FPGA HLS Design Flow Lab

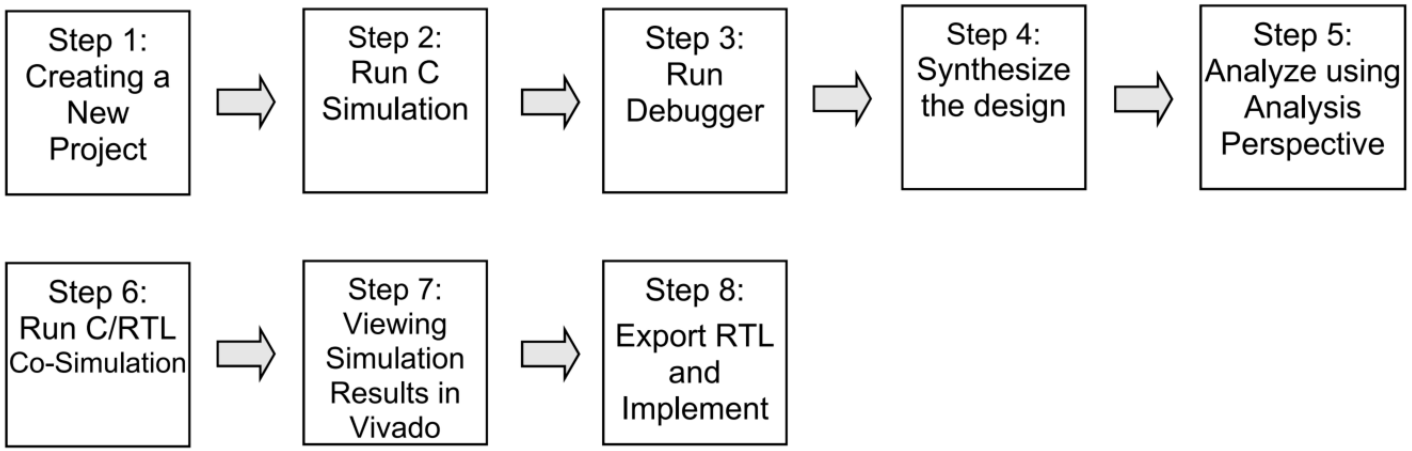
With Vivado

24th April 2023

Judong Park

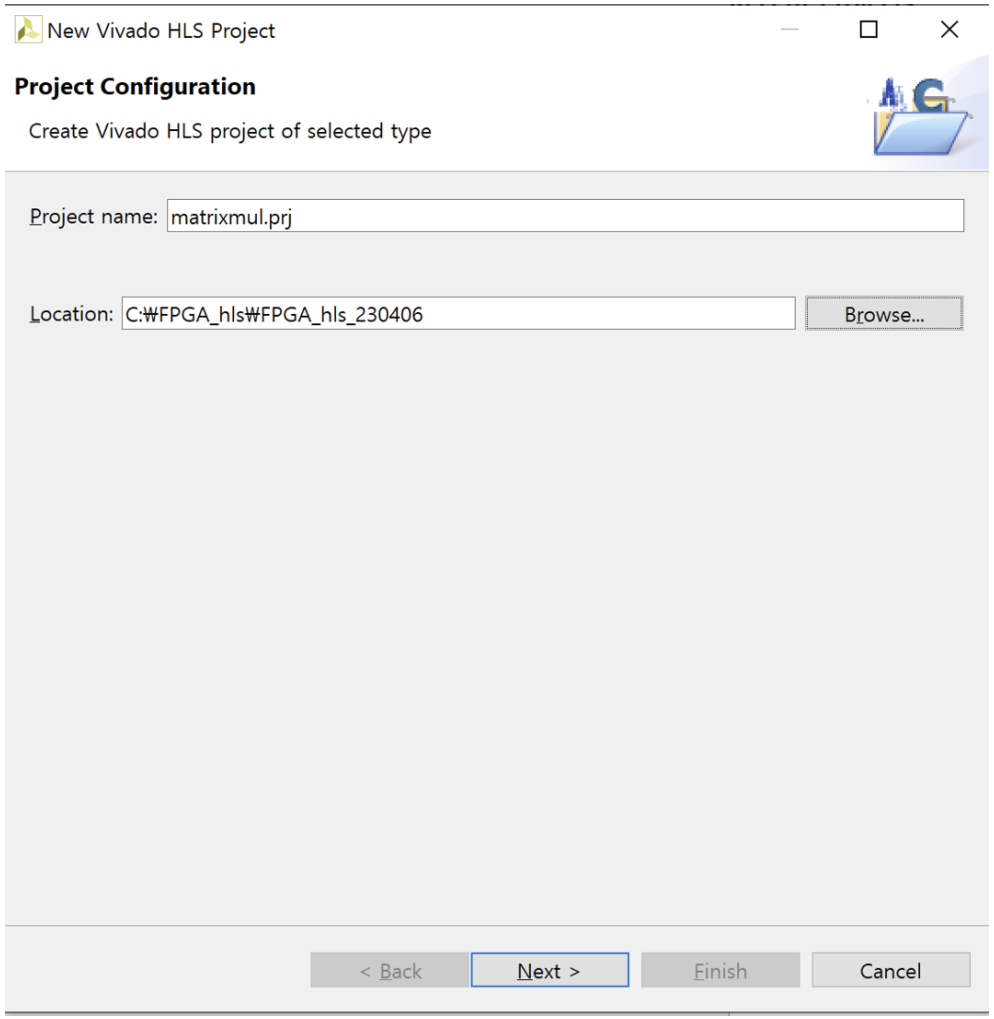
This LAB…

- 기본적인 비바도 HLS(High-level synthesis) tool flow
- 학습 목표:
  - Create a new project using Vivado HLS GUI
  - Simulate a design
  - Synthesize a design
  - Implement a design
  - Perform design analysis using the Analysis capability of Vivado HLS
  - Analyze simulator output using Vivado and XSim simulator



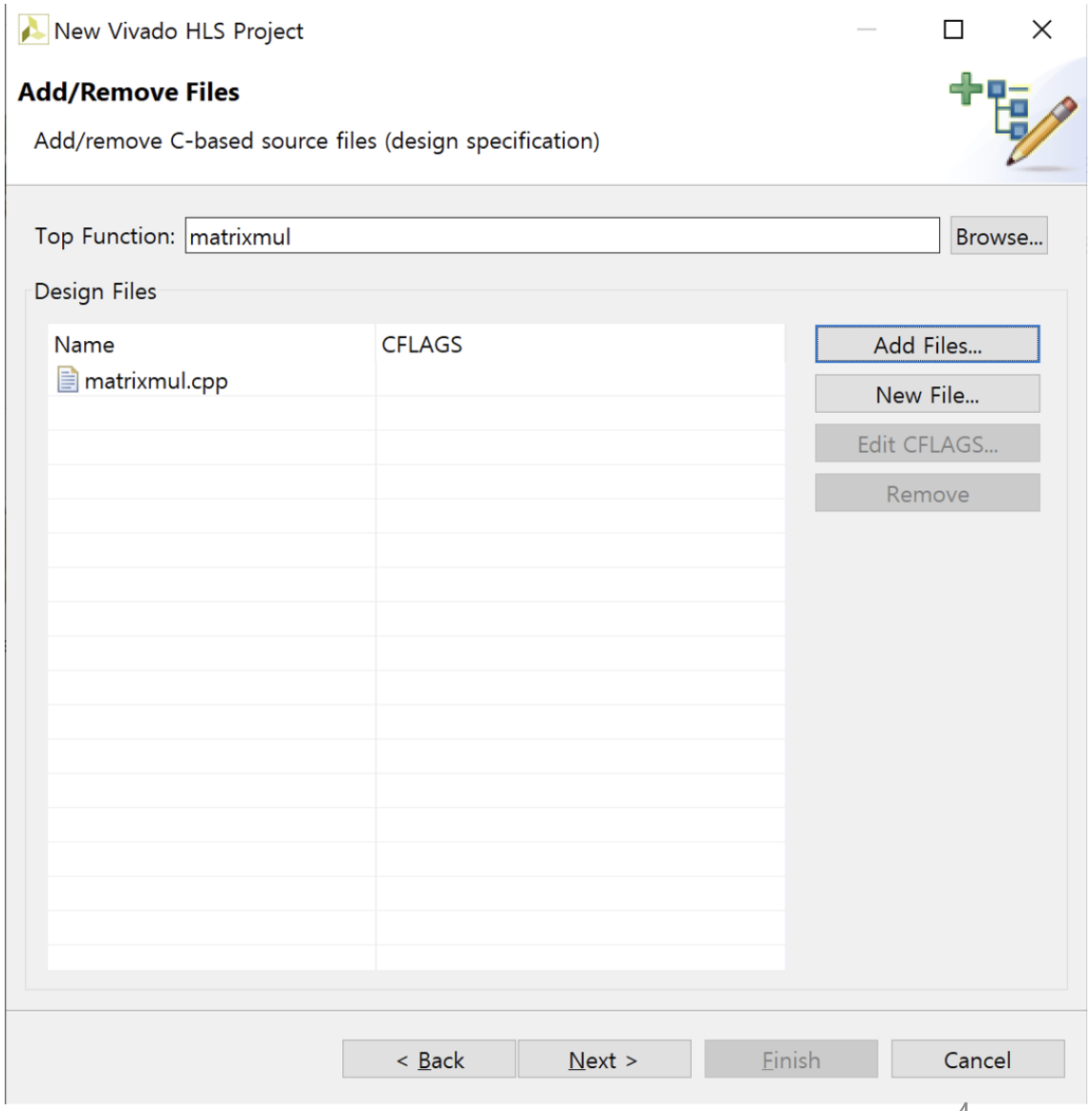
Create a New Project

- HLS 프로젝트 생성
- Create New Project 클릭
- Project name: **matrixmul.prj**
- Location: **원하는 위치**
- Next



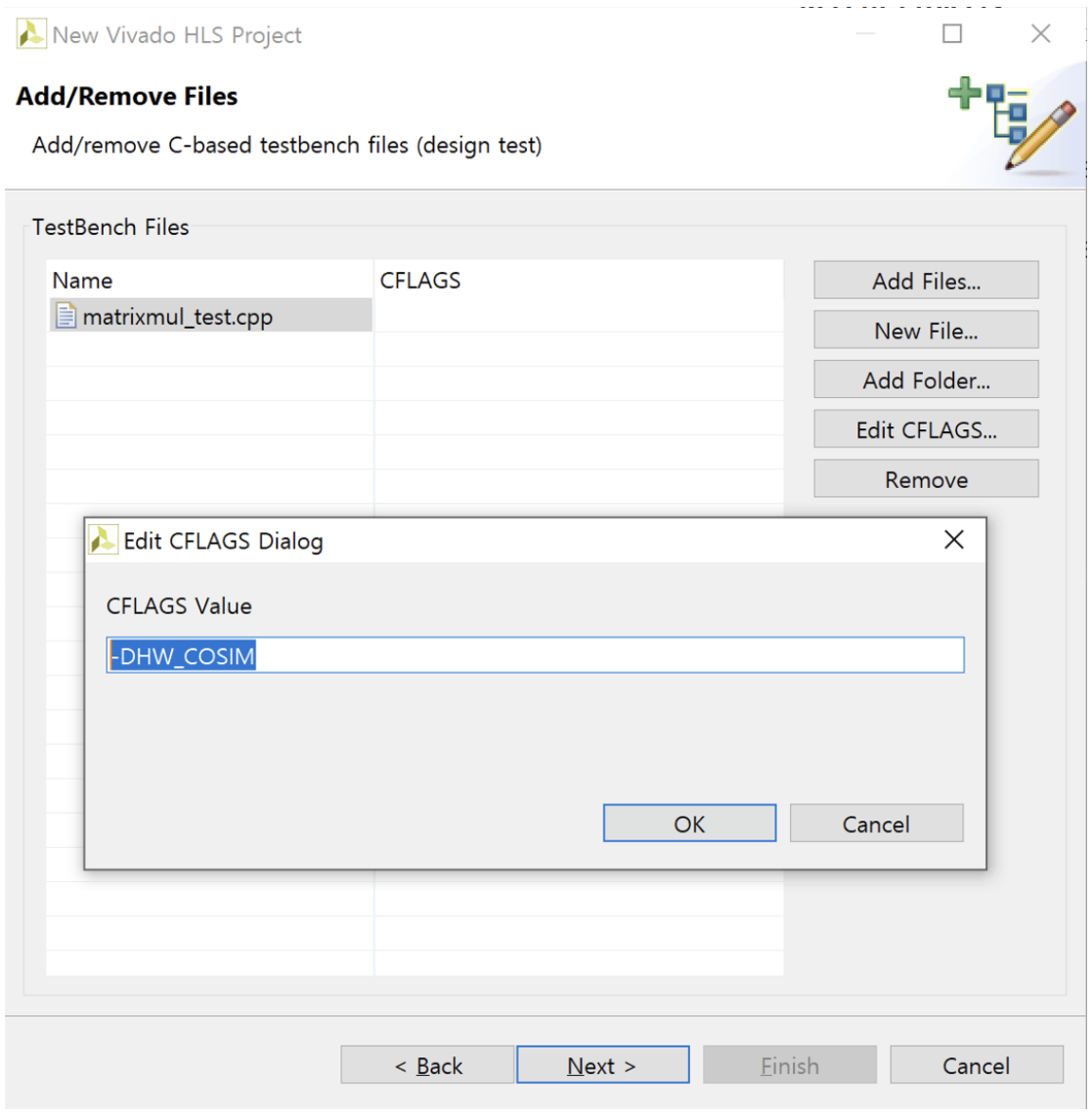
Create a New Project

- In Add/Remove Files window,
- **Top Function: matrixmul**
- Add Files: **matrixmul.cpp**
- Next



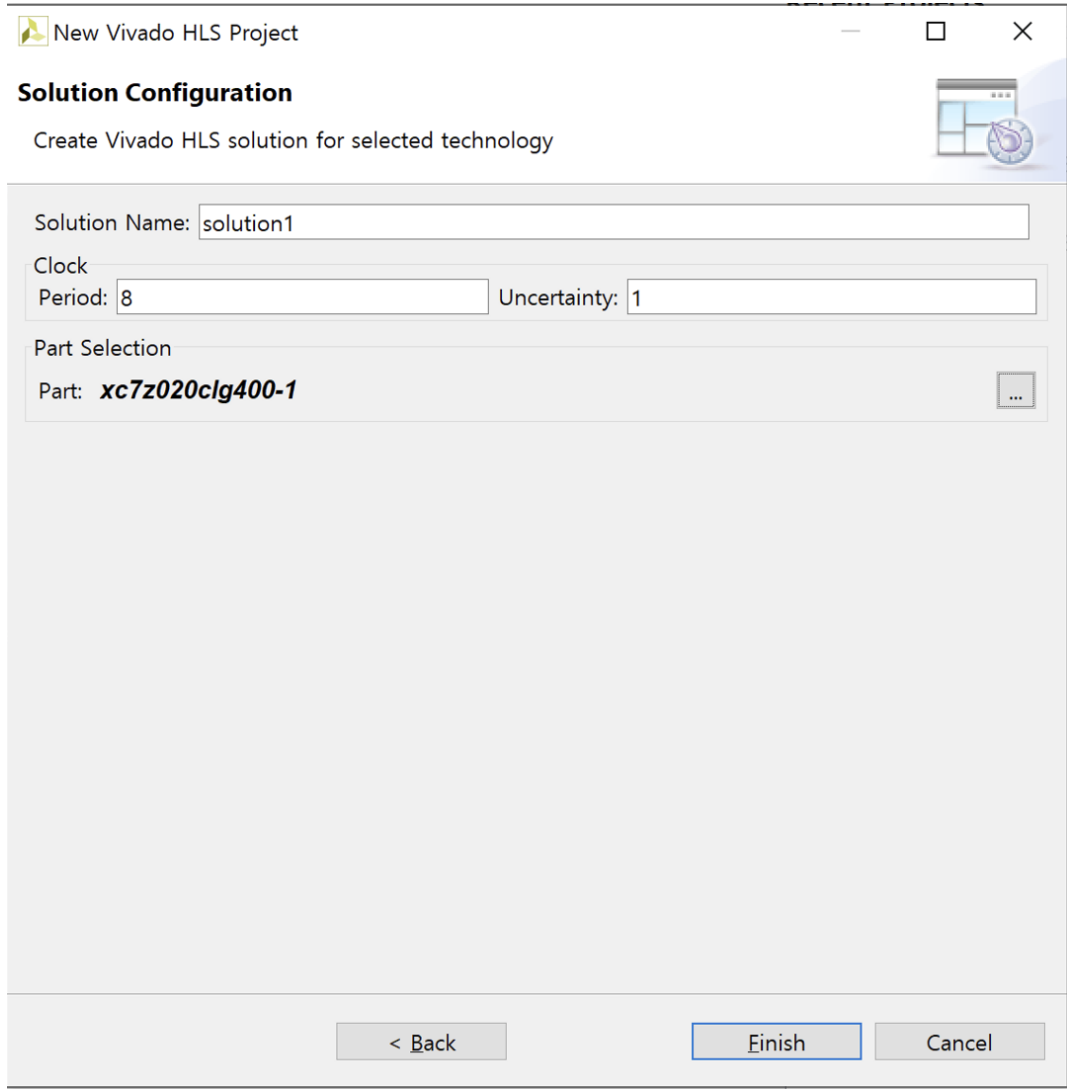
Create a New Project

- In Add/Remove Files window,
- Add Files: **matrixmul_test.cpp**
- Edit CFLAGS: **-DHW_COSIM**
- Next



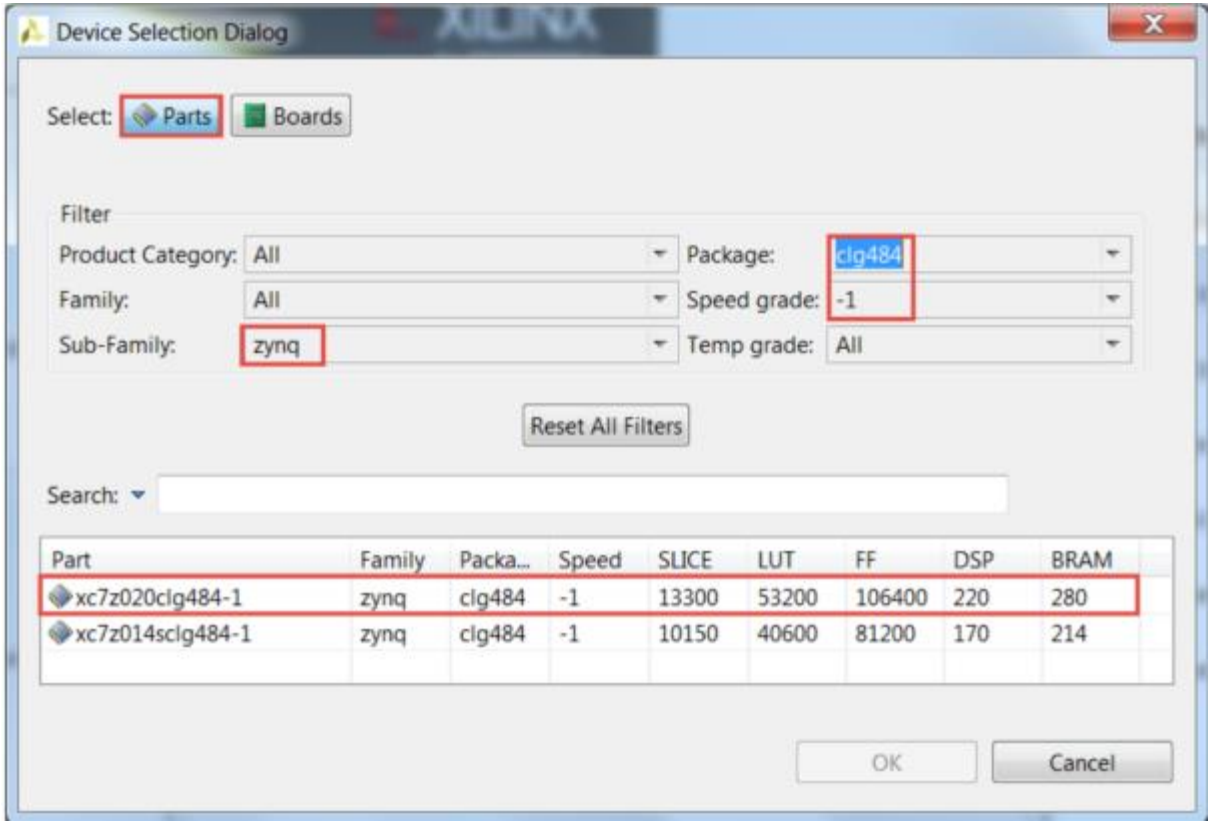
Create a New Project

- In Solution Configuration,
- Solution Name: **solution1**
- Period: **8 (Zybo), 10 (ZedBoard)**
- Uncertainty: **1 (Zybo), 1.25 (ZedBoard)**
- Finish



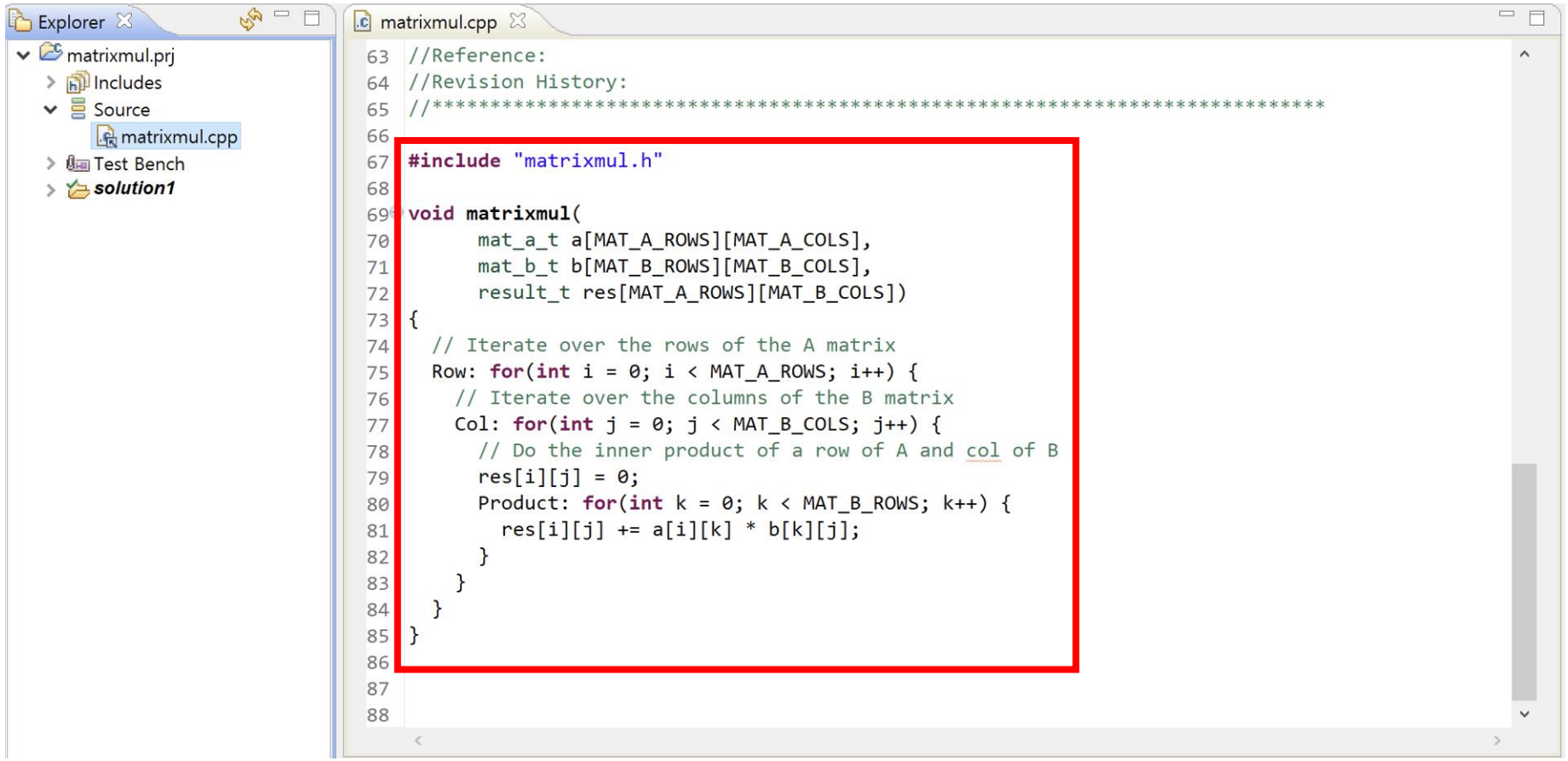
Create a New Project

- In Device Selection Dialog,
- Zedboard: **xc7z020clg484-1**
- Zybo: **xc7z010clg400-1**



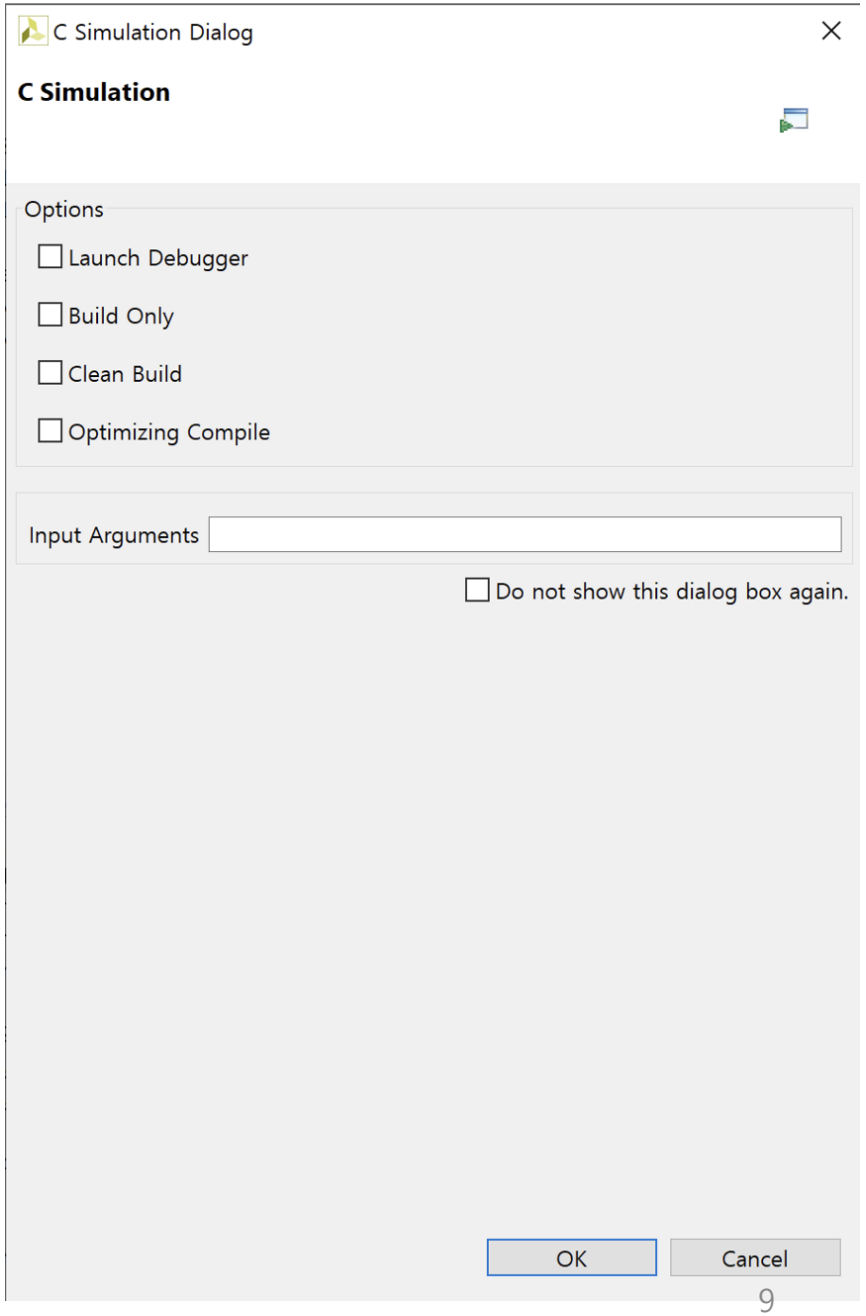
Create a New Project

- Explorer – Source – **matrixmul.cpp** 확인



Run C Simulation

- Project – Run C Simulation 클릭 
- OK 클릭



Run C Simulation

- Console window – 결과 확인

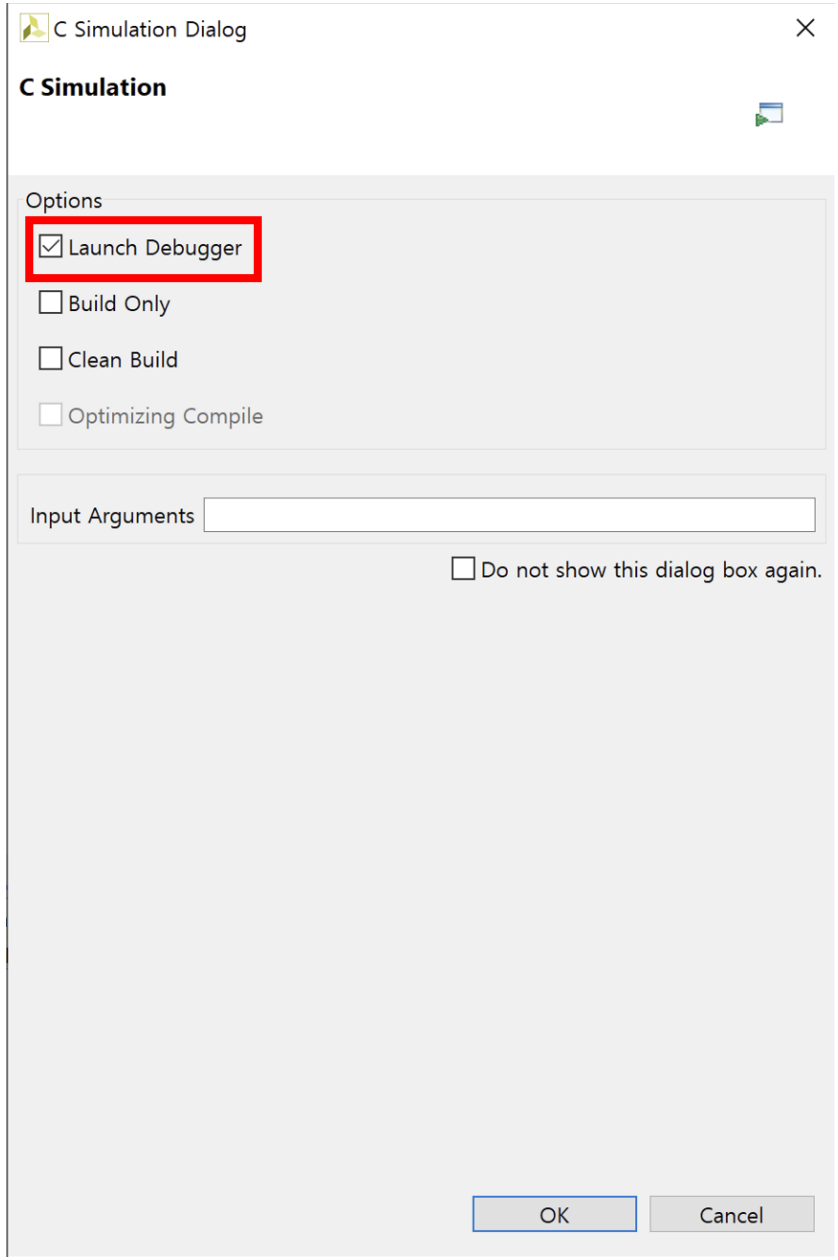
Console | Errors | Warnings | Progress | Debugger Console

Vivado HLS Console

```
Starting C simulation ...
C:/Xilinx/Vivado/2018.2/bin/vivado_hls.bat C:/FPGA_hls/FPGA_hls_230406/matrixmul.prj/solution1/csim.tcl
INFO: [HLS 200-10] Running 'C:/Xilinx/Vivado/2018.2/bin/unwrapped/win64.o/vivado_hls.exe'
INFO: [HLS 200-10] For user 'Judong' on host 'desktop-g5m57lb' (Windows NT_amd64 version 6.2) on Thu Apr 06 01:36:10 +0900 2023
INFO: [HLS 200-10] In directory 'C:/FPGA_hls/FPGA_hls_230406'
INFO: [HLS 200-10] Opening project 'C:/FPGA_hls/FPGA_hls_230406/matrixmul.prj'.
INFO: [HLS 200-10] Opening solution 'C:/FPGA_hls/FPGA_hls_230406/matrixmul.prj/solution1'.
INFO: [SYN 201-201] Setting up clock 'default' with a period of 8ns.
INFO: [SYN 201-201] Setting up clock 'default' with an uncertainty of 1ns.
INFO: [HLS 200-10] Setting target device to 'xc7z020clg400-1'
INFO: [SIM 211-2] *************** CSIM start ***************
INFO: [SIM 211-4] CSIM will launch GCC as the compiler.
make: 'csim.exe' is up to date.
{
{870,906,942}
{1086,1131,1176}
{1302,1356,1410}
}
Test passed.
INFO: [SIM 211-1] CSim done with 0 errors.
INFO: [SIM 211-3] *************** CSIM finish ***************
Finished C simulation.
```

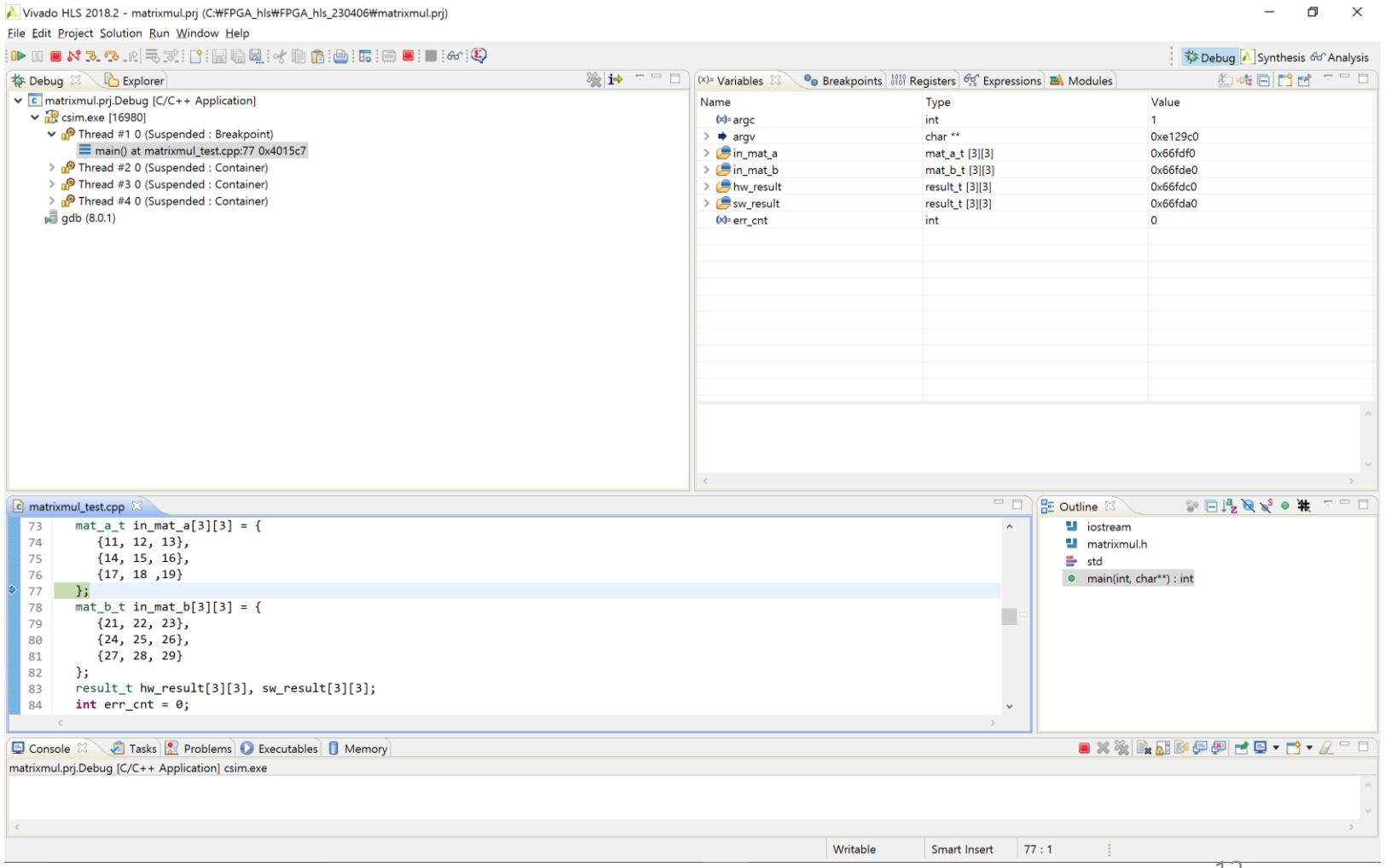
Run Debugger

- Project – Run C Simulation 클릭 
- Launch Debugger 체크
- OK 클릭



Run Debugger

- C 코드 디버깅을 위한 창



Run Debugger

- Matrixmul_test.cpp – **105, 101번째 blue line** 더블 클릭

➔ **Breakpoint 생성**

matrixmul_test.cpp

```
97      }
98
99  #ifdef HW_COSIM
100     // Run the AutoESL matrix multiply block
101     matrixmul(in_mat_a, in_mat_b, hw_result);
102 #endif
103
104     // Print result matrix
105     cout << "{" << endl;
106     //cout << setw(6);
107     for (int i = 0; i < MAT_A_ROWS; i++) {
108        cout << "{";
```

matrixmul_test.cpp

```
97      }
98
99  #ifdef HW_COSIM
100     // Run the AutoESL matrix multiply block
101     matrixmul(in_mat_a, in_mat_b, hw_result);
102 #endif
103
104     // Print result matrix
105     cout << "{" << endl;
106     //cout << setw(6);
107     for (int i = 0; i < MAT_A_ROWS; i++) {
108        cout << "{";
```

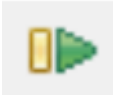
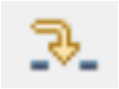
Run Debugger

- Step Over (F6)  여러 번 클릭

➔SW 연산 결과 확인 가능

sw_result	result_t [3][3]	0x66fda0
sw_result[0]	result_t [3]	0x66fda0
sw_result[0][0]	result_t	870
sw_result[0][1]	result_t	906
sw_result[0][2]	result_t	942
sw_result[1]	result_t [3]	0x66fda6
sw_result[1][0]	result_t	1086
sw_result[1][1]	result_t	1131
sw_result[1][2]	result_t	1176
sw_result[2]	result_t [3]	0x66fdac
sw_result[2][0]	result_t	1302
sw_result[2][1]	result_t	1356
sw_result[2][2]	result_t	1410
err_cnt	int	0

Run Debugger

- Resume (F8)  클릭 ➔ 101번째 라인 전까지 코드 실행
- Step Into (F5)  클릭 ➔ matrixmul.cpp 디버깅 시작

matrixmul_test.cpp | matrixmul.cpp

```
65 //*****************************************************************
66
67 #include "matrixmul.h"
68
69 void matrixmul(
70       mat_a_t a[MAT_A_ROWS][MAT_A_COLS],
71       mat_b_t b[MAT_B_ROWS][MAT_B_COLS],
72       result_t res[MAT_A_ROWS][MAT_B_COLS])
73 {
74   // Iterate over the rows of the A matrix
75   Row: for(int i = 0; i < MAT_A_ROWS; i++) {
76     // Iterate over the columns of the B matrix
77     Col: for(int j = 0; j < MAT_B_COLS; j++) {
78       // Do the inner product of a row of A and col of B
79       res[i][j] = 0;
```

Run Debugger

- Step Over (F6)  여러 번 클릭 ➔ 연산 결과 확인
- Step Return (F7)  클릭 ➔ 다시 test로 돌아옴
- 연산 결과가 동일함을 확인

hw_result	result_t [3][3]	6749632 (Decimal)
hw_result[0]	result_t [3]	6749632 (Decimal)
hw_result[0][0]	result_t	870
hw_result[0][1]	result_t	906
hw_result[0][2]	result_t	942
hw_result[1]	result_t [3]	0x66fdc6
hw_result[1][0]	result_t	1086
hw_result[1][1]	result_t	1131
hw_result[1][2]	result_t	1176
hw_result[2]	result_t [3]	0x66fdcc
hw_result[2][0]	result_t	1302
hw_result[2][1]	result_t	1356
hw_result[2][2]	result_t	1410
sw_result	result_t [3][3]	0x66fda0
sw_result[0]	result_t [3]	0x66fda0
sw_result[0][0]	result_t	870
sw_result[0][1]	result_t	906
sw_result[0][2]	result_t	942
sw_result[1]	result_t [3]	0x66fda6
sw_result[1][0]	result_t	1086
sw_result[1][1]	result_t	1131
sw_result[1][2]	result_t	1176
sw_result[2]	result_t [3]	0x66fdac
sw_result[2][0]	result_t	1302
sw_result[2][1]	result_t	1356
sw_result[2][2]	result_t	1410
err_cnt	int	0 (Decimal)

Run Debugger

- Matrixmul_test. – **134번째 blue line** 더블 클릭

➔ **Breakpoint 생성**

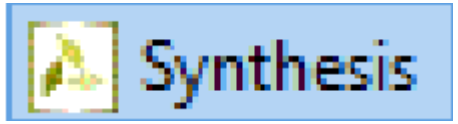
- **Resume** 클릭
- 다시 **Resume** 클릭

➔ 디버깅 종료

matrixmul_test.cpp | matrixmul.cpp

```
124         }
125      }
126      cout << "}" << endl;
127
128   #ifdef HW_COSIM
129      if (err_cnt)
130         cout << "ERROR: " << err_cnt << " mismatches detected!" << endl;
131      else
132         cout << "Test passed." << endl;
133   #endif
134      return err_cnt;
135   }
136
137
```

Synthesize the Design

- **Synthesis 버튼** 

클릭
- **Solution - Run C Synthesis - Active Solution**  클릭

➔ 합성 및 결과 확인 가능

Performance Estimates

Timing (ns)

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00	7.305	1.25

Latency (clock cycles)

Summary

Latency		Interval		
min	max	min	max	Type
79	79	79	79	none

Detail

Instance

Loop

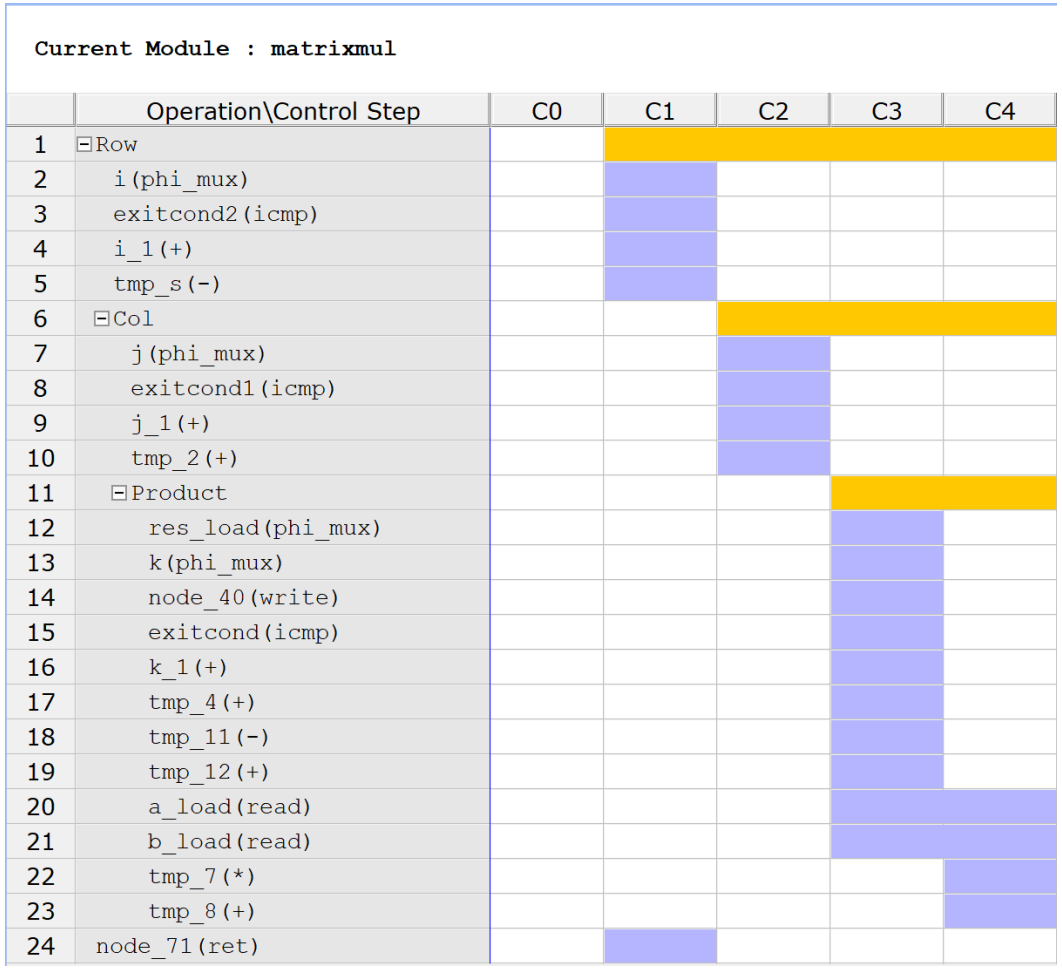
Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	1	-	-
Expression	-	-	0	115
FIFO	-	-	-	-
Instance	-	-	-	-
Memory	-	-	-	-
Multiplexer	-	-	-	69
Register	-	-	44	-
Total	0	1	44	184
Available	280	220	106400	53200
Utilization (%)	0	~0	~0	~0

Analyze using Analysis Perspective

- **Solution - Open Analysis Perspective or** Analysis 클릭
- Matrixml 오른쪽 클릭 – **open performance/resource viewer** 클릭
- **Row, col, product + 클릭**



Analyze using Analysis Perspective

- **C1 – adder 보라색 칸 오른쪽 버튼 클릭 – goto source**

➔해당 코드 확인 가능

File: C:₩Users₩Judong₩Desktop₩fpga_2023₩fpga_hls₩labs₩lab1₩matrixmul.cpp

```
75  Row: for(int i = 0; i < MAT_A_ROWS; i++) {
76    // Iterate over the columns of the B matrix
77    Col: for(int j = 0; j < MAT_B_COLS; j++) {
78      // Do the inner product of a row of A and col of B
79      res[i][j] = 0;
80      Product: for(int k = 0; k < MAT_B_ROWS; k++) {
81        res[i][j] += a[i][k] * b[k][j];
```

Analyze using Analysis Perspective

- **Resource Profile 클릭**

➔ Design 합성에 사용되는 resource 확인 가능

Performance Profile | Resource Profile

	BRAM	FF	LUT	Bits P0	Bits P1
matrixmul	0	44	184		
I/O Ports(3)				32	
Instances(0)	0	0	0		
Memories(0)	0	0	0	0	
Expressions(11)	0	0	115	37	34
Registers(11)		44		47	
Channels(0)	0	0	0	0	
Multiplexers(5)	0	0	69	23	

Analyze using Analysis Perspective

- **Performance Matrix – Resource 클릭**

➔ 어떤 단계에서
➔ 어떤 리소스가
➔ 어떤 연산을 하는지 확인 가능

- **Synthesis view 클릭**

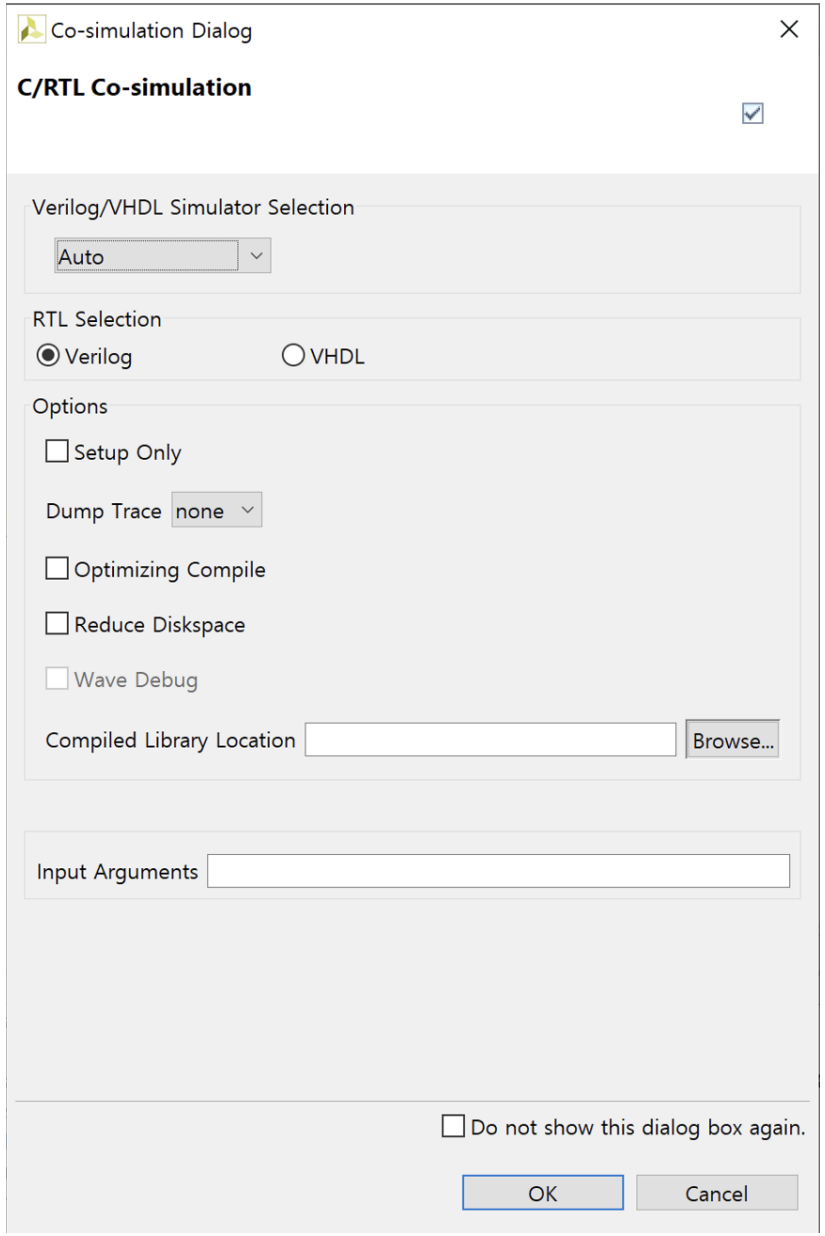
matrixmul_test.cpp | matrixmul.cpp | Synthesis(solution1) | Schedule Viewer(soluti

Current Module : matrixmul

	Resource\Control Step	C0	C1	C2	C3	C4
1	⊟I/O Ports					
2	b(p0)				read	
3	a(p0)				read	
4	res(p0)				write	
5	⊟Memory Ports					
6	b(p0)				read	
7	res(p0)				write	
8	a(p0)				read	
9	⊟Expressions					
10	i_phi_fu_82		phi_mux			
11	i_1_fu_130		+			
12	tmp_s_fu_152		-			
13	exitcond2_fu_124		icmp			
14	j_phi_fu_93			phi_mux		
15	j_1_fu_164			+		
16	tmp_2_fu_174			+		
17	exitcond1_fu_158			icmp		
18	k_phi_fu_117				phi_mux	
19	res_load_phi_fu_104				phi_mux	
20	tmp_12_fu_228				+	
21	tmp_4_fu_200				+	
22	k_1_fu_190				+	
23	tmp_11_fu_222				-	
24	exitcond_fu_184				icmp	

Run C/RTL Co-simulation

- Solution - Run C/RTL Cosimulation 클릭
- RTL Selection: Verilog 선택
- OK 클릭



Run C/RTL Co-simulation

- **Simulation verification 결과**
  - **Latency, interval 확인 가능**

Cosimulation Report for 'matrixmul'

Result

		Latency			Interval		
RTL	Status	min	avg	max	min	avg	max
VHDL	NA	NA	NA	NA	NA	NA	NA
Verilog	Pass	79	79	79	NA	NA	NA

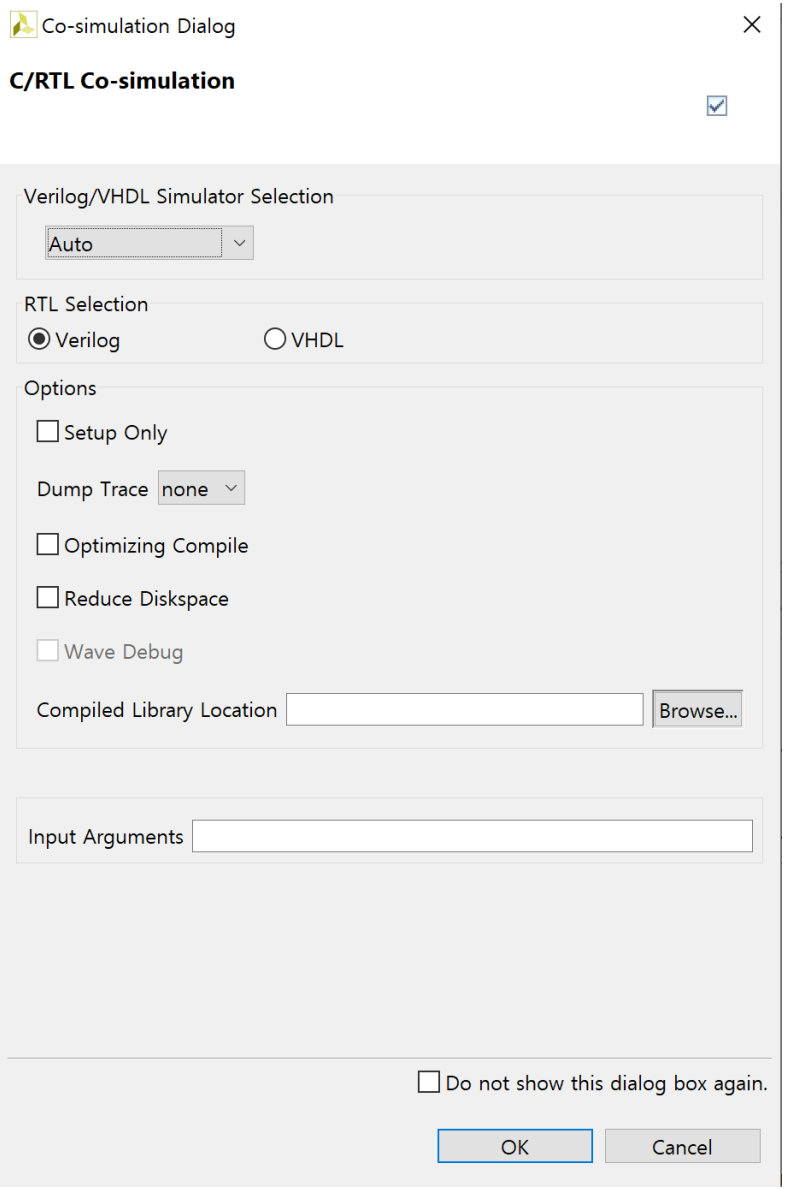
Export the report(.html) using the Export Wizard

Viewing Simulation Results in Vivado

- Solution > Run C/RTL Co-simulation or  클릭

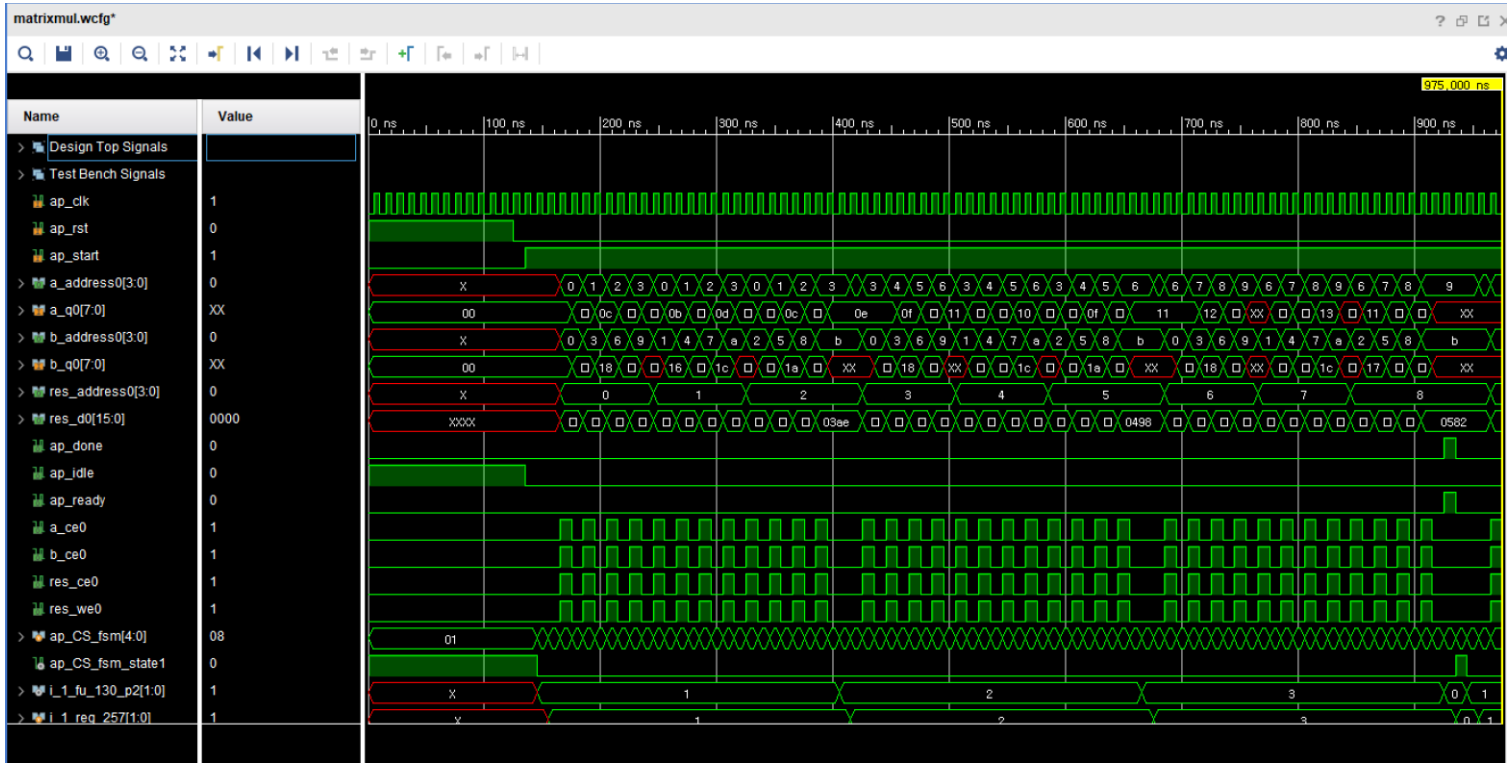
- Verilog/VHDL Simulator Section: Auto
- RTL selection: Verilog
- Dump trace: All

- Ok 클릭



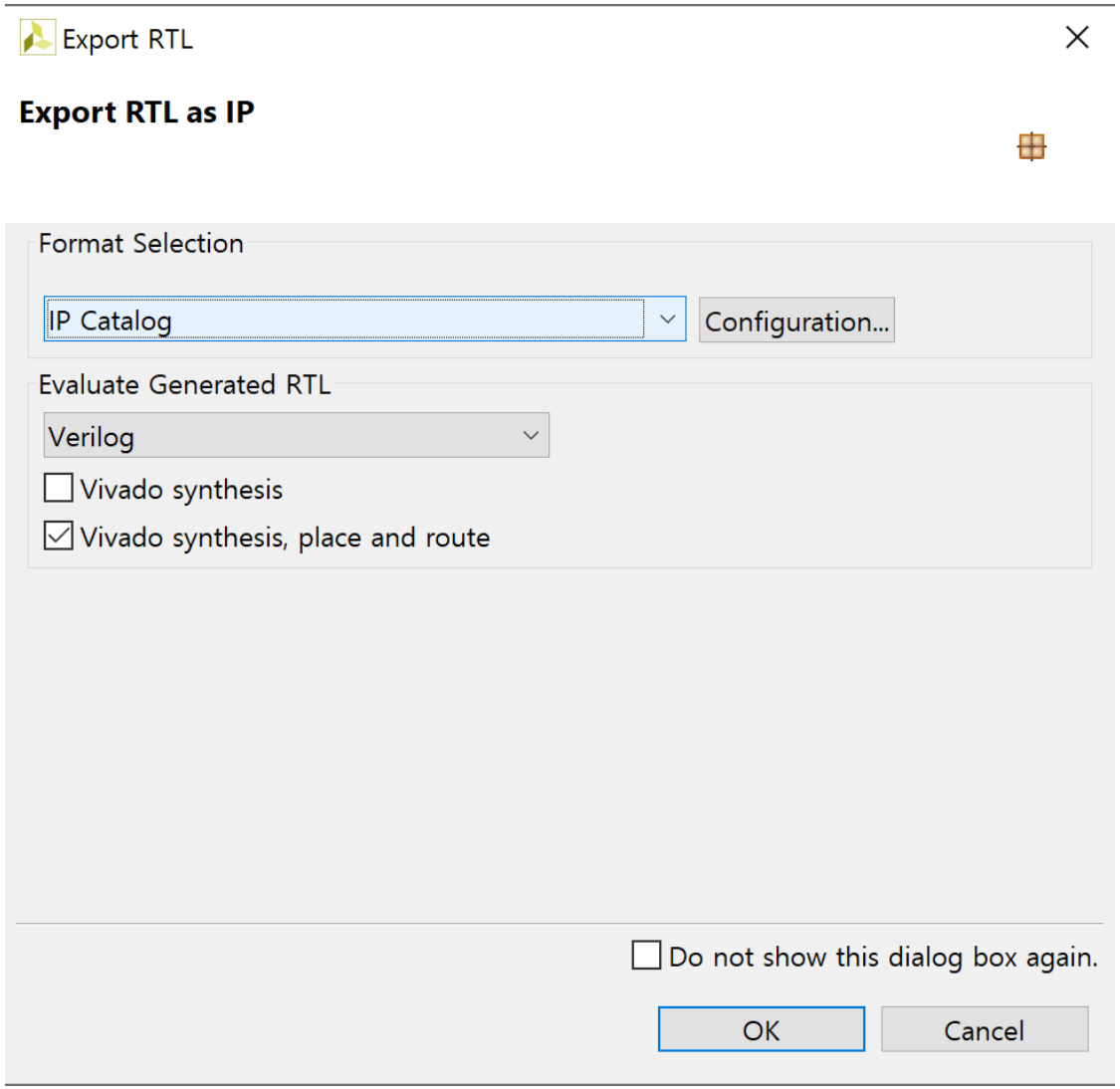
Viewing Simulation Results in Vivado

- wave viewer 클릭 ➔ 시뮬레이션 결과 확인 가능
- a/b/res_address0: radix - unsigned decimal
- A/b/res_q0: redix – signed decimal



Export RTL and Implement

- Solution > Export RTL  클릭
- Evaluate Generated RTL: Verilog
- Vivado synthesis, place and route 체크
- Ok 클릭



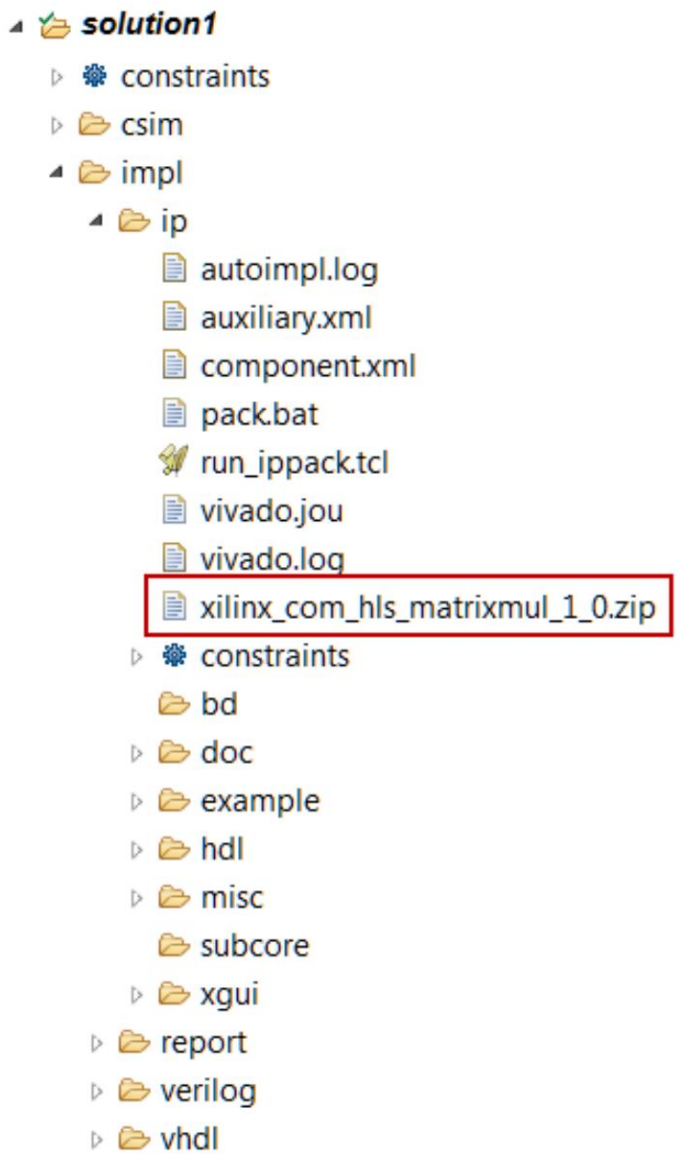
Export RTL and Implement

- Solution1 - Impl - IP – IP.zip 파일 확인 가능

- Verilog 폴더 확장
  - .xdc
  - .xpr

- Project.runs 폴더 확장
  - 합성, P&R 단계 파일

- Report 폴더
  - RTL timing/utilization



감사합니다!

- **Q&A**