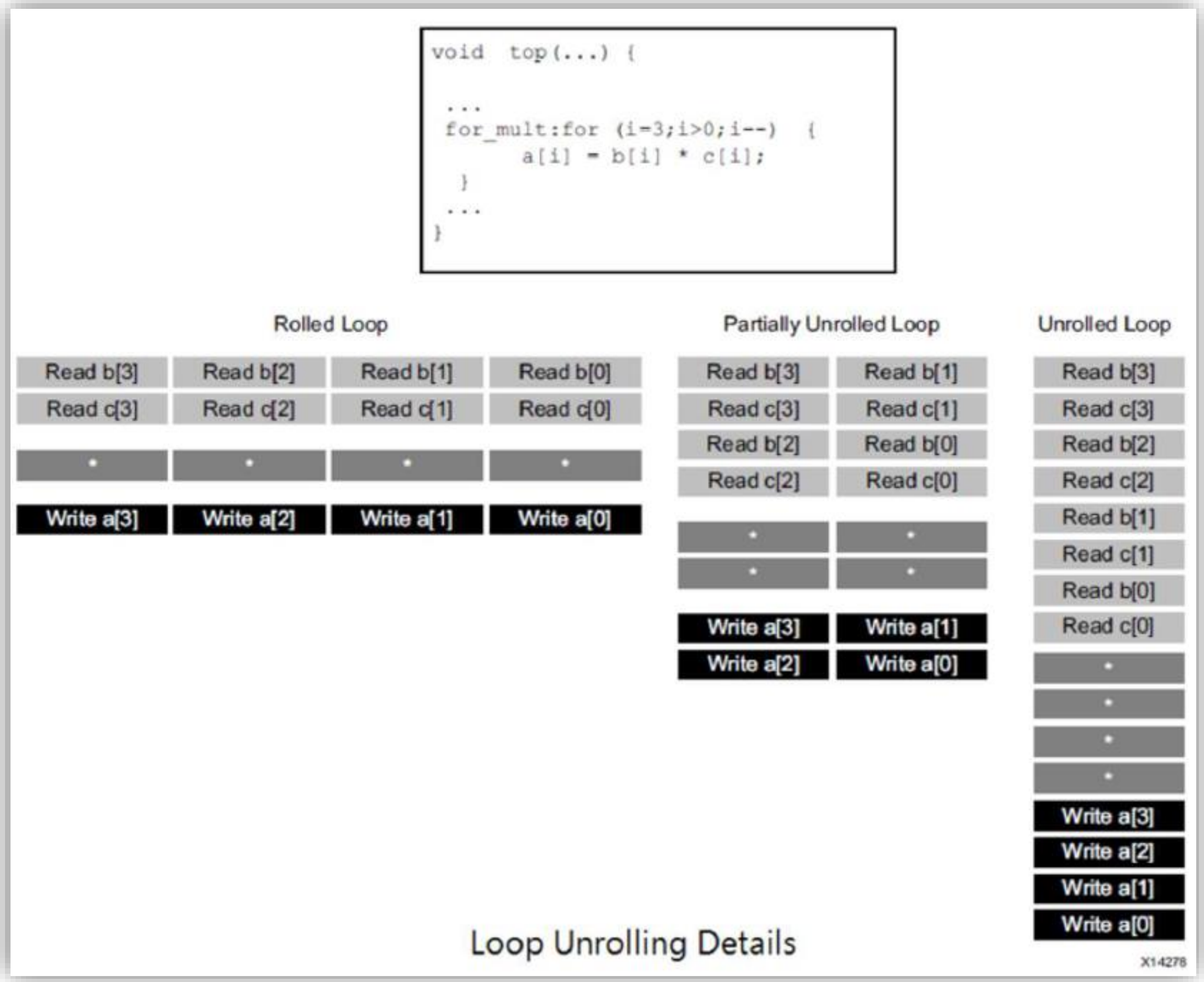
Each function operates within its own time cycle.

Partially unrolled loop

The loop will be unrolled by a factor of 2.

Unrolled loop

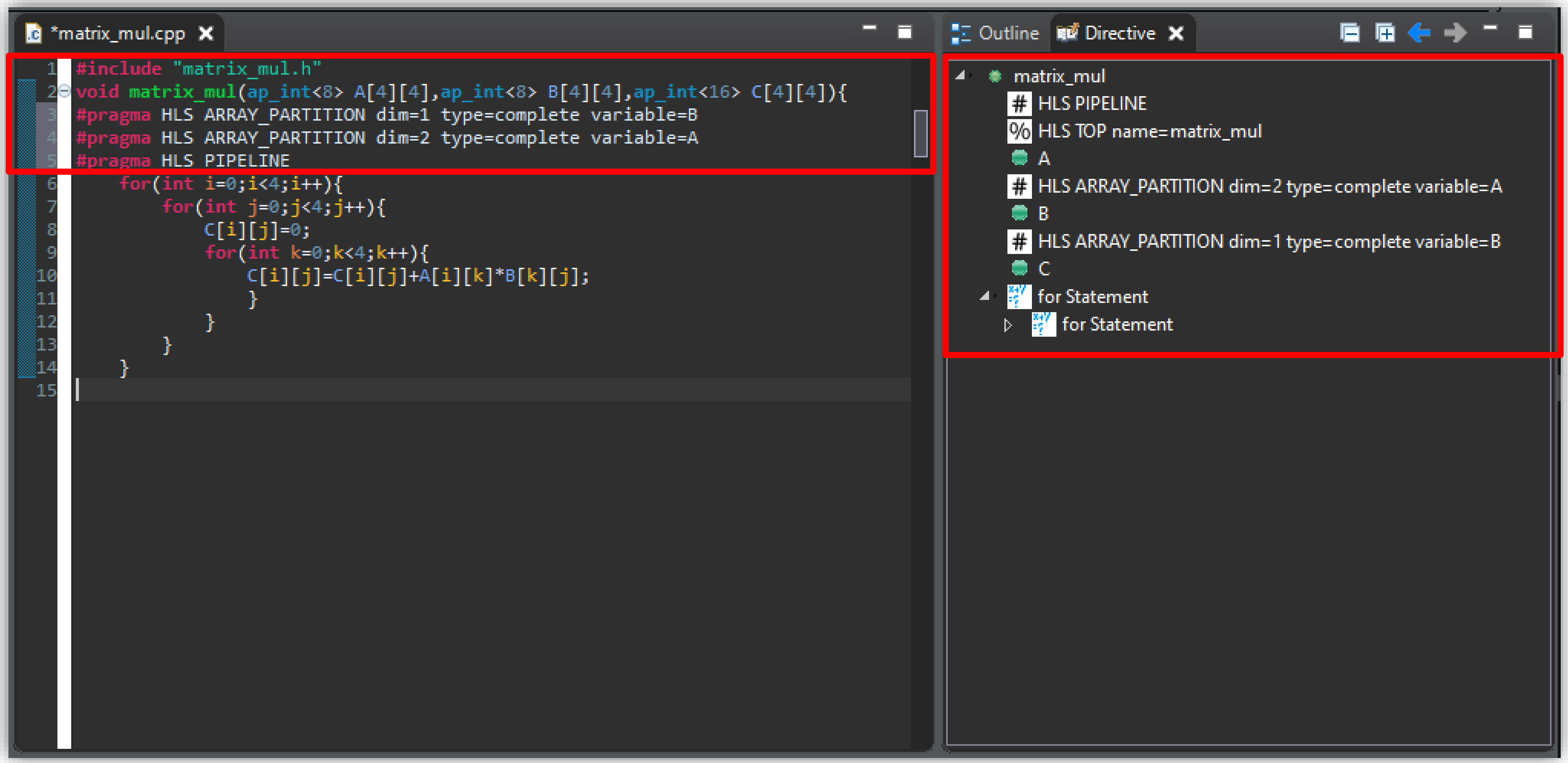
Fully unrolled, hence data partitioning is required.



```
void  top(...) {

  ...
  for_mult:for (i=3;i>0;i--)  {
        a[i] = b[i] * c[i];
   }
  ...
}
```

Loop Unrolling Details

Insert Directive



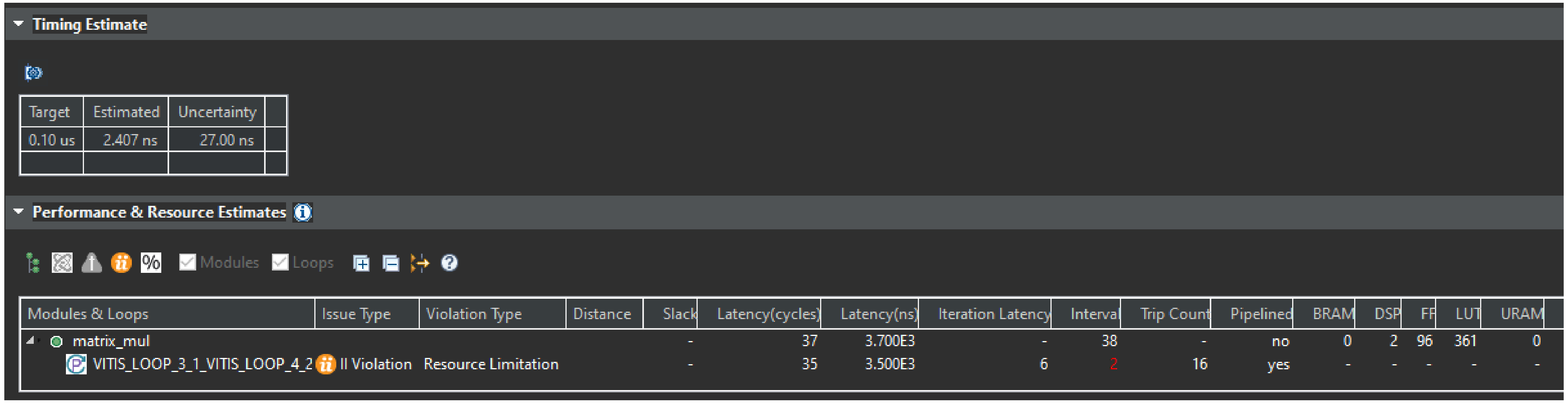
```
#include "matrix_mul.h"
void matrix_mul(ap_int<8> A[4][4],ap_int<8> B[4][4],ap_int<16> C[4][4]){
#pragma HLS ARRAY_PARTITION dim=1 type=complete variable=B
#pragma HLS ARRAY_PARTITION dim=2 type=complete variable=A
#pragma HLS PIPELINE
    for(int i=0;i<4;i++){
        for(int j=0;j<4;j++){
            C[i][j]=0;
            for(int k=0;k<4;k++){
                C[i][j]=C[i][j]+A[i][k]*B[k][j];
            }
        }
    }
}
```

```
matrix_mul
    HLS PIPELINE
    HLS TOP name=matrix_mul
    A
    HLS ARRAY_PARTITION dim=2 type=complete variable=A
    B
    HLS ARRAY_PARTITION dim=1 type=complete variable=B
    C
    for Statement
        for Statement
```

[Public]

HLS solutions comparison

Solution1:
No Optimization



Timing Estimate

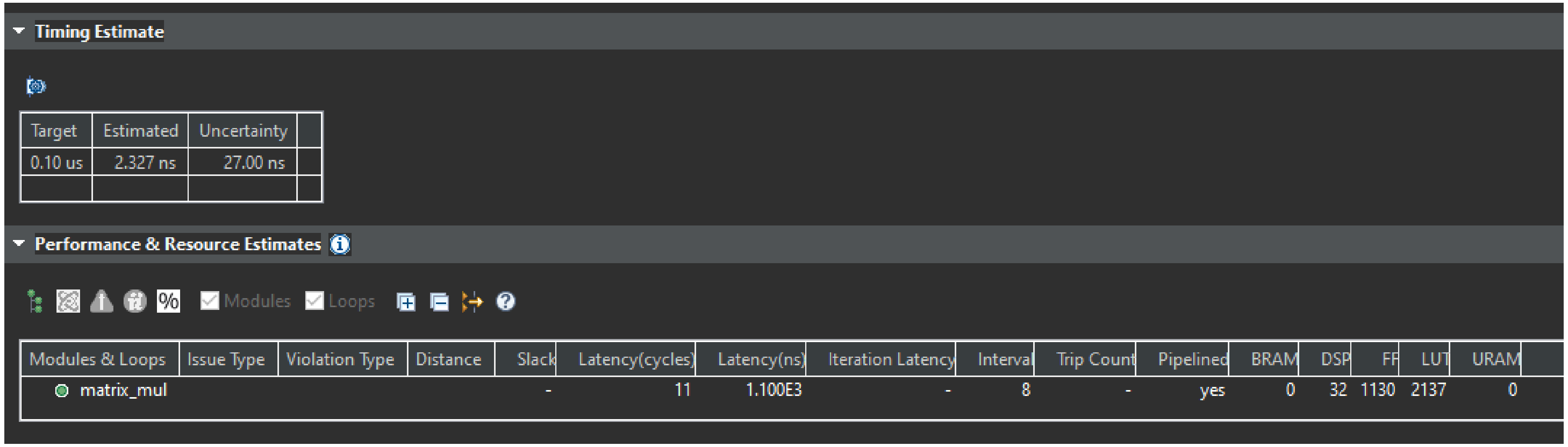
Target	Estimated	Uncertainty
0.10 us	2.407 ns	27.00 ns

Performance & Resource Estimates

Modules Loops

Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
matrix_mul				-	37	3.700E3	-	38	-	no	0	2	96	361	0
VITIS_LOOP_3_1_VITIS_LOOP_4_2	II Violation	Resource Limitation		-	35	3.500E3	6	2	16	yes	-	-	-	-	-

Solution2:
Pipeline
ARRAY_PARTITION



Timing Estimate

Target	Estimated	Uncertainty
0.10 us	2.327 ns	27.00 ns

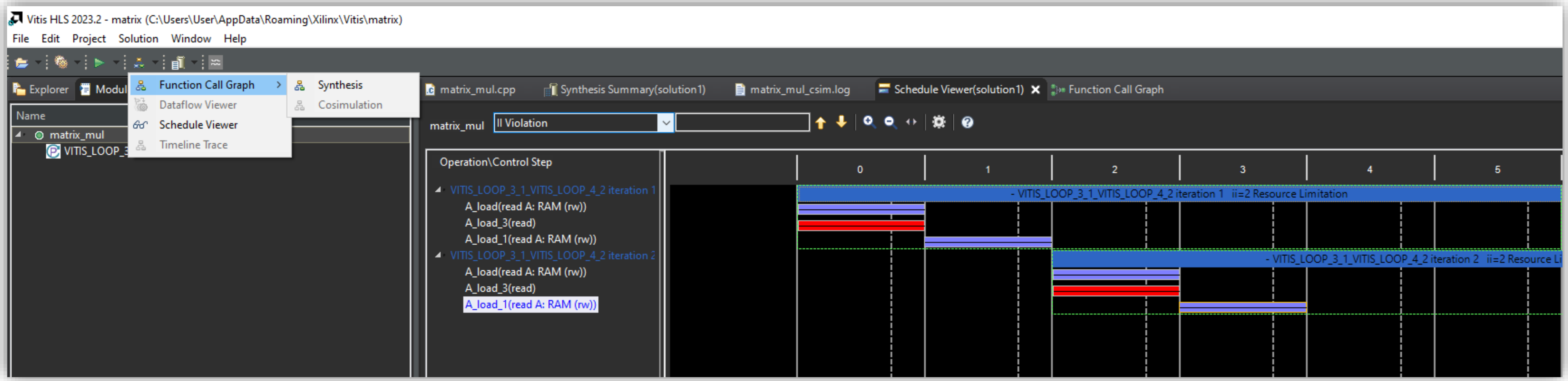
Performance & Resource Estimates

Modules Loops

Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
matrix_mul				-	11	1.100E3	-	8	-	yes	0	32	1130	2137	0

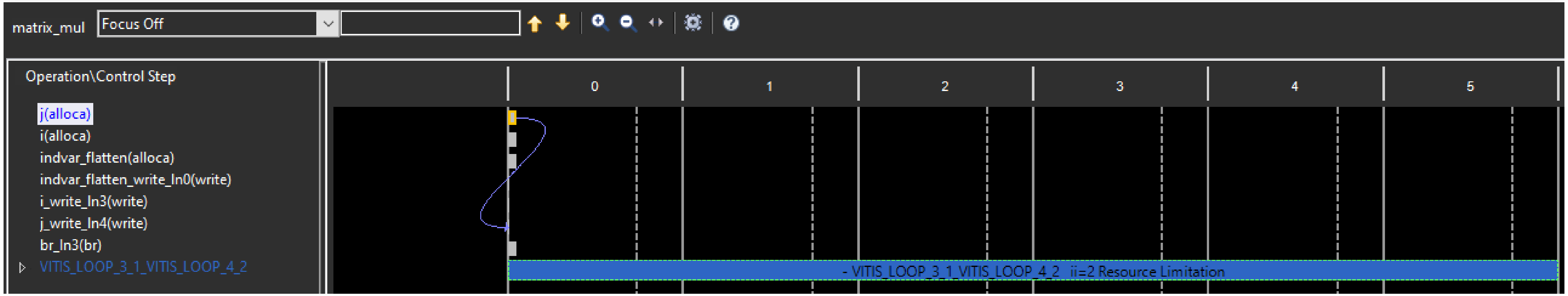
AMD
together we advance_

Schedule Viewer


Vitis HLS 2023.2 - matrix (C:\Users\User\AppData\Roaming\Xilinx\Vitis\matrix)

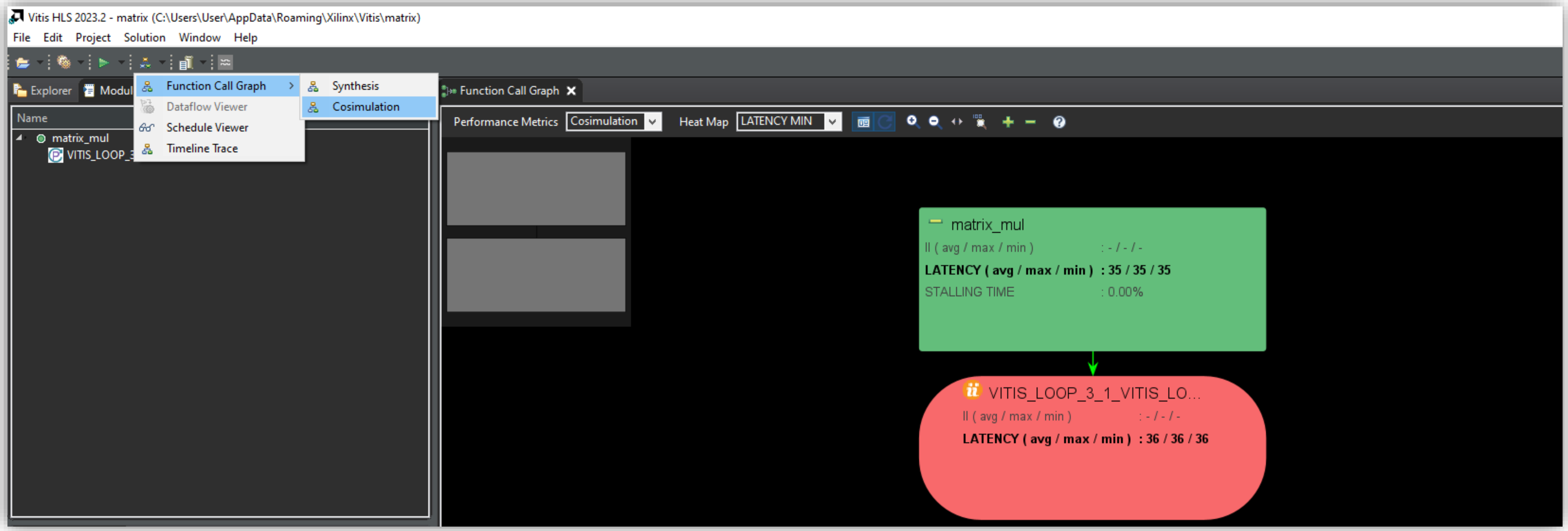
Operation\Control Step
- VITIS_LOOP_3_1_VITIS_LOOP_4_2 iteration 1
  - A_load(read A: RAM (rw))
  - A_load_3(read)
  - A_load_1(read A: RAM (rw))
- VITIS_LOOP_3_1_VITIS_LOOP_4_2 iteration 2
  - A_load(read A: RAM (rw))
  - A_load_3(read)
  - A_load_1(read A: RAM (rw))

0
1
2
3
4
5
- VITIS_LOOP_3_1_VITIS_LOOP_4_2 iteration 1 ii=2 Resource Limitation
- VITIS_LOOP_3_1_VITIS_LOOP_4_2 iteration 2 ii=2 Resource Li



Operation\Control Step
- j(alloca)
- i(alloca)
- indvar_flatten(alloca)
- indvar_flatten_write_ln0(write)
- i_write_ln3(write)
- j_write_ln4(write)
- br_ln3(br)
- VITIS_LOOP_3_1_VITIS_LOOP_4_2

Function Call Graph

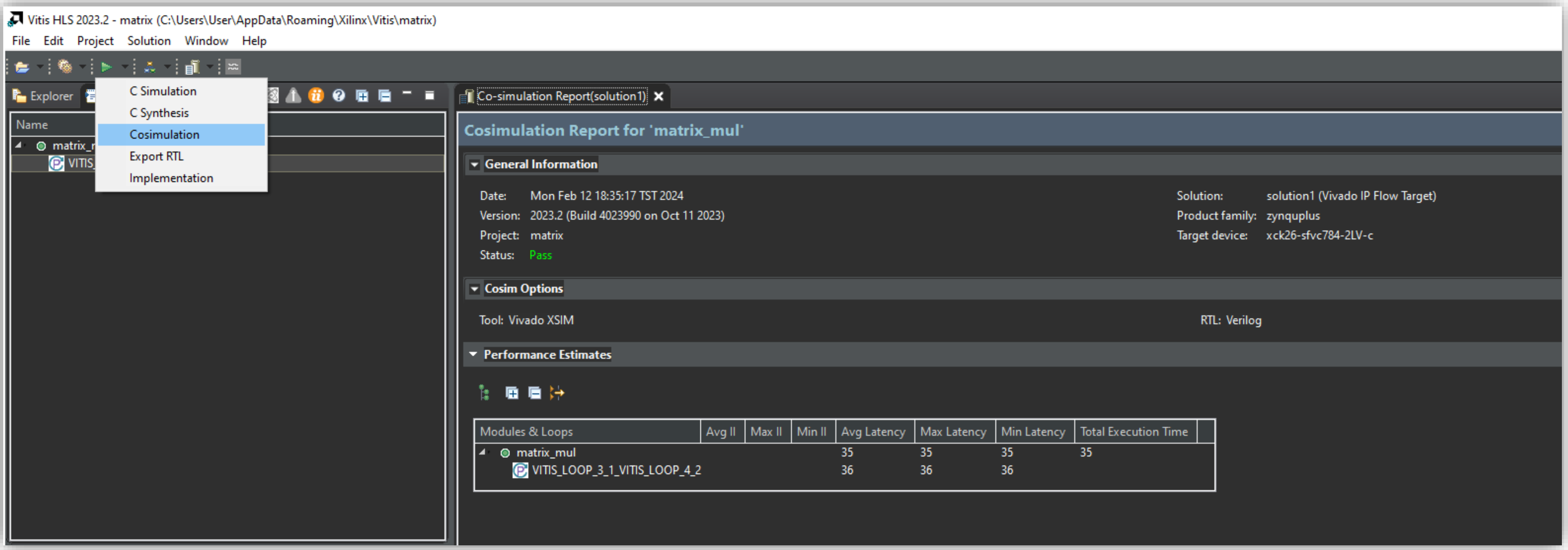

Vitis HLS 2023.2 - matrix (C:\Users\User\AppData\Roaming\Xilinx\Vitis\matrix)

matrix_mul
II (avg / max / min) : - / - / -
LATENCY (avg / max / min) : 35 / 35 / 35
STALLING TIME : 0.00%

VITIS_LOOP_3_1_VITIS_LO...
II (avg / max / min) : - / - / -
LATENCY (avg / max / min) : 36 / 36 / 36

AMD
together we advance_

Cosimulation & Timeline Trace


Vitis HLS 2023.2 - matrix (C:\Users\User\AppData\Roaming\Xilinx\Vitis\matrix)
File Edit Project Solution Window Help
Explorer
Name
matrix_
VITIS_
C Simulation
C Synthesis
Cosimulation
Export RTL
Implementation
Co-simulation Report(solution1)

Cosimulation Report for 'matrix_mul'

General Information

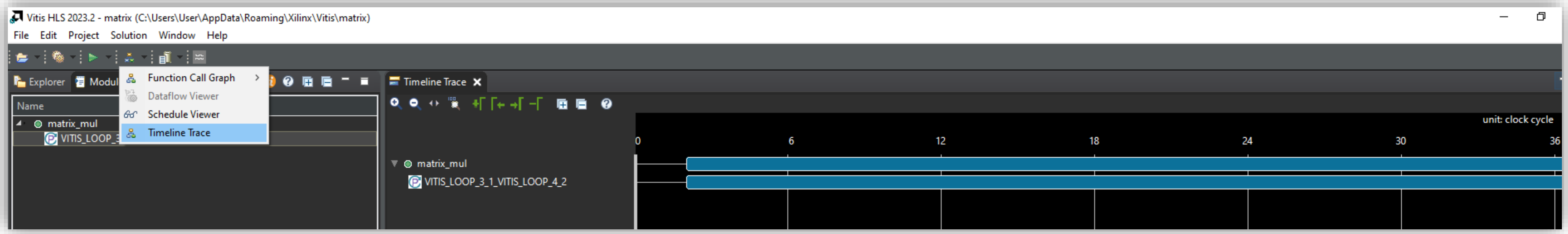
Date:	Mon Feb 12 18:35:17 TST 2024	Solution:	solution1 (Vivado IP Flow Target)
Version:	2023.2 (Build 4023990 on Oct 11 2023)	Product family:	zynquplus
Project:	matrix	Target device:	xck26-sfvc784-2LV-c
Status:	Pass		

Cosim Options

Tool: Vivado XSIM RTL: Verilog

Performance Estimates

Modules & Loops	Avg II	Max II	Min II	Avg Latency	Max Latency	Min Latency	Total Execution Time
matrix_mul				35	35	35	35
VITIS_LOOP_3_1_VITIS_LOOP_4_2				36	36	36	

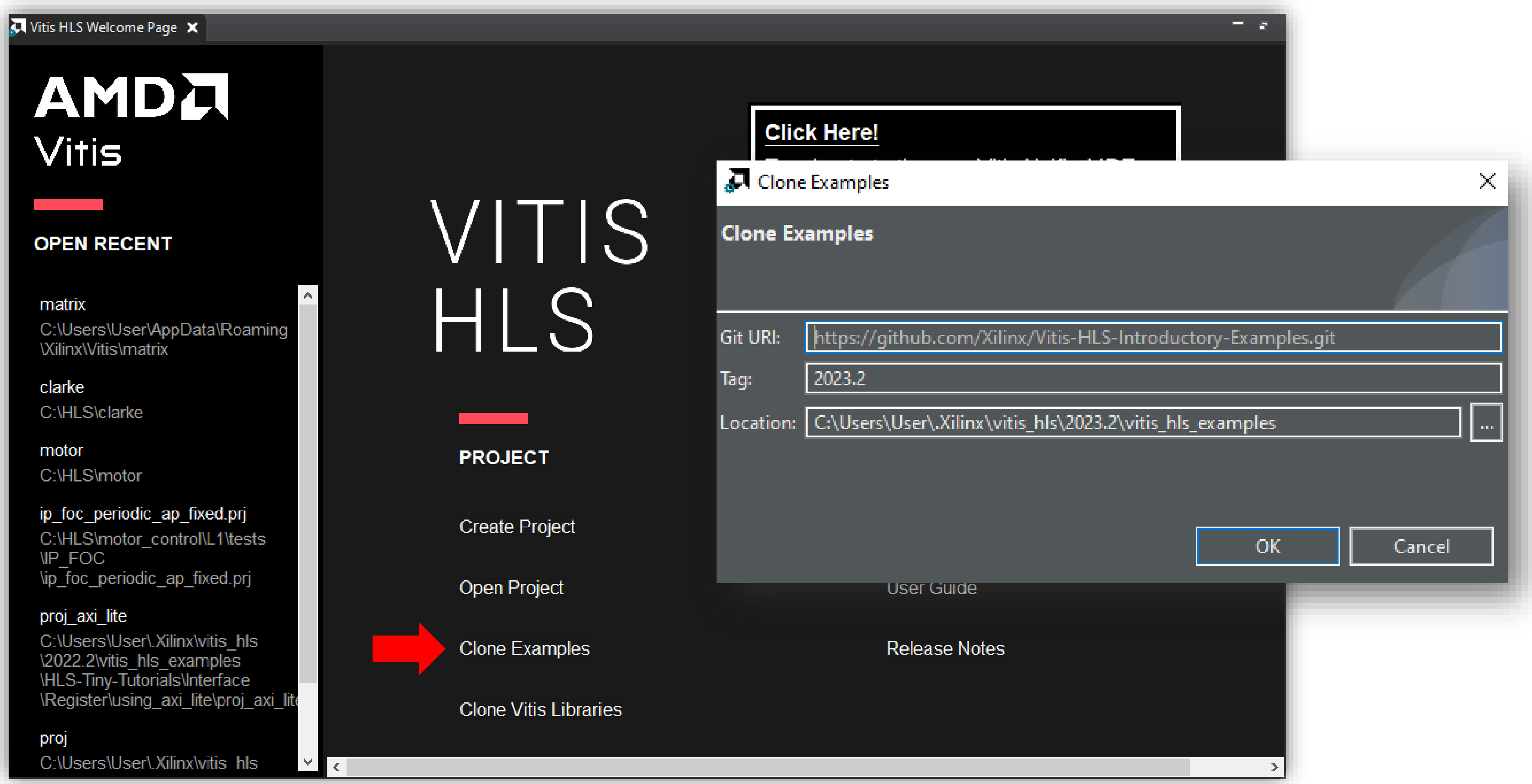


AMD
together we advance_

HLS Example

Clone Examples

Click OK will download the whole example code directory automatically.



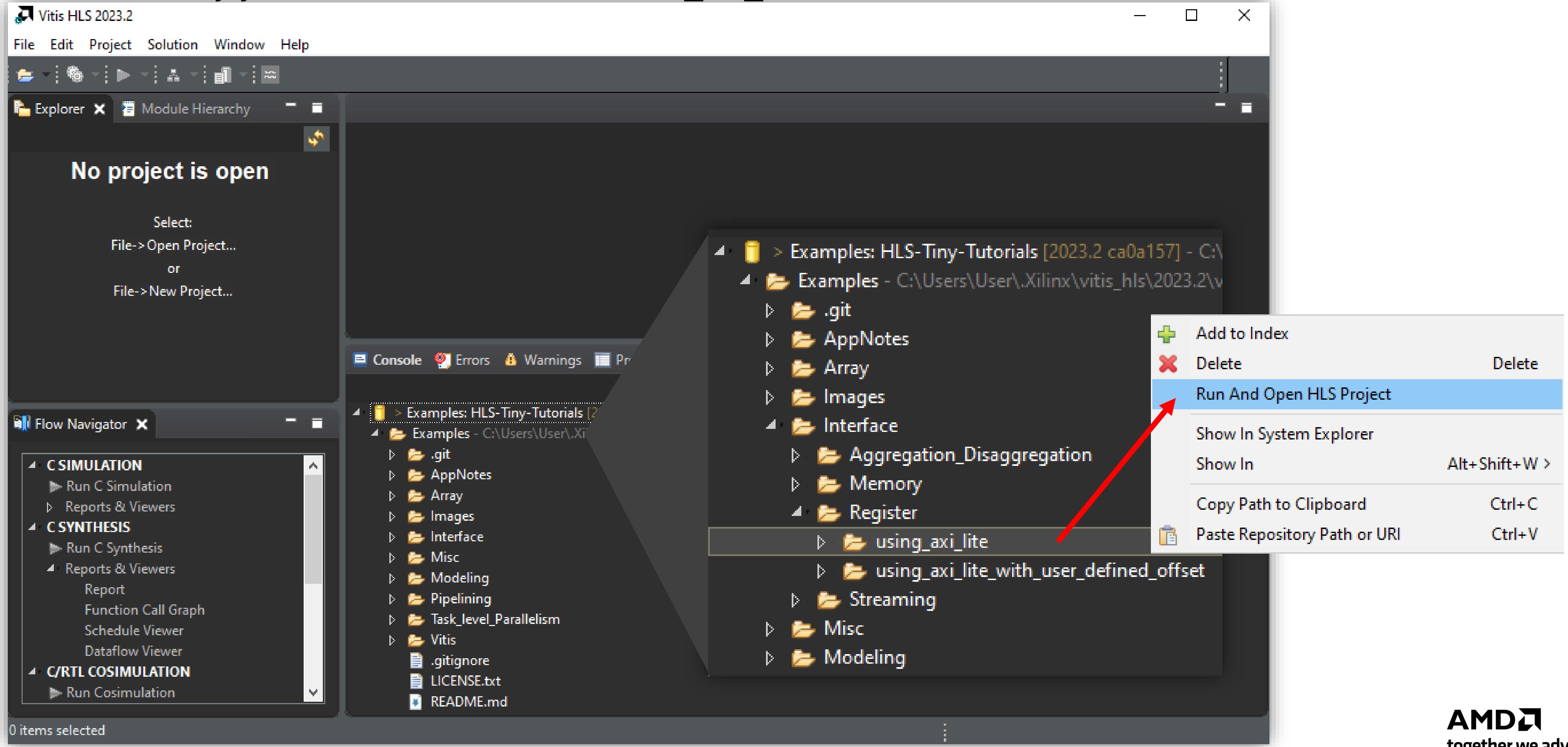
matrix
C:\Users\User\AppData\Roaming\Xilinx\Vitis\matrix
clarke
C:\HLS\clarke
motor
C:\HLS\motor
ip_foc_periodic_ap_fixed.prj
C:\HLS\motor_control\L1\tests\IP_FOC\ip_foc_periodic_ap_fixed.prj
proj_axi_lite
C:\Users\User\.Xilinx\vitis_hls\2022.2\vitis_hls_examples\HLS-Tiny-Tutorials\Interface\Register\using_axi_lite\proj_axi_lit
proj
C:\Users\User\.Xilinx\vitis_hls

Clone Examples

Git URI: https://github.com/Xilinx/Vitis-HLS-Introductory-Examples.git
Tag: 2023.2
Location: C:\Users\User\.Xilinx\vitis_hls\2023.2\vitis_hls_examples

Clone Examples

Click the directory you want, we choose Interface_axi_lite this time



[Public]

example.cpp

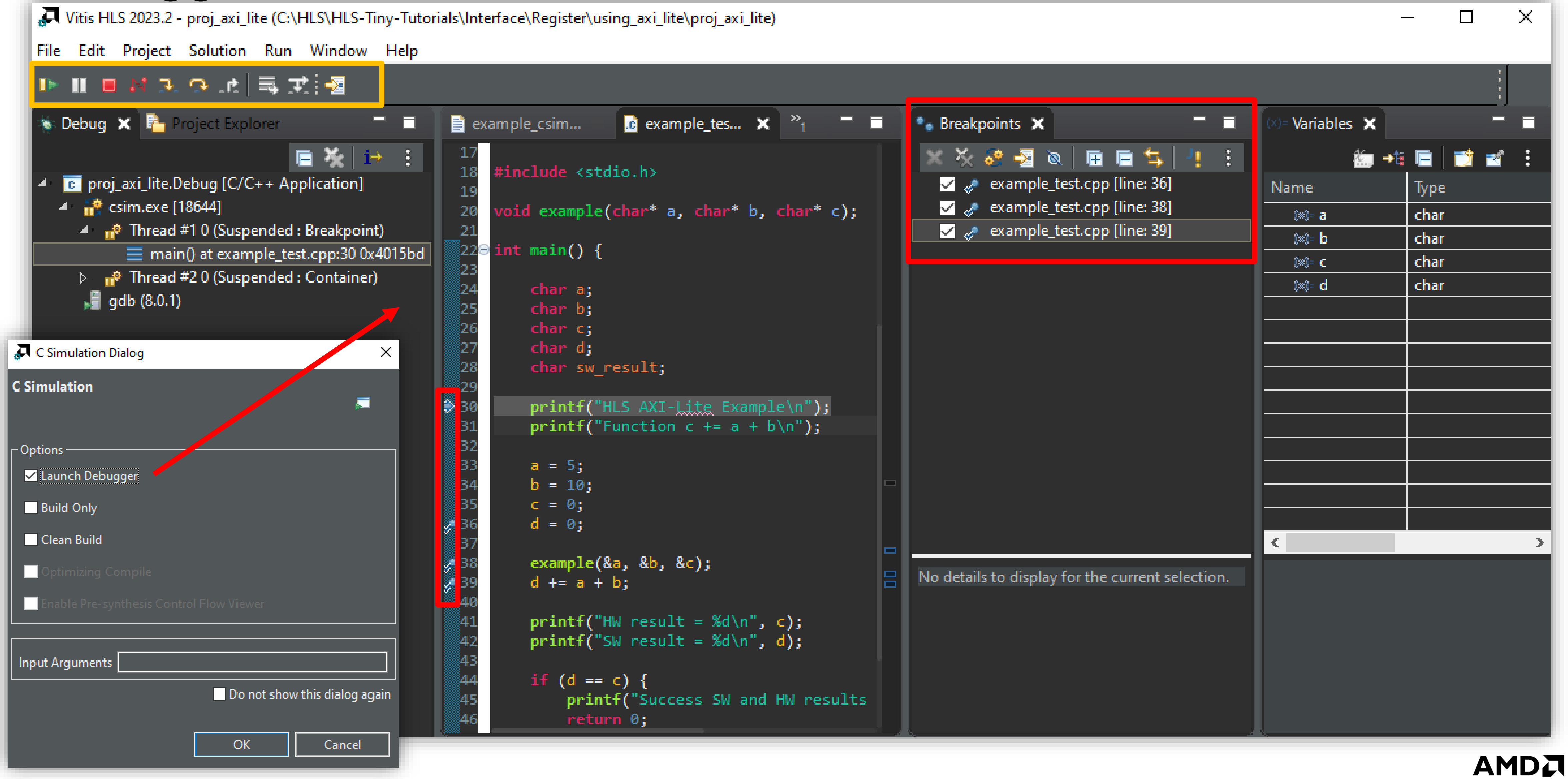
```
#include <stdio.h>

void example(char* a, char* b, char* c) {
        //Define AXI-Lite Port
        #pragma HLS INTERFACE s_axilite port = a bundle = BUS_A
        #pragma HLS INTERFACE s_axilite port = b bundle = BUS_A
        #pragma HLS INTERFACE s_axilite port = c bundle = BUS_A
        #pragma HLS INTERFACE s_axilite port = return bundle = BUS_A
        //#pragma HLS interface <mode> port=<name> (register) bundle=<string>

        *c += *a + *b;
    return *c;
}
```

Example_test.cpp

```
#include <stdio.h>

void example(char* a, char* b, char* c);

int main() {

    char a;
    char b;
    char c;
    char d;
    char sw_result;

    printf("HLS AXI-Lite Example\n");
    printf("Function c += a + b\n");

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

```
 example(&a, &b, &c);
    d += a + b;

    printf("HW result = %d\n", c);
    printf("SW result = %d\n", d);

    if (d == c) {
        printf("SW and HW results match\n");
        return 0;
    } else {
        printf("SW and HW results mismatch\n");
        return 1;
    }
}
```

Debugger


Vitis HLS 2023.2 - proj_axi_lite (C:\HLS\HLS-Tiny-Tutorials\Interface\Register\using_axi_lite\proj_axi_lite)
File Edit Project Solution Run Window Help
Debug
Project Explorer
example_csim...
example_tes...
Breakpoints
Variables
C Simulation Dialog
C Simulation
OK
Cancel

```
proj_axi_lite.Debug [C/C++ Application]
  csim.exe [18644]
    Thread #1 0 (Suspended : Breakpoint)
      main() at example_test.cpp:30 0x4015bd
    Thread #2 0 (Suspended : Container)
  gdb (8.0.1)
```

```
17
18 #include <stdio.h>
19
20 void example(char* a, char* b, char* c);
21
22 int main() {
23
24     char a;
25     char b;
26     char c;
27     char d;
28     char sw_result;
29
30     printf("HLS AXI-Lite Example\n");
31     printf("Function c += a + b\n");
32
33     a = 5;
34     b = 10;
35     c = 0;
36     d = 0;
37
38     example(&a, &b, &c);
39     d += a + b;
40
41     printf("HW result = %d\n", c);
42     printf("SW result = %d\n", d);
43
44     if (d == c) {
45         printf("Success SW and HW results
46         return 0;
```

Breakpoints:

- ✓ example_test.cpp [line: 36]
- ✓ example_test.cpp [line: 38]
- ✓ example_test.cpp [line: 39]

No details to display for the current selection.

Name	Type
a	char
b	char
c	char
d	char

Options

- [✓] Launch Debugger
- [ ] Build Only
- [ ] Clean Build
- [ ] Optimizing Compile
- [ ] Enable Pre-synthesis Control Flow Viewer

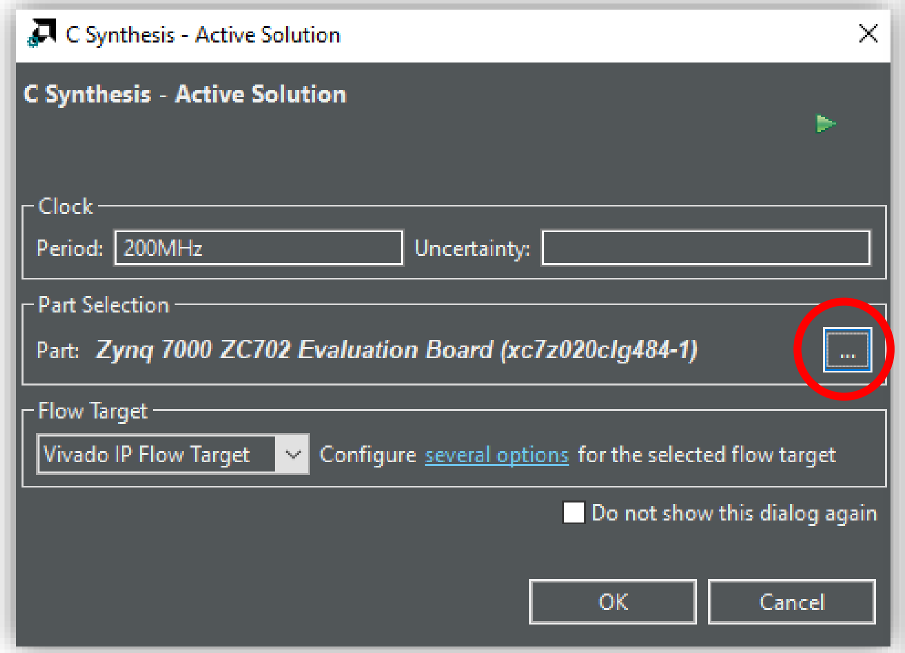
Input Arguments

- [ ] Do not show this dialog again

AMD
together we advance_

C Synthesis

KD240 is only available starting from version 2023.2.



C Synthesis - Active Solution

C Synthesis - Active Solution

Clock

Period: 200MHz Uncertainty:

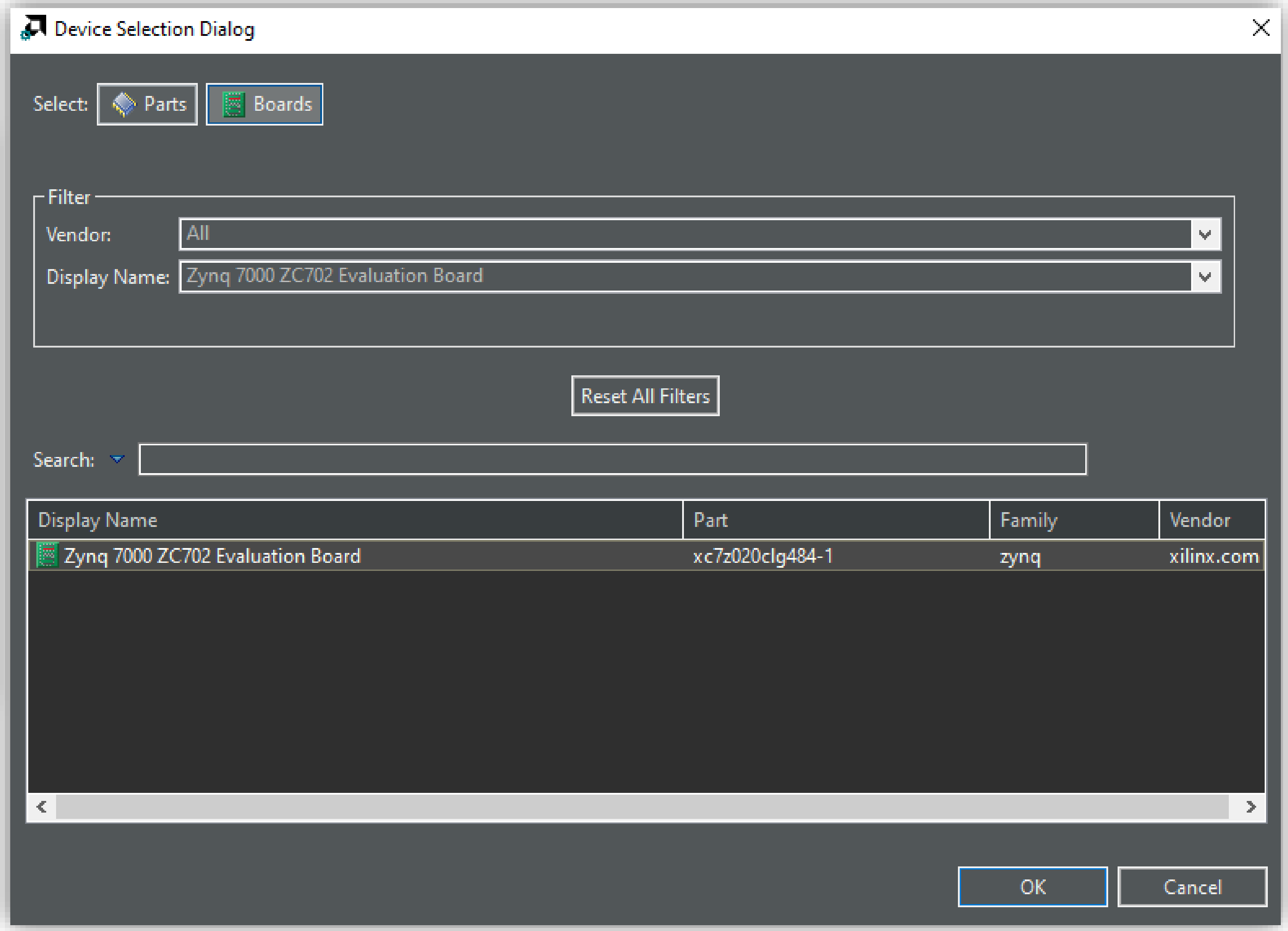
Part Selection

Part: *Zynq 7000 ZC702 Evaluation Board (xc7z020clg484-1)*

Flow Target

Vivado IP Flow Target Configure several options for the selected flow target

Do not show this dialog again



Device Selection Dialog

Select: Parts Boards

Filter

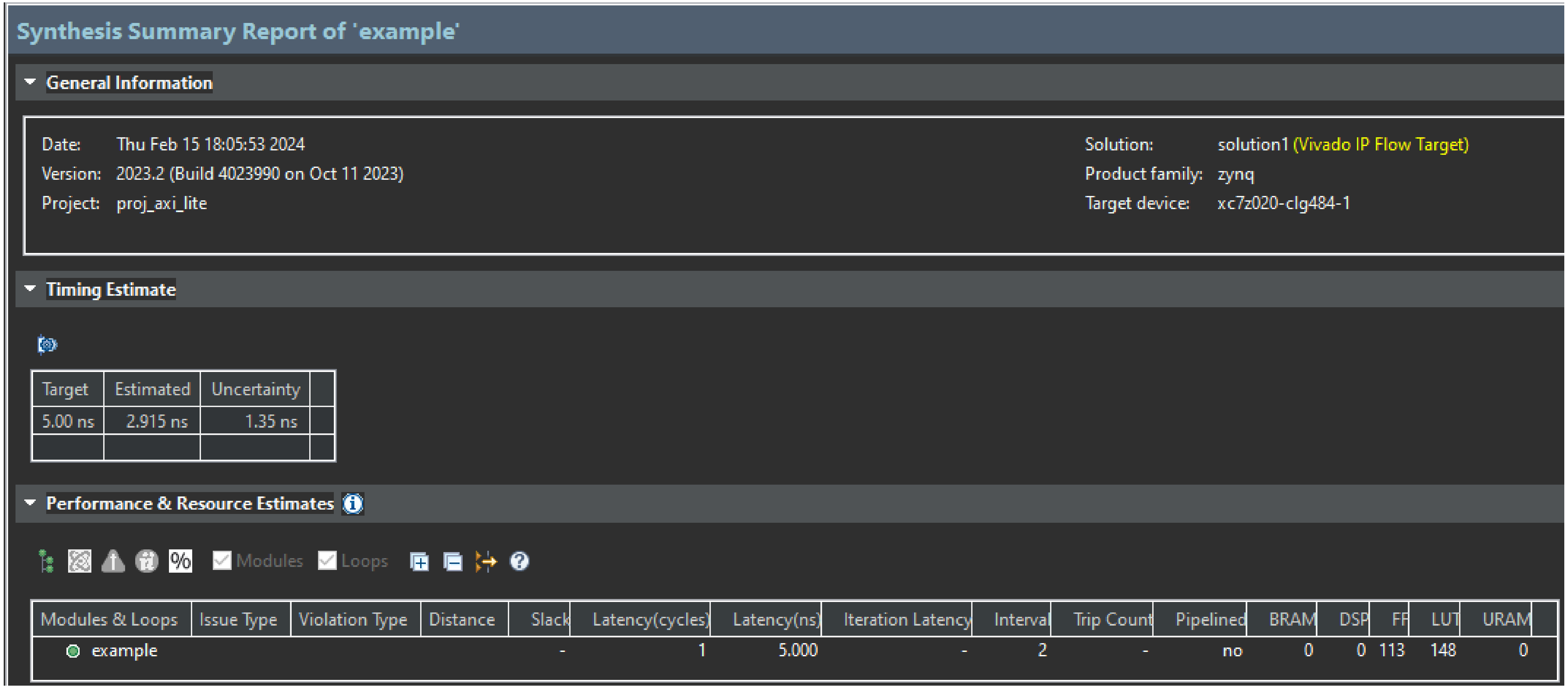
Vendor: All

Display Name: Zynq 7000 ZC702 Evaluation Board

Search:

Display Name	Part	Family	Vendor
Zynq 7000 ZC702 Evaluation Board	xc7z020clg484-1	zynq	xilinx.com

C Synthesis



Synthesis Summary Report of 'example'

General Information

Date:	Thu Feb 15 18:05:53 2024	Solution:	solution1 (Vivado IP Flow Target)
Version:	2023.2 (Build 4023990 on Oct 11 2023)	Product family:	zynq
Project:	proj_axi_lite	Target device:	xc7z020-clg484-1

Timing Estimate

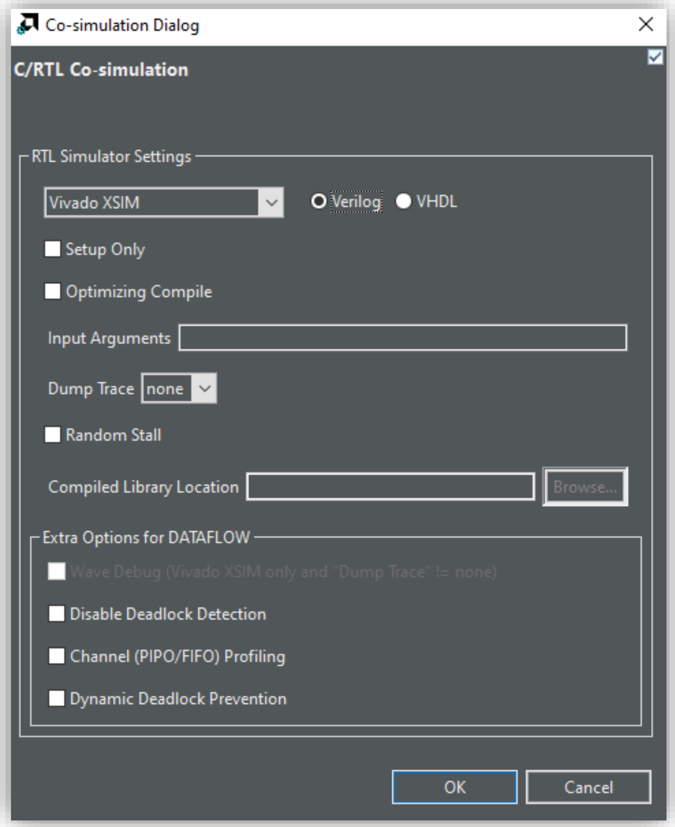
Target	Estimated	Uncertainty
5.00 ns	2.915 ns	1.35 ns

Performance & Resource Estimates

Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
● example				-	1	5.000	-	2	-	no	0	0	113	148	0

AMD
together we advance_

Co-Simulation



Co-simulation Dialog

C/RTL Co-simulation

RTL Simulator Settings

Vivado XSIM [✓ Verilog] VHDL

- ☐ Setup Only
- ☐ Optimizing Compile

Input Arguments:

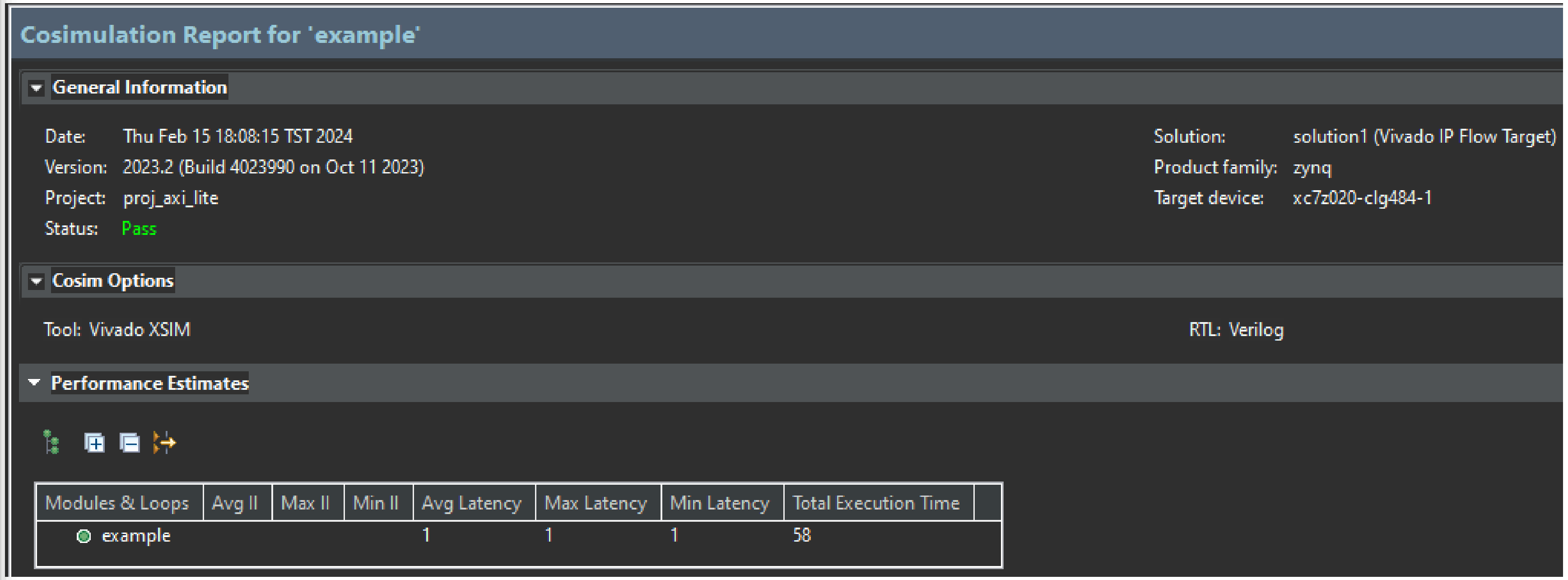
Dump Trace: none

- ☐ Random Stall

Compiled Library Location:

Extra Options for DATAFLOW

- ☐ Wave Debug (Vivado XSIM only and "Dump Trace" != none)
- ☐ Disable Deadlock Detection
- ☐ Channel (PIPO/FIFO) Profiling
- ☐ Dynamic Deadlock Prevention



Cosimulation Report for 'example'

General Information

Date:	Thu Feb 15 18:08:15 TST 2024	Solution:	solution1 (Vivado IP Flow Target)
Version:	2023.2 (Build 4023990 on Oct 11 2023)	Product family:	zynq
Project:	proj_axi_lite	Target device:	xc7z020-clg484-1
Status:	Pass		

Cosim Options

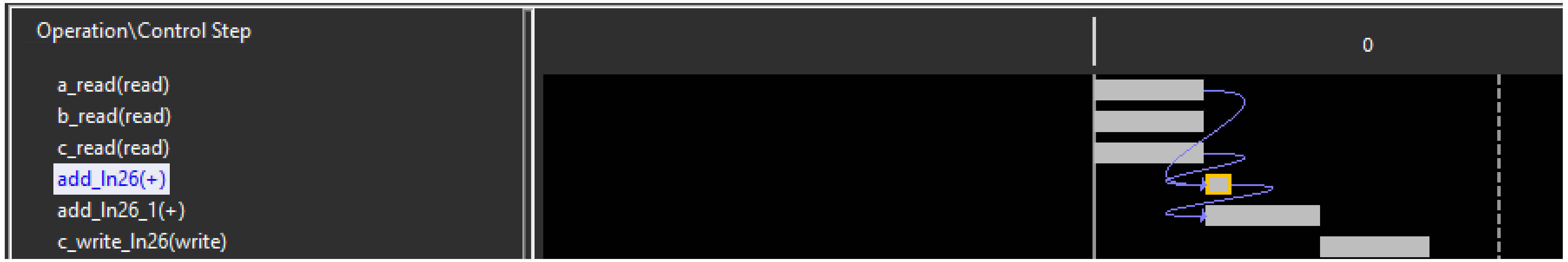
Tool: Vivado XSIM   RTL: Verilog

Performance Estimates

Modules & Loops	Avg II	Max II	Min II	Avg Latency	Max Latency	Min Latency	Total Execution Time
● example				1	1	1	58

Schedule View

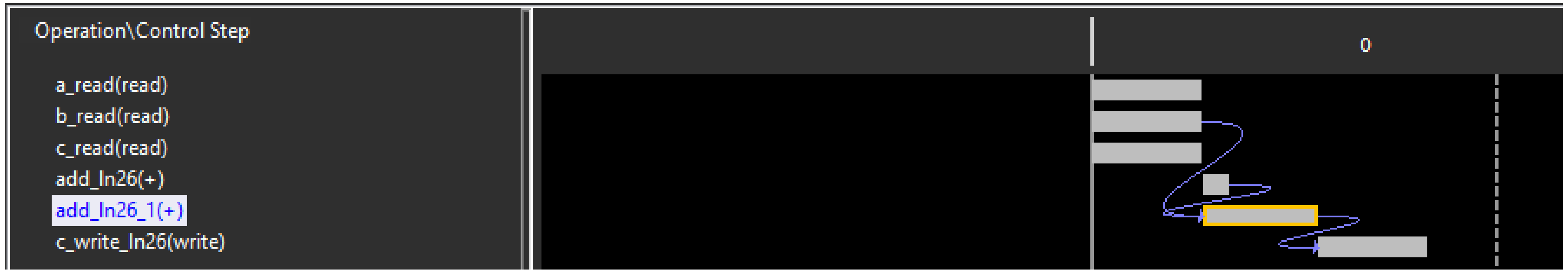
First, read the values of 'a' and 'c' and then add them together.



Operation\Control Step

a_read(read)
b_read(read)
c_read(read)
add_ln26(+)
add_ln26_1(+)
c_write_ln26(write)

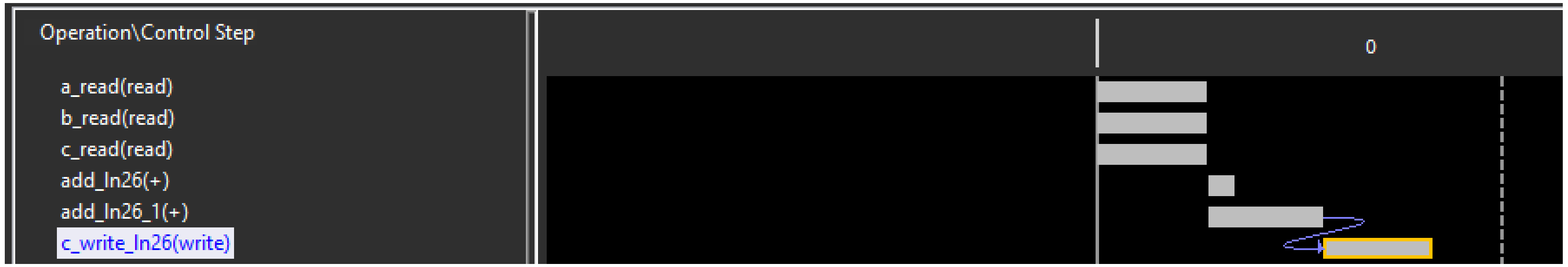
Then, add the value of 'b' to the sum of 'a' and 'c'.



Operation\Control Step

a_read(read)
b_read(read)
c_read(read)
add_ln26(+)
add_ln26_1(+)
c_write_ln26(write)

Write the final value into 'c'.

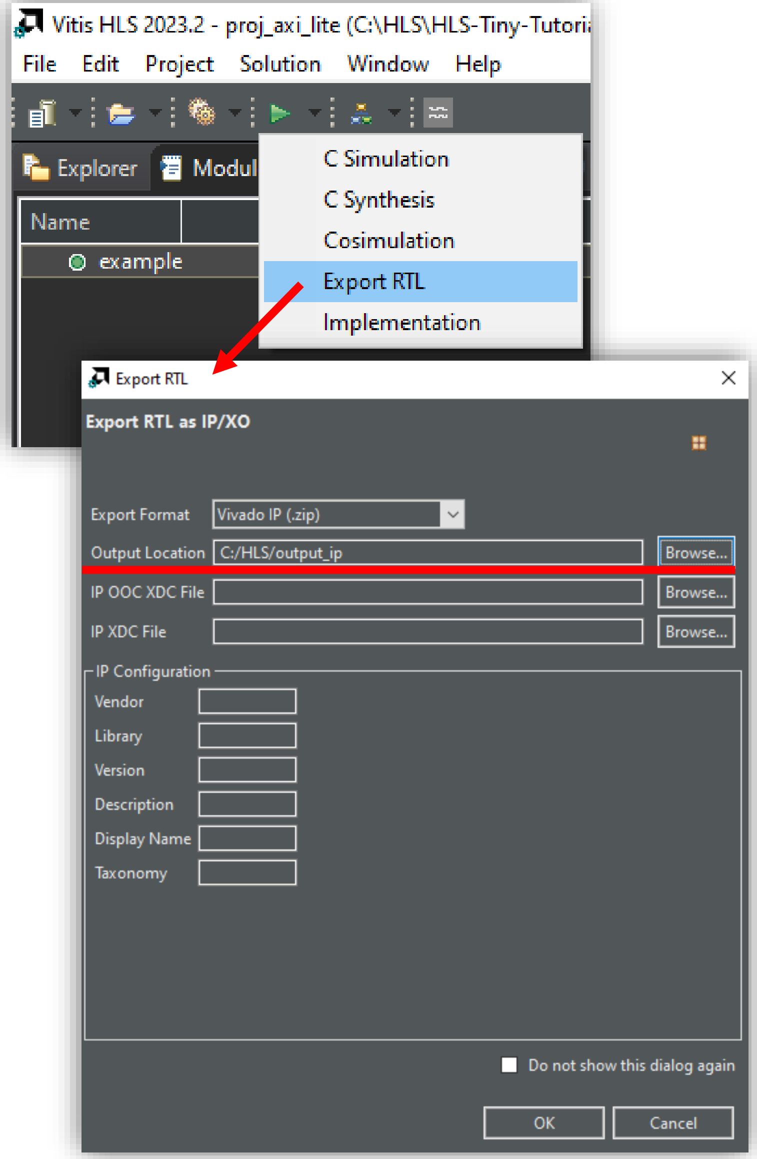


Operation\Control Step

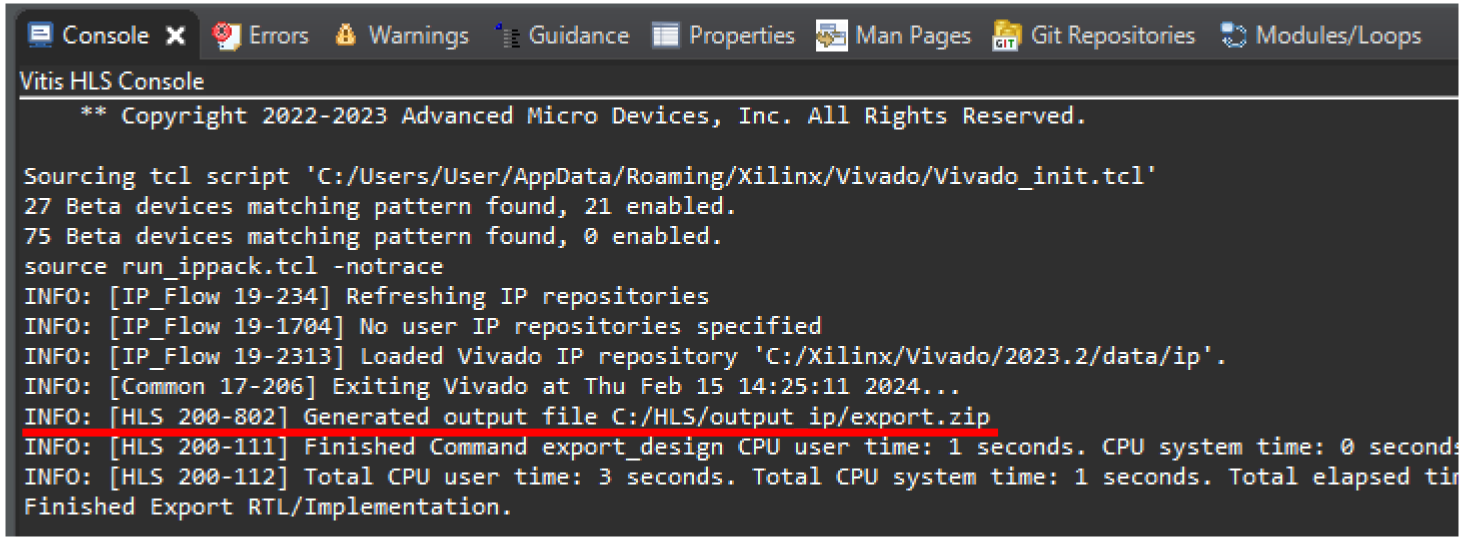
a_read(read)
b_read(read)
c_read(read)
add_ln26(+)
add_ln26_1(+)
c_write_ln26(write)

AMD
together we advance_

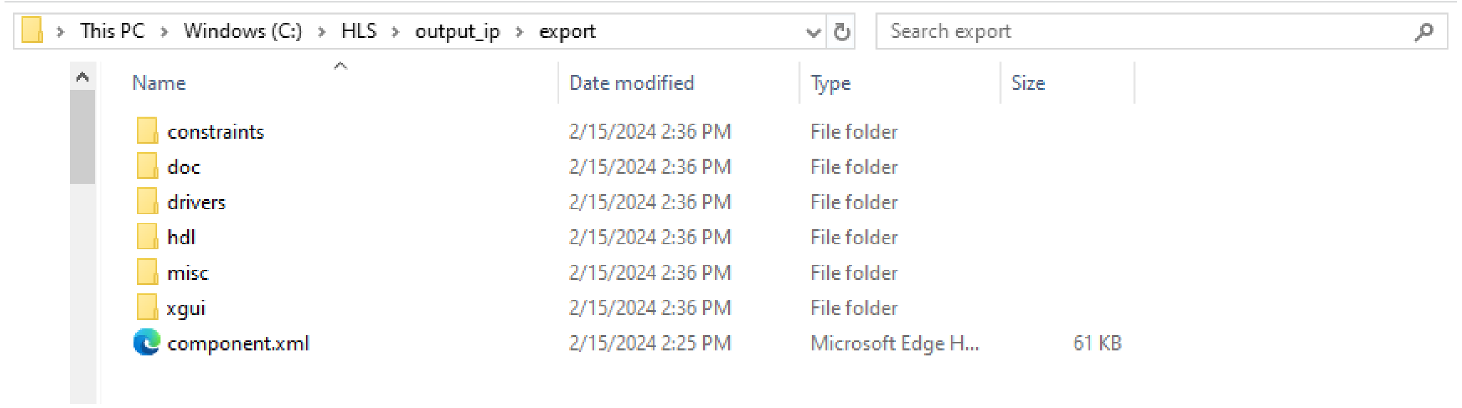
Export RTL



Export Format: Vivado IP (.zip)
Output Location: C:/HLS/output_ip
IP OOC XDC File:
IP XDC File:
Vendor:
Library:
Version:
Description:
Display Name:
Taxonomy:



```
    ** Copyright 2022-2023 Advanced Micro Devices, Inc. All Rights Reserved.

Sourcing tcl script 'C:/Users/User/AppData/Roaming/Xilinx/Vivado/Vivado_init.tcl'
27 Beta devices matching pattern found, 21 enabled.
75 Beta devices matching pattern found, 0 enabled.
source run_ippack.tcl -notrace
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 'C:/Xilinx/Vivado/2023.2/data/ip'.
INFO: [Common 17-206] Exiting Vivado at Thu Feb 15 14:25:11 2024...
INFO: [HLS 200-802] Generated output file C:/HLS/output_ip/export.zip
INFO: [HLS 200-111] Finished Command export_design CPU user time: 1 seconds. CPU system time: 0 second
INFO: [HLS 200-112] Total CPU user time: 3 seconds. Total CPU system time: 1 seconds. Total elapsed ti
Finished Export RTL/Implementation.
```

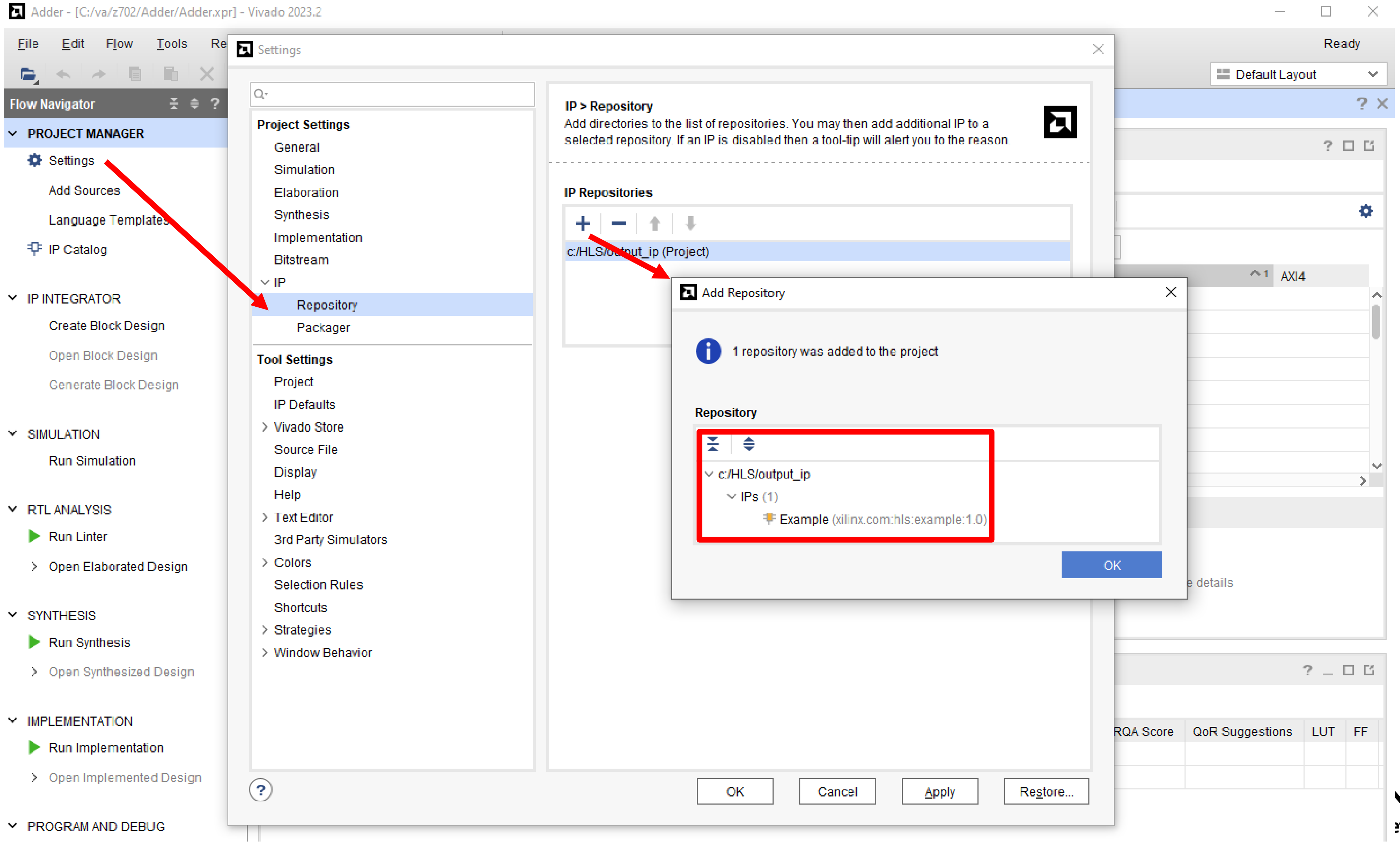

This PC > Windows (C:) > HLS > output_ip > export

Search export

Name	Date modified	Type	Size
constraints	2/15/2024 2:36 PM	File folder	
doc	2/15/2024 2:36 PM	File folder	
drivers	2/15/2024 2:36 PM	File folder	
hdl	2/15/2024 2:36 PM	File folder	
misc	2/15/2024 2:36 PM	File folder	
xgui	2/15/2024 2:36 PM	File folder	
component.xml	2/15/2024 2:25 PM	Microsoft Edge H...	61 KB

AMD together we advance_

Vivado IP repo


Adder - [C:/va/z702/Adder/Adder.xpr] - Vivado 2023.2

IP > Repository

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

c:/HLS/output_ip (Project)

Add Repository

1 repository was added to the project

Repository

c:/HLS/output_ip
IPs (1)
Example (xilinx.com:hls:example:1.0)

[Public]

Vivado IP repo

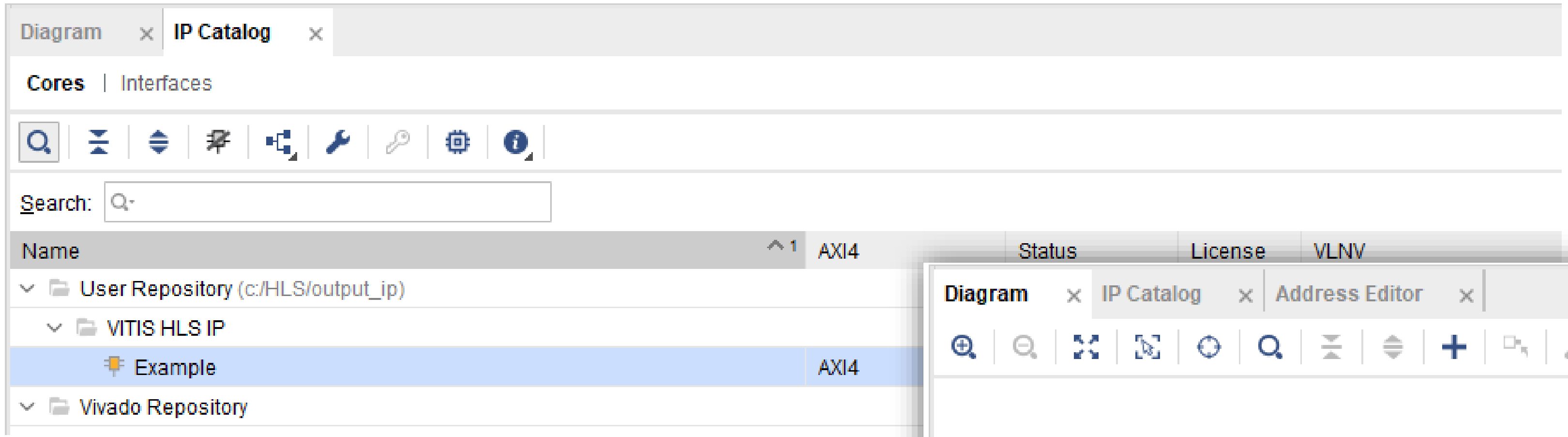


Diagram | IP Catalog

Cores | Interfaces

Search:

Name	AXI4	Status	License	VLNV
User Repository (c:/HLS/output_ip)				
VITIS HLS IP				
Example	AXI4			
Vivado Repository				

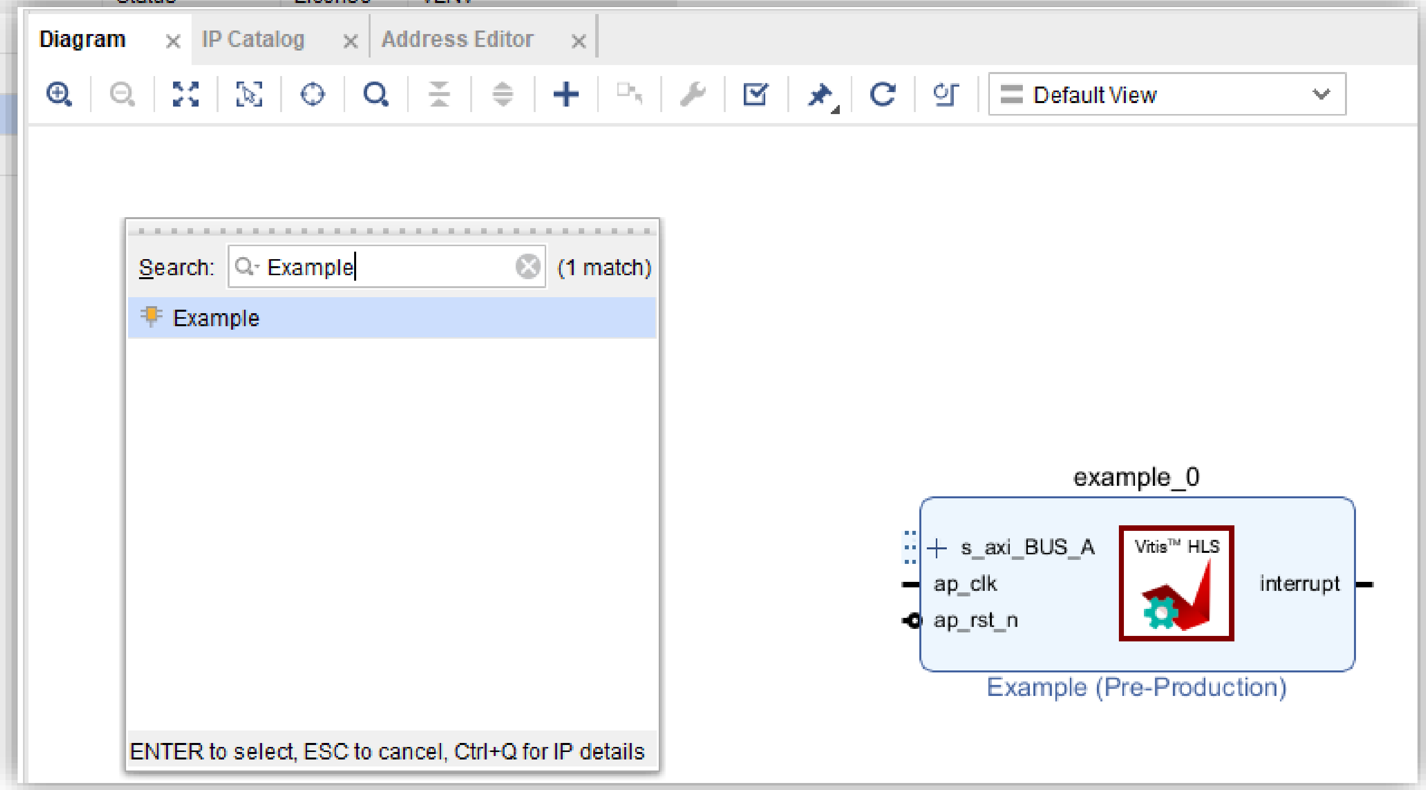


Diagram | IP Catalog | Address Editor

Default View

Search: Example (1 match)

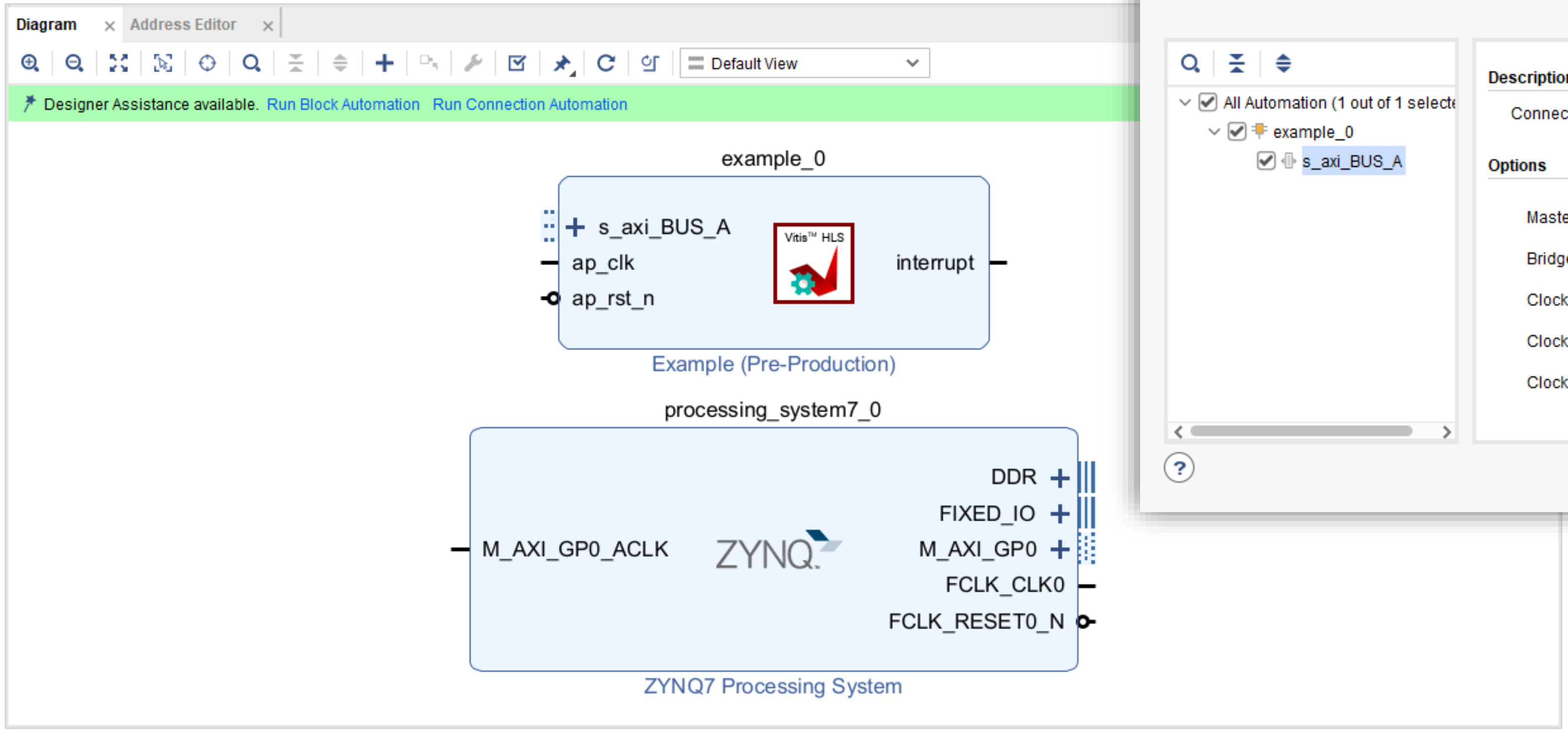
Example

ENTER to select, ESC to cancel, Ctrl+Q for IP details

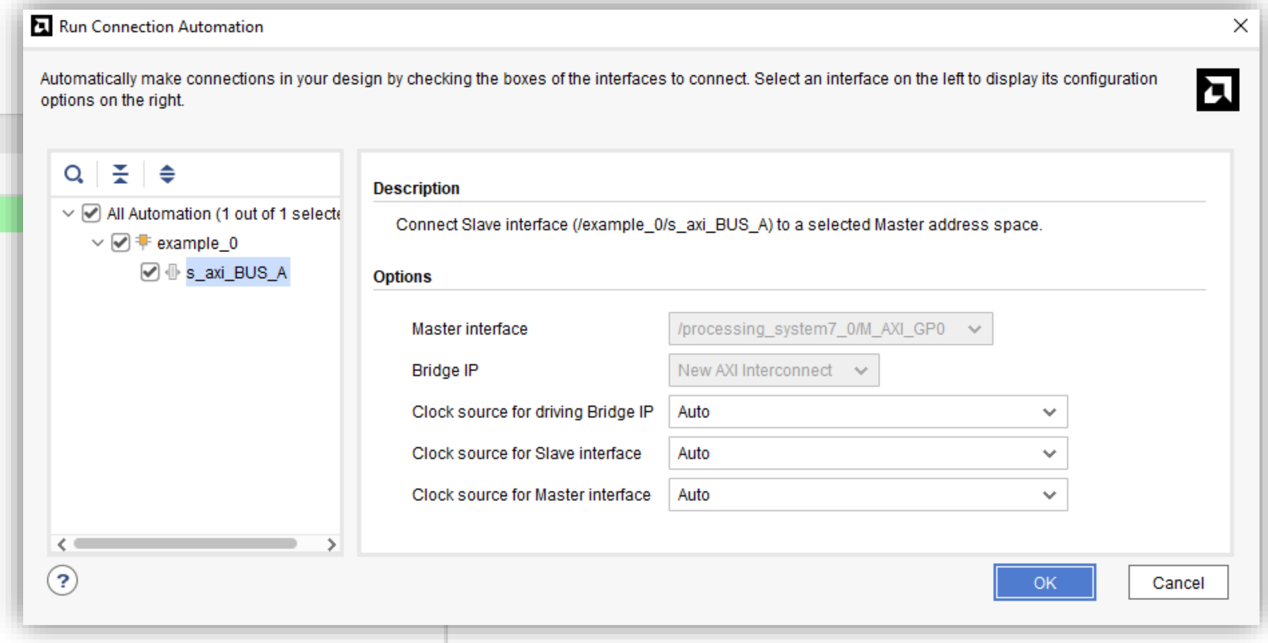
[Public]

Add Zynq Core IP

Add Zynq MPSoC and Run Block Automation & Run Connection Automation



Designer Assistance available. Run Block Automation 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.

Description

Connect Slave interface (/example_0/s_axi_BUS_A) to a selected Master address space.

Options

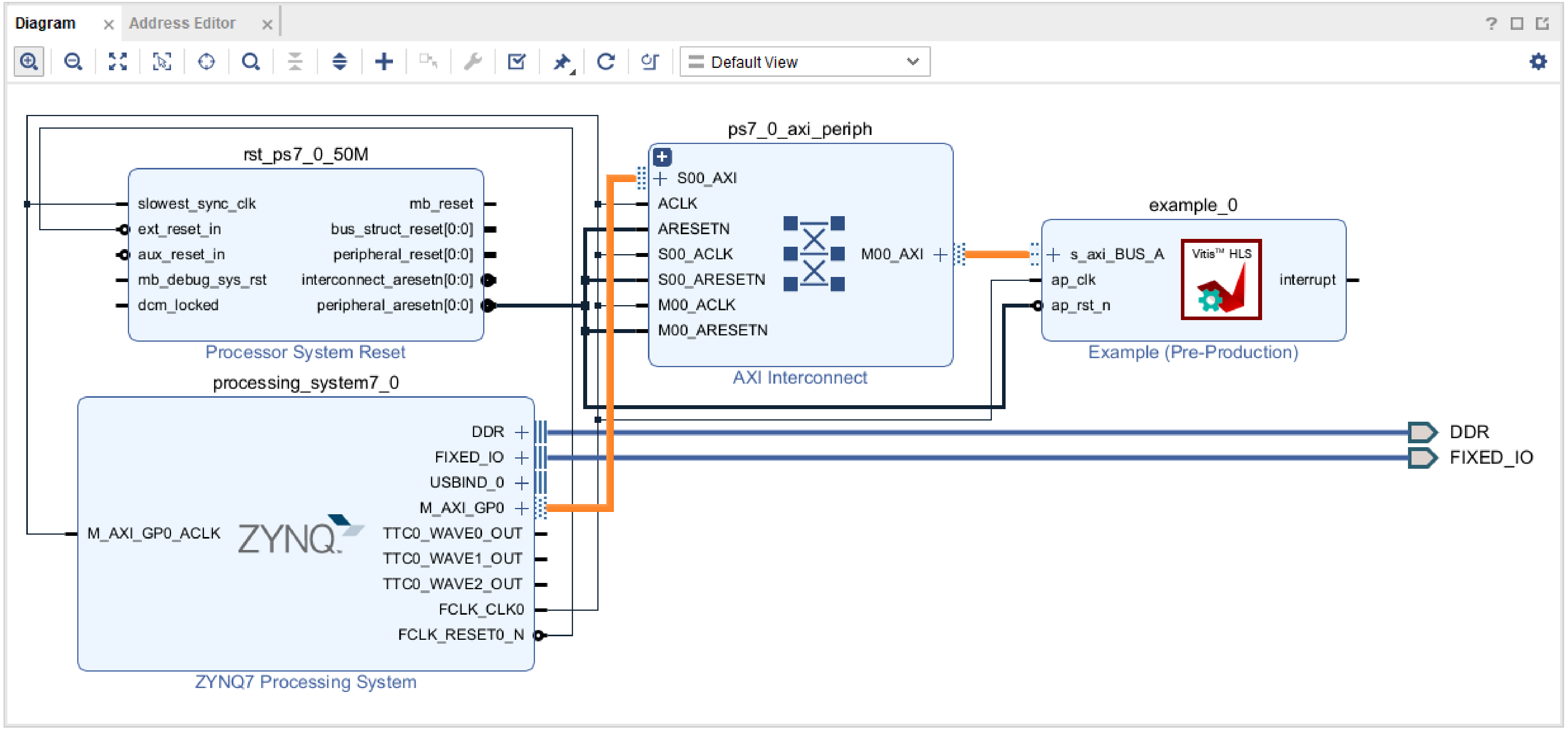
Master interface	/processing_system7_0/M_AXI_GP0
Bridge IP	New AXI Interconnect
Clock source for driving Bridge IP	Auto
Clock source for Slave interface	Auto
Clock source for Master interface	Auto

AMD
together we advance_

[Public]

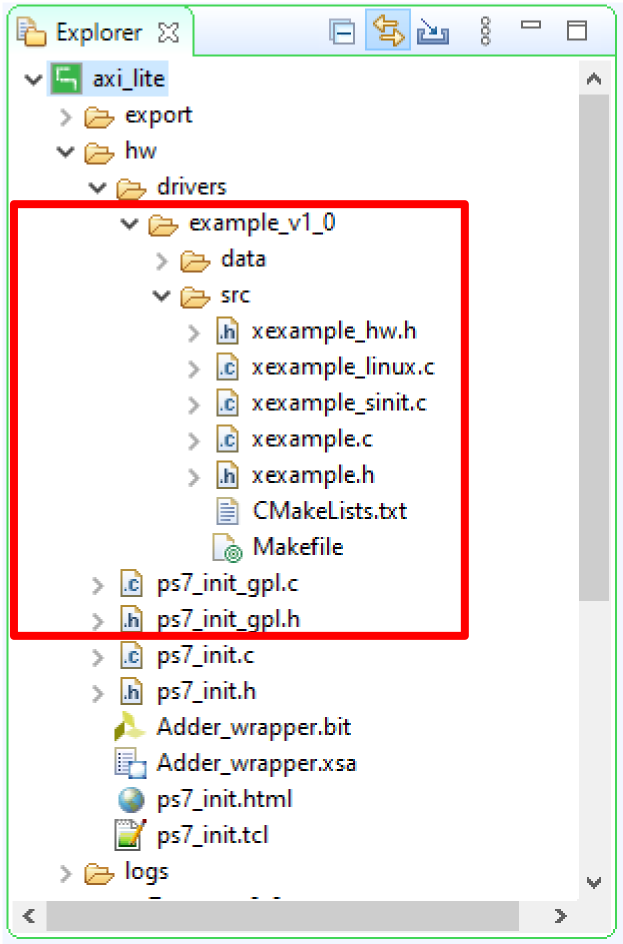
Final Block Design

- Create HDL Wrapper
- Generate Bitstream
- Export Xsa

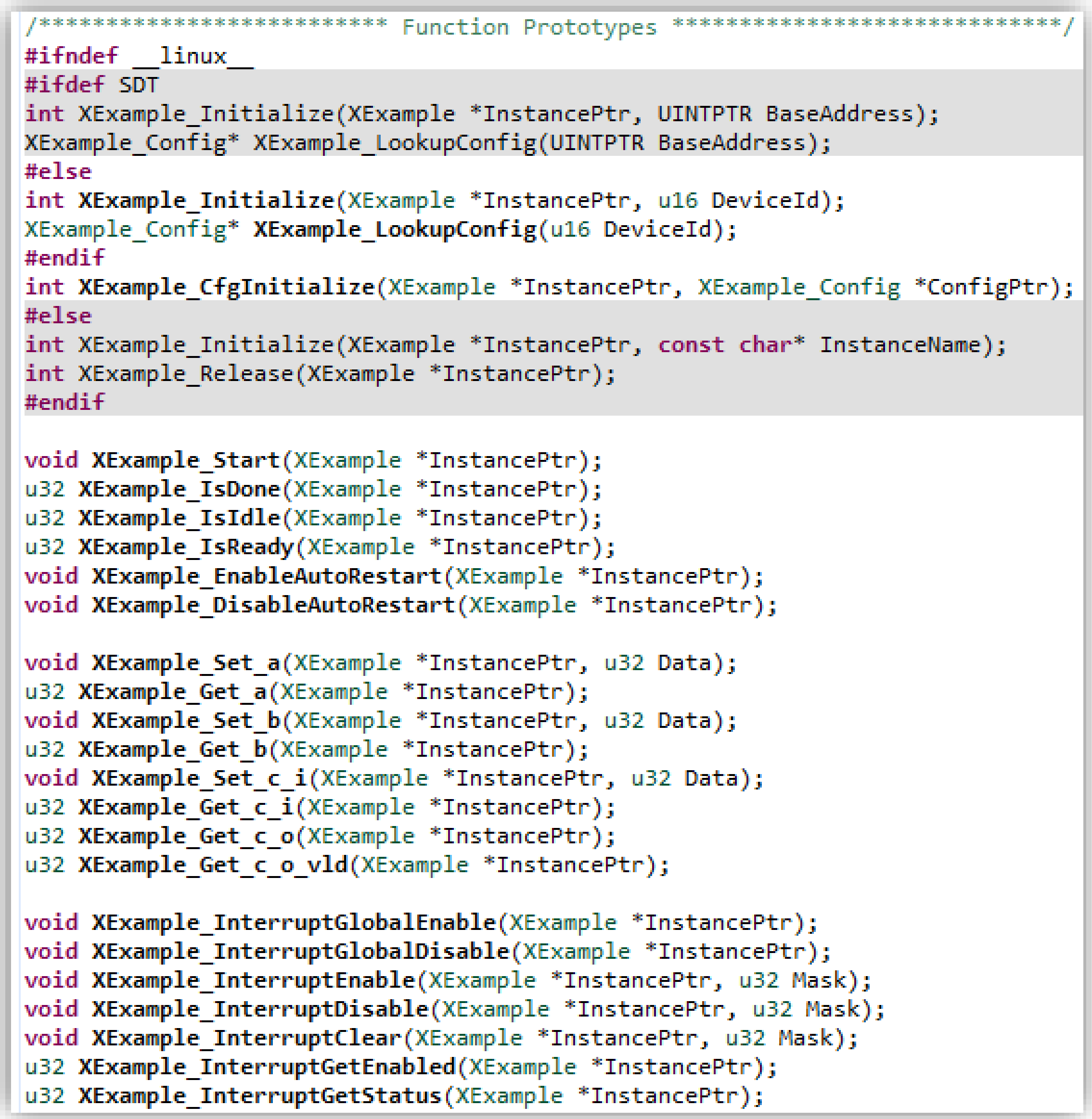


AMD
together we advance_

Vitis Driver API



```
Explorer
axi_lite
  export
  hw
    drivers
      example_v1_0
        data
        src
          xexample_hw.h
          xexample_linux.c
          xexample_sinit.c
          xexample.c
          xexample.h
          CMakeLists.txt
          Makefile
    ps7_init_gpl.c
    ps7_init_gpl.h
    ps7_init.c
    ps7_init.h
    Adder_wrapper.bit
    Adder_wrapper.xsa
    ps7_init.html
    ps7_init.tcl
  logs
```



```
/************************** Function Prototypes ****************************/
#ifndef __linux__
#ifdef SDT
int XExample_Initialize(XExample *InstancePtr, UINTPTR BaseAddress);
XExample_Config* XExample_LookupConfig(UINTPTR BaseAddress);
#else
int XExample_Initialize(XExample *InstancePtr, u16 DeviceId);
XExample_Config* XExample_LookupConfig(u16 DeviceId);
#endif
int XExample_CfgInitialize(XExample *InstancePtr, XExample_Config *ConfigPtr);
#else
int XExample_Initialize(XExample *InstancePtr, const char* InstanceName);
int XExample_Release(XExample *InstancePtr);
#endif

void XExample_Start(XExample *InstancePtr);
u32 XExample_IsDone(XExample *InstancePtr);
u32 XExample_IsIdle(XExample *InstancePtr);
u32 XExample_IsReady(XExample *InstancePtr);
void XExample_EnableAutoRestart(XExample *InstancePtr);
void XExample_DisableAutoRestart(XExample *InstancePtr);

void XExample_Set_a(XExample *InstancePtr, u32 Data);
u32 XExample_Get_a(XExample *InstancePtr);
void XExample_Set_b(XExample *InstancePtr, u32 Data);
u32 XExample_Get_b(XExample *InstancePtr);
void XExample_Set_c_i(XExample *InstancePtr, u32 Data);
u32 XExample_Get_c_i(XExample *InstancePtr);
u32 XExample_Get_c_o(XExample *InstancePtr);
u32 XExample_Get_c_o_vld(XExample *InstancePtr);

void XExample_InterruptGlobalEnable(XExample *InstancePtr);
void XExample_InterruptGlobalDisable(XExample *InstancePtr);
void XExample_InterruptEnable(XExample *InstancePtr, u32 Mask);
void XExample_InterruptDisable(XExample *InstancePtr, u32 Mask);
void XExample_InterruptClear(XExample *InstancePtr, u32 Mask);
u32 XExample_InterruptGetEnabled(XExample *InstancePtr);
u32 XExample_InterruptGetStatus(XExample *InstancePtr);
```

Vitis test code

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

//Define Adder ID
#define Adder_ID XPAR_XEXAMPLE_0_DEVICE_ID

//Define global values for HLS example IP
XExample Adder;
XExample_Config *Adder_cfg;
```

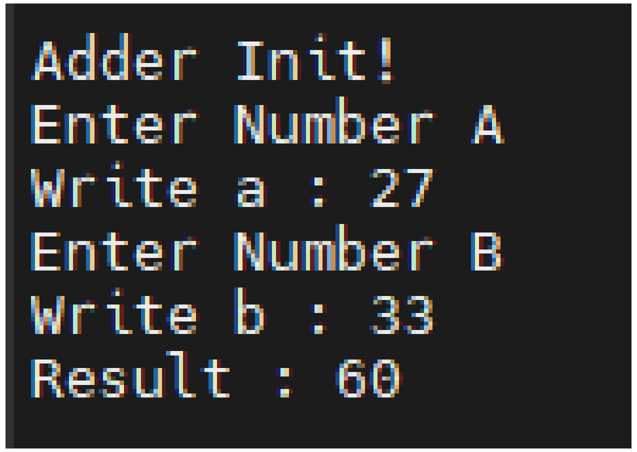
Vitis test code

```
int main(){

    init_platform();
    init_Adder();
    xil_printf("Adder Init!\n\r");

    int a = 0;
    int b = 0;

    Adder_HLS(a,b);

    cleanup_platform();
    return 0;
}
```

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

 int Status;

 Adder_cfg = XExample_LookupConfig(Adder_ID);
 Status = XExample_CfgInitialize(&Adder, Adder_cfg);
 if (Status != XST_SUCCESS) {
       xil_printf("Init Fail.");
 return XST_FAILURE;
 }
}
```

Vitis test code

```
void Adder_HLS(int a,int b){

 unsigned int c;
 c = 0;

 xil_printf("Enter Number A\n\r");
 scanf("%d",&a);
 XExample_Set_a(&Adder, a);
 xil_printf("Write a : %d \n\r",a);
 xil_printf("Enter Number B\n\r");
 scanf("%d",&b);
 XExample_Set_b(&Adder, b);
 xil_printf("Write b : %d \n\r",b);

 XExample_Start(&Adder);

 while(!XExample_IsDone(&Adder));
 c = XExample_Get_c_o(&Adder);
 xil_printf("Result : %d \n\r",c);
}
```



```
Adder Init!
Enter Number A
Write a : 27
Enter Number B
Write b : 33
Result : 60
```