Improving Performance Lab

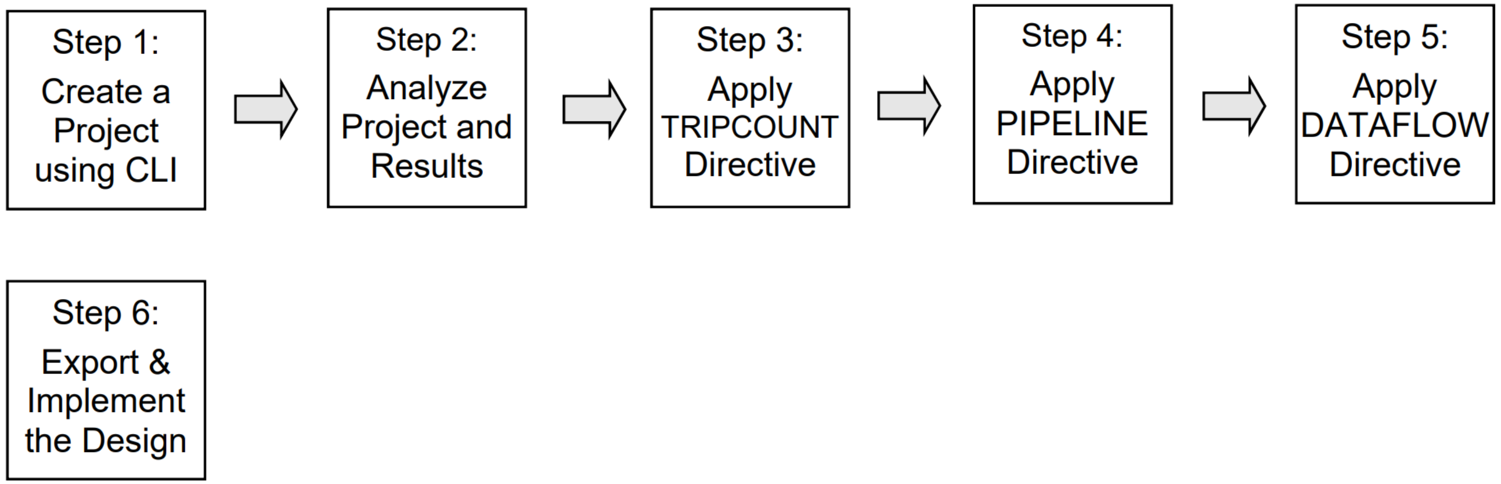
With Vivado

1st May 2023

Juwon Seo

This LAB…

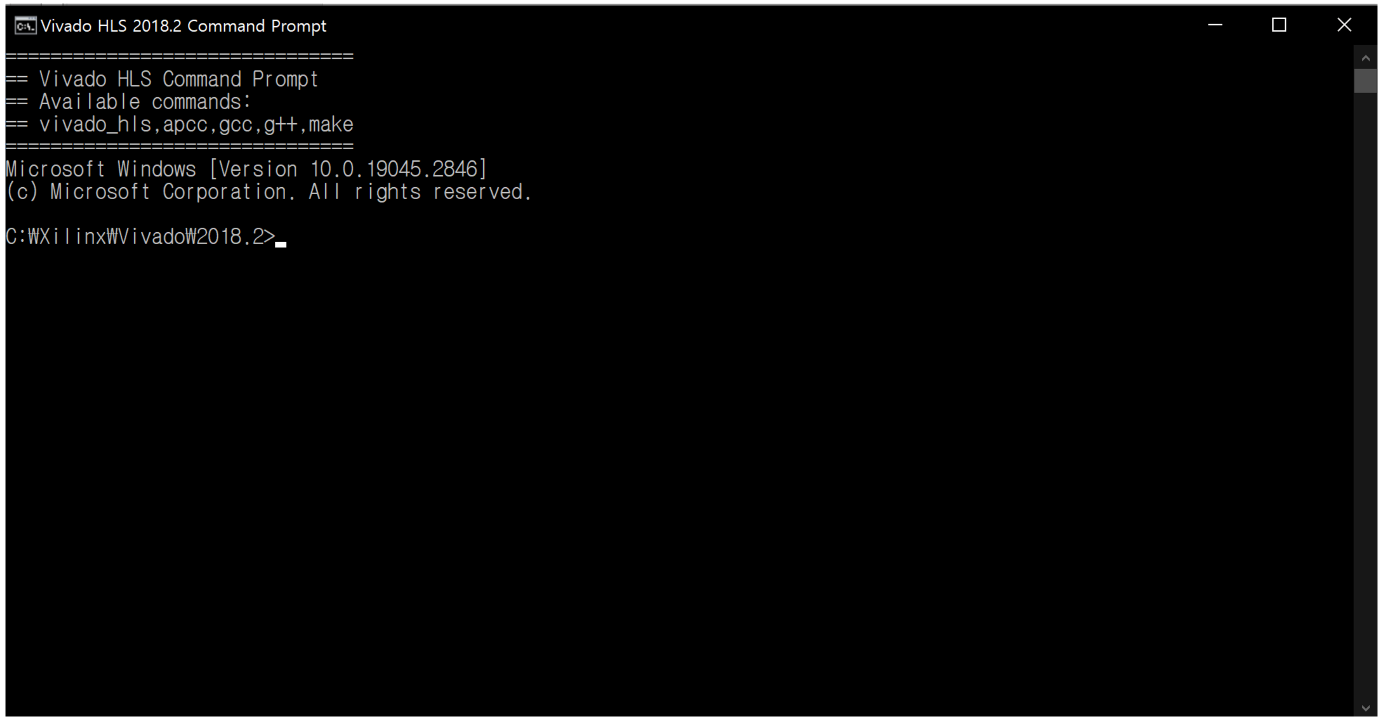
- HLS(High-level synthesis)를 활용해 design 성능 향상을 위한 flow
- Design: RGB 이미지 변환 및 필터 적용
- 학습 목표:
    - Add directives in your design
    - Understand the effect of INLINE directive
    - Improve performance using PIPELINE directive
    - Distinguish between DATAFLOW directive and Configuration Command functionality


Step 1: Create a Project using CLI
Step 2: Analyze Project and Results
Step 3: Apply TRIPCOUNT Directive
Step 4: Apply PIPELINE Directive
Step 5: Apply DATAFLOW Directive
Step 6: Export & Implement the Design

Create a Vivado HLS Project from Command Line

- HLS Prompt 생성

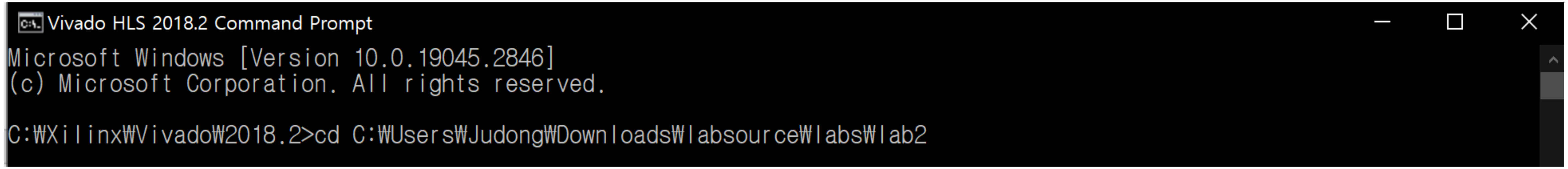
- Start - All Programs - Xilinx Design Tools - Vivado 2017.4 - Vivado HLS - Vivado HLS 2017.4 Command Prompt



```
================================
== Vivado HLS Command Prompt
== Available commands:
== vivado_hls,apcc,gcc,g++,make
================================
Microsoft Windows [Version 10.0.19045.2846]
(c) Microsoft Corporation. All rights reserved.

C:₩Xilinx₩Vivado₩2018.2>_
```

Create a Vivado HLS Project from Command Line

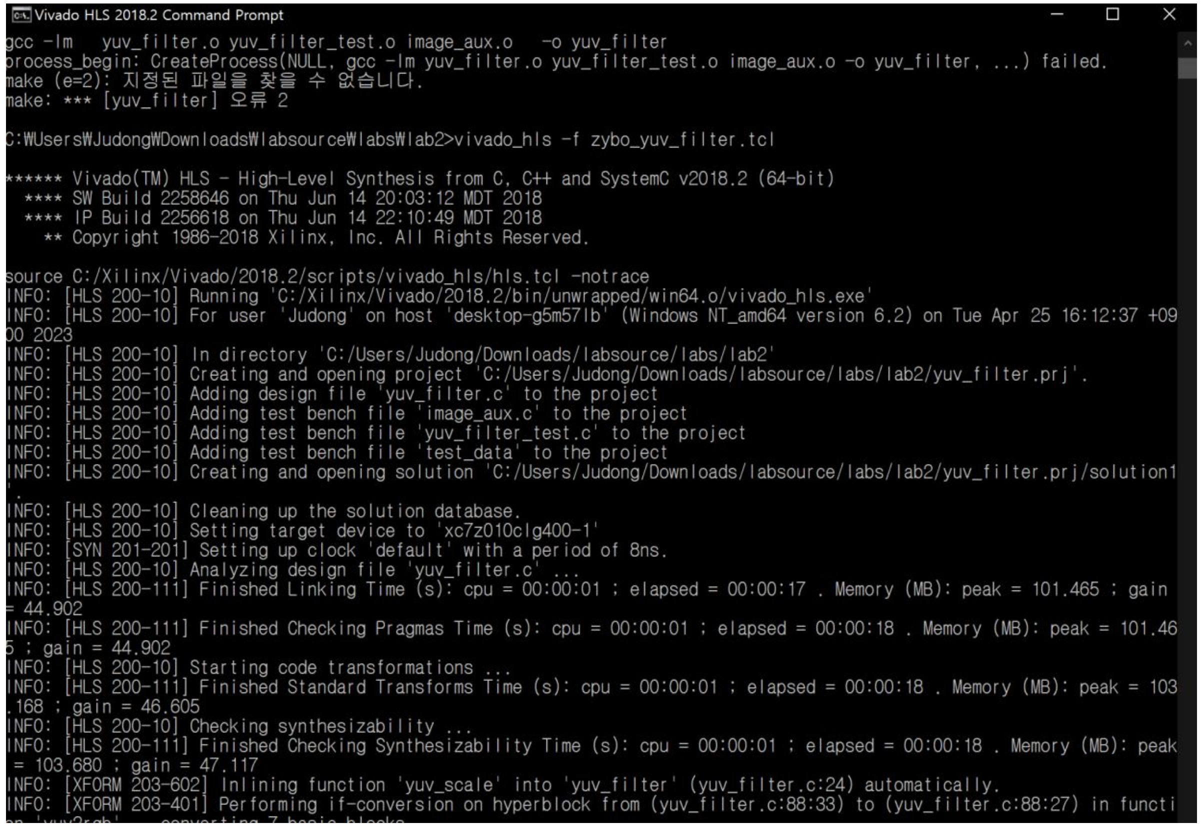
- Cd 소스파일 위치
- Ex) cd C:₩Users₩Judong₩Downloads₩labsource₩labs₩lab2



```
Microsoft Windows [Version 10.0.19045.2846]
(c) Microsoft Corporation. All rights reserved.

C:₩Xilinx₩Vivado₩2018.2>cd C:₩Users₩Judong₩Downloads₩labsource₩labs₩lab2
```

Create a Vivado HLS Project from Command Line

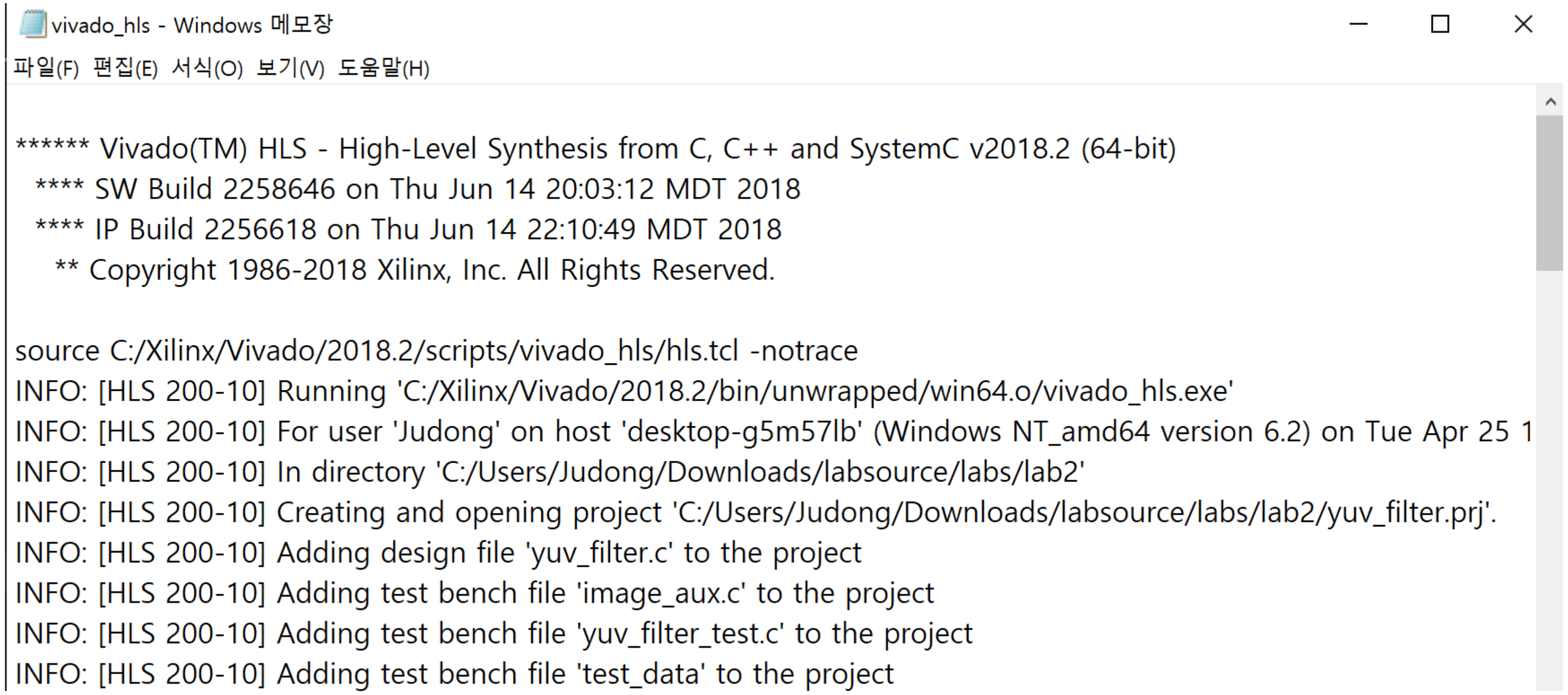
- Zybo 보드를 기반으로 한 hls 프로젝트 생성
- vivado_hls –f zybo_yuv_filter.tcl 입력



```
Vivado HLS 2018.2 Command Prompt
gcc -lm   yuv_filter.o yuv_filter_test.o image_aux.o   -o yuv_filter
process_begin: CreateProcess(NULL, gcc -lm yuv_filter.o yuv_filter_test.o image_aux.o -o yuv_filter, ...) failed.
make (e=2): 지정된 파일을 찾을 수 없습니다.
make: *** [yuv_filter] 오류 2

C:\Users\Judong\Downloads\labsource\labs\lab2>vivado_hls -f zybo_yuv_filter.tcl

****** Vivado(TM) HLS - High-Level Synthesis from C, C++ and SystemC v2018.2 (64-bit)
  **** SW Build 2258646 on Thu Jun 14 20:03:12 MDT 2018
  **** IP Build 2256618 on Thu Jun 14 22:10:49 MDT 2018
    ** Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.

source C:/Xilinx/Vivado/2018.2/scripts/vivado_hls/hls.tcl -notrace
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 Tue Apr 25 16:12:37 +09
00 2023
INFO: [HLS 200-10] In directory 'C:/Users/Judong/Downloads/labsource/labs/lab2'
INFO: [HLS 200-10] Creating and opening project 'C:/Users/Judong/Downloads/labsource/labs/lab2/yuv_filter.prj'.
INFO: [HLS 200-10] Adding design file 'yuv_filter.c' to the project
INFO: [HLS 200-10] Adding test bench file 'image_aux.c' to the project
INFO: [HLS 200-10] Adding test bench file 'yuv_filter_test.c' to the project
INFO: [HLS 200-10] Adding test bench file 'test_data' to the project
INFO: [HLS 200-10] Creating and opening solution 'C:/Users/Judong/Downloads/labsource/labs/lab2/yuv_filter.prj/solution1
'.
INFO: [HLS 200-10] Cleaning up the solution database.
INFO: [HLS 200-10] Setting target device to 'xc7z010clg400-1'
INFO: [SYN 201-201] Setting up clock 'default' with a period of 8ns.
INFO: [HLS 200-10] Analyzing design file 'yuv_filter.c' ...
INFO: [HLS 200-111] Finished Linking Time (s): cpu = 00:00:01 ; elapsed = 00:00:17 . Memory (MB): peak = 101.465 ; gain
= 44.902
INFO: [HLS 200-111] Finished Checking Pragmas Time (s): cpu = 00:00:01 ; elapsed = 00:00:18 . Memory (MB): peak = 101.46
5 ; gain = 44.902
INFO: [HLS 200-10] Starting code transformations ...
INFO: [HLS 200-111] Finished Standard Transforms Time (s): cpu = 00:00:01 ; elapsed = 00:00:18 . Memory (MB): peak = 103
.168 ; gain = 46.605
INFO: [HLS 200-10] Checking synthesizability ...
INFO: [HLS 200-111] Finished Checking Synthesizability Time (s): cpu = 00:00:01 ; elapsed = 00:00:18 . Memory (MB): peak
 = 103.680 ; gain = 47.117
INFO: [XFORM 203-602] Inlining function 'yuv_scale' into 'yuv_filter' (yuv_filter.c:24) automatically.
INFO: [XFORM 203-401] Performing if-conversion on hyperblock from (yuv_filter.c:88:33) to (yuv_filter.c:88:27) in functi
```

Create a Vivado HLS Project from Command Line

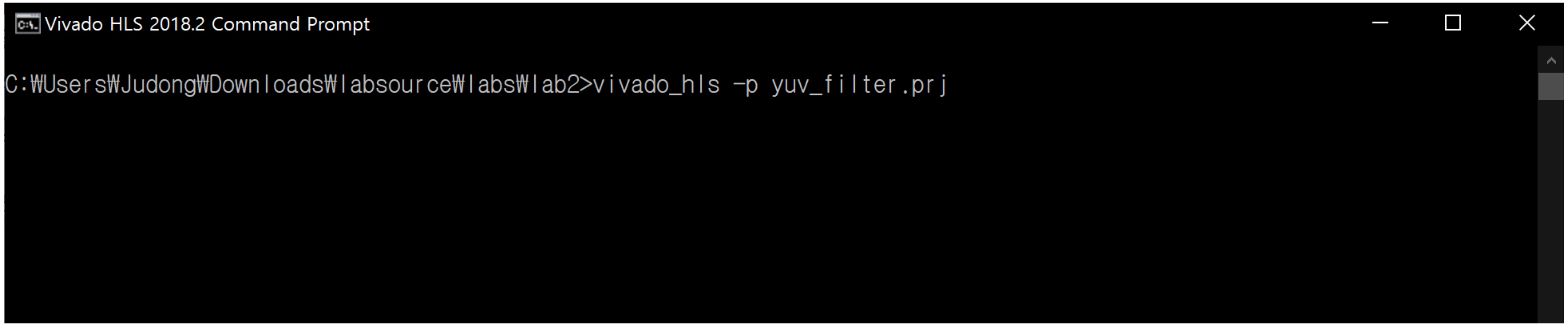
- Log파일 확인 ➔ 코드 생성 과정 확인
- <현재 위치>₩vivado_hls.log 더블 클릭


vivado_hls - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
****** Vivado(TM) HLS - High-Level Synthesis from C, C++ and SystemC v2018.2 (64-bit)
  **** SW Build 2258646 on Thu Jun 14 20:03:12 MDT 2018
  **** IP Build 2256618 on Thu Jun 14 22:10:49 MDT 2018
    ** Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.

source C:/Xilinx/Vivado/2018.2/scripts/vivado_hls/hls.tcl -notrace
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 Tue Apr 25 1
INFO: [HLS 200-10] In directory 'C:/Users/Judong/Downloads/labsource/labs/lab2'
INFO: [HLS 200-10] Creating and opening project 'C:/Users/Judong/Downloads/labsource/labs/lab2/yuv_filter.prj'.
INFO: [HLS 200-10] Adding design file 'yuv_filter.c' to the project
INFO: [HLS 200-10] Adding test bench file 'image_aux.c' to the project
INFO: [HLS 200-10] Adding test bench file 'yuv_filter_test.c' to the project
INFO: [HLS 200-10] Adding test bench file 'test_data' to the project
```

Create a Vivado HLS Project from Command Line

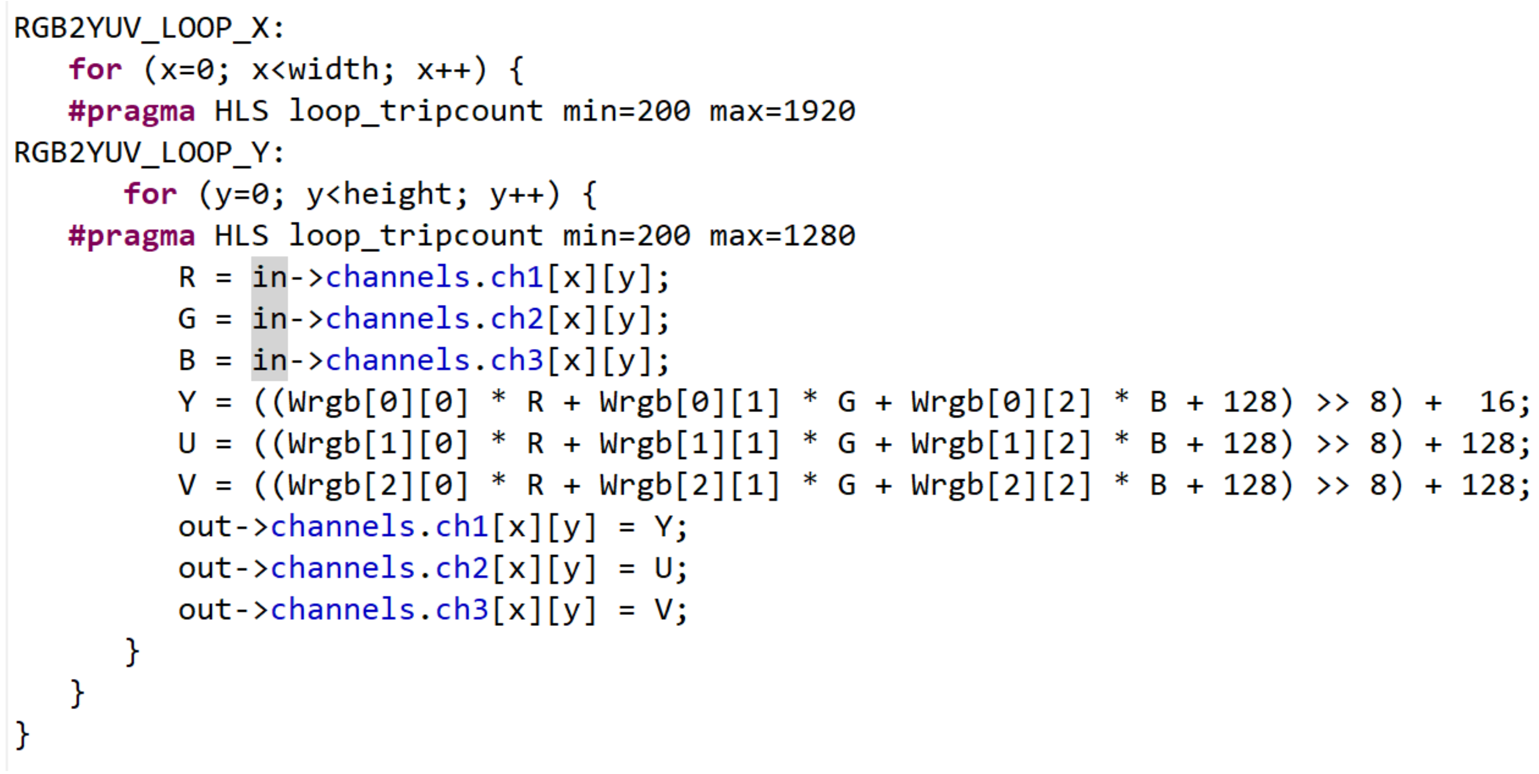
- HLS gui 생성하기
- vivado_hls –p yuv_filter.prj 입력



```
C:₩Users₩Judong₩Downloads₩labsource₩labs₩lab2>vivado_hls -p yuv_filter.prj
```

Analyze the Created Project and Results

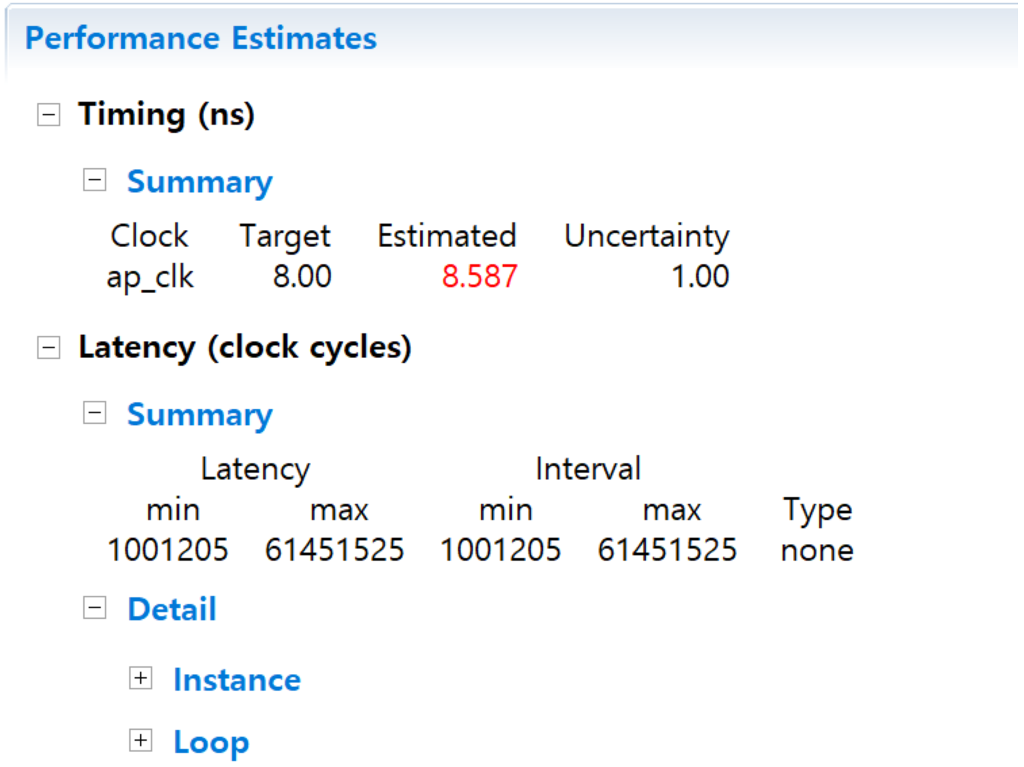
- Source file 확인
- 함수: rgb2yuv, yuv_scale, yuv2rgb



```
RGB2YUV_LOOP_X:
   for (x=0; x<width; x++) {
   #pragma HLS loop_tripcount min=200 max=1920
RGB2YUV_LOOP_Y:
      for (y=0; y<height; y++) {
   #pragma HLS loop_tripcount min=200 max=1280
         R = in->channels.ch1[x][y];
         G = in->channels.ch2[x][y];
         B = in->channels.ch3[x][y];
         Y = ((Wrgb[0][0] * R + Wrgb[0][1] * G + Wrgb[0][2] * B + 128) >> 8) +  16;
         U = ((Wrgb[1][0] * R + Wrgb[1][1] * G + Wrgb[1][2] * B + 128) >> 8) + 128;
         V = ((Wrgb[2][0] * R + Wrgb[2][1] * G + Wrgb[2][2] * B + 128) >> 8) + 128;
         out->channels.ch1[x][y] = Y;
         out->channels.ch2[x][y] = U;
         out->channels.ch3[x][y] = V;
      }
   }
}
```

Analyze the Created Project and Results

- syn – report - yuv_filter_csynh.rpt 더블 클릭
- 합성된 결과 확인 가능



Performance Estimates

⊟ **Timing (ns)**

⊟ **Summary**

Clock	Target	Estimated	Uncertainty
ap_clk	8.00	8.587	1.00

⊟ **Latency (clock cycles)**

⊟ **Summary**

Latency		Interval		
min	max	min	max	Type
1001205	61451525	1001205	61451525	none

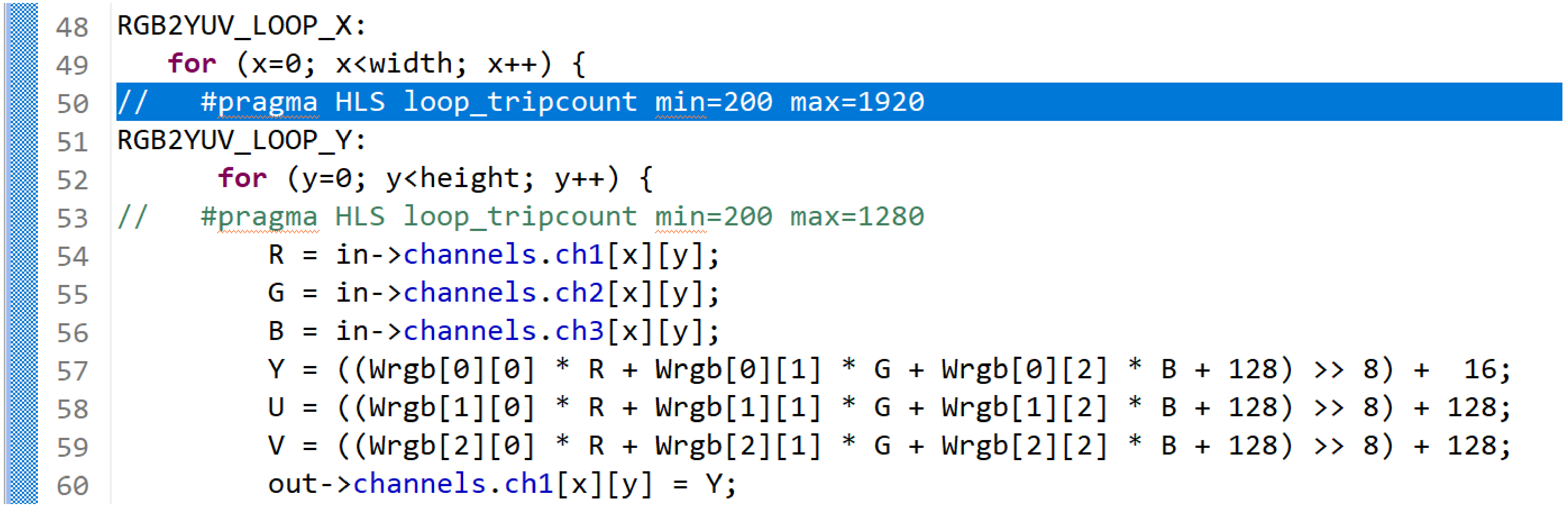
⊟ **Detail**

⊞ **Instance**

⊞ **Loop**

Apply TRIPCOUNT Pragma

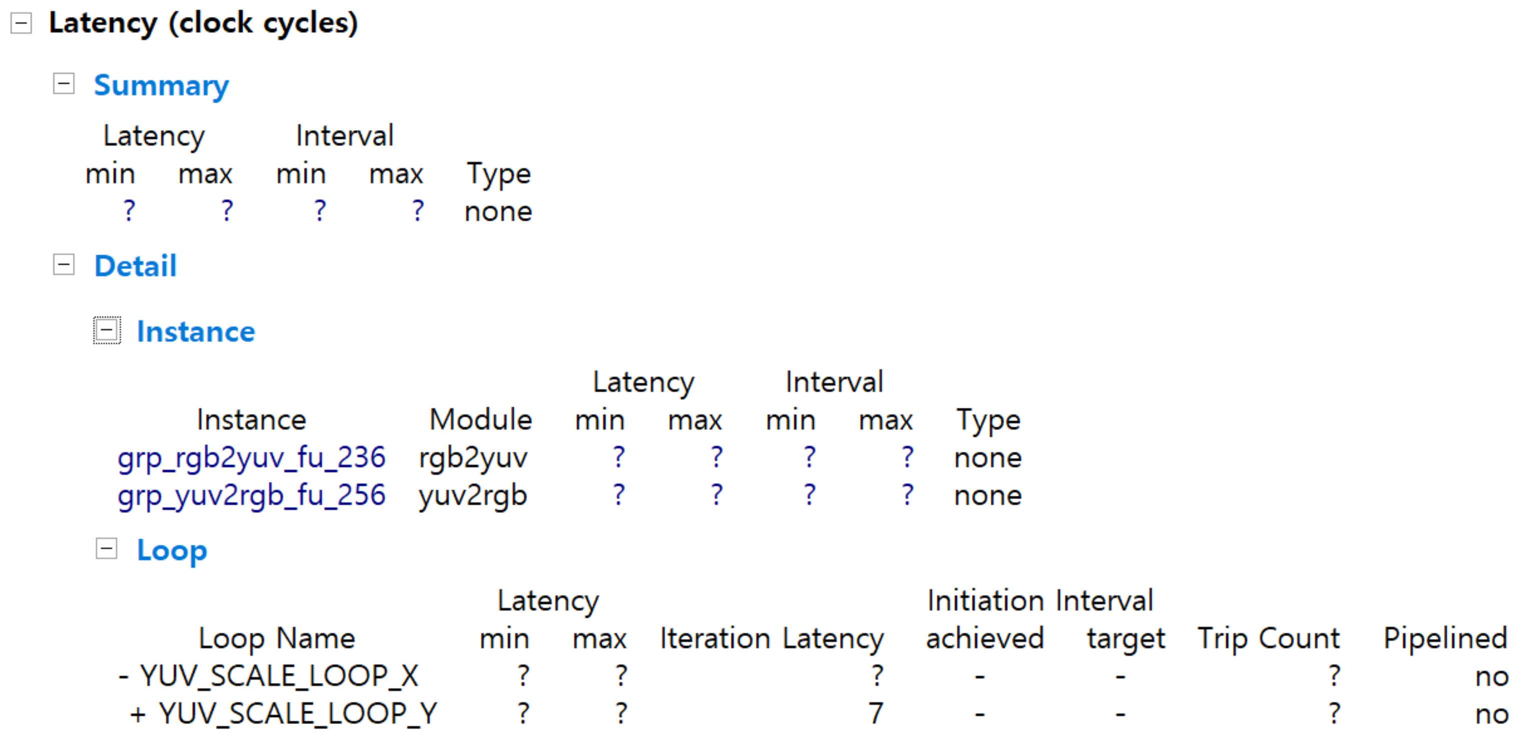
- **TRIPCOUNT**: 직접 변수가 반복되는 횟수를 지정
- **50, 53, 90, 93, 130, 133**번째 라인 #pragma 코드를 주석처리
- File - Save



```
48  RGB2YUV_LOOP_X:
49     for (x=0; x<width; x++) {
50  //   #pragma HLS loop_tripcount min=200 max=1920
51  RGB2YUV_LOOP_Y:
52        for (y=0; y<height; y++) {
53  //   #pragma HLS loop_tripcount min=200 max=1280
54           R = in->channels.ch1[x][y];
55           G = in->channels.ch2[x][y];
56           B = in->channels.ch3[x][y];
57           Y = ((Wrgb[0][0] * R + Wrgb[0][1] * G + Wrgb[0][2] * B + 128) >> 8) +  16;
58           U = ((Wrgb[1][0] * R + Wrgb[1][1] * G + Wrgb[1][2] * B + 128) >> 8) + 128;
59           V = ((Wrgb[2][0] * R + Wrgb[2][1] * G + Wrgb[2][2] * B + 128) >> 8) + 128;
60           out->channels.ch1[x][y] = Y;
```

Apply TRIPCOUNT Pragma

- **Solution - Run C Synthesis - Active Solution**
- 재합성 및 결과 확인



Latency (clock cycles)

Summary

Latency min	Latency max	Interval min	Interval max	Type
?	?	?	?	none

Detail

Instance

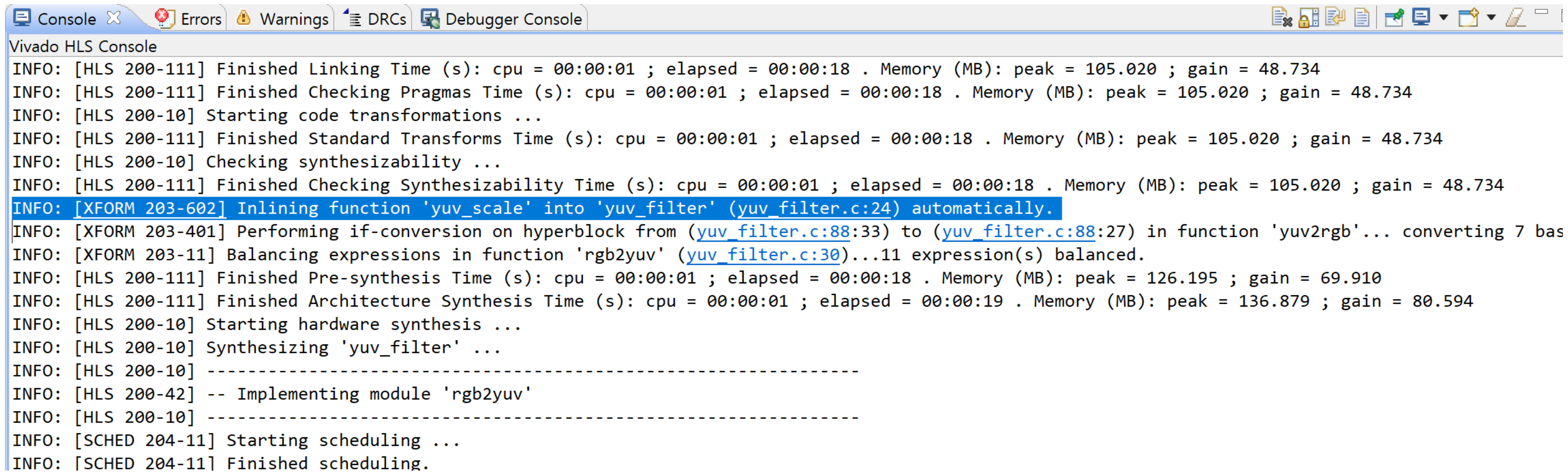
Instance	Module	Latency min	Latency max	Interval min	Interval max	Type
grp_rgb2yuv_fu_236	rgb2yuv	?	?	?	?	none
grp_yuv2rgb_fu_256	yuv2rgb	?	?	?	?	none

Loop

Loop Name	Latency min	Latency max	Iteration Latency	Initiation Interval achieved	Initiation Interval target	Trip Count	Pipelined
- YUV_SCALE_LOOP_X	?	?	?	-	-	?	no
+ YUV_SCALE_LOOP_Y	?	?	7	-	-	?	no

Apply TRIPCOUNT Pragma

- **Explorer view – Syn - Report ➔ 각 모듈 report 확인**
- Console – yuv_scale 함수가 yuv_filter에 포함되어 합성됨을 확인



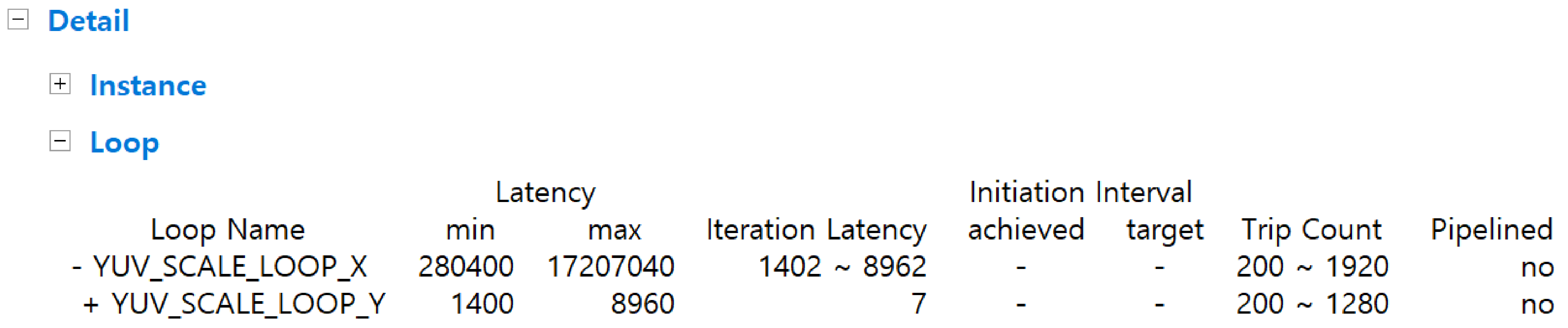
```
INFO: [HLS 200-111] Finished Linking Time (s): cpu = 00:00:01 ; elapsed = 00:00:18 . Memory (MB): peak = 105.020 ; gain = 48.734
INFO: [HLS 200-111] Finished Checking Pragmas Time (s): cpu = 00:00:01 ; elapsed = 00:00:18 . Memory (MB): peak = 105.020 ; gain = 48.734
INFO: [HLS 200-10] Starting code transformations ...
INFO: [HLS 200-111] Finished Standard Transforms Time (s): cpu = 00:00:01 ; elapsed = 00:00:18 . Memory (MB): peak = 105.020 ; gain = 48.734
INFO: [HLS 200-10] Checking synthesizability ...
INFO: [HLS 200-111] Finished Checking Synthesizability Time (s): cpu = 00:00:01 ; elapsed = 00:00:18 . Memory (MB): peak = 105.020 ; gain = 48.734
INFO: [XFORM 203-602] Inlining function 'yuv_scale' into 'yuv_filter' (yuv_filter.c:24) automatically.
INFO: [XFORM 203-401] Performing if-conversion on hyperblock from (yuv_filter.c:88:33) to (yuv_filter.c:88:27) in function 'yuv2rgb'... converting 7 bas
INFO: [XFORM 203-11] Balancing expressions in function 'rgb2yuv' (yuv_filter.c:30)...11 expression(s) balanced.
INFO: [HLS 200-111] Finished Pre-synthesis Time (s): cpu = 00:00:01 ; elapsed = 00:00:18 . Memory (MB): peak = 126.195 ; gain = 69.910
INFO: [HLS 200-111] Finished Architecture Synthesis Time (s): cpu = 00:00:01 ; elapsed = 00:00:19 . Memory (MB): peak = 136.879 ; gain = 80.594
INFO: [HLS 200-10] Starting hardware synthesis ...
INFO: [HLS 200-10] Synthesizing 'yuv_filter' ...
INFO: [HLS 200-10] ----------------------------------------------------------------
INFO: [HLS 200-42] -- Implementing module 'rgb2yuv'
INFO: [HLS 200-10] ----------------------------------------------------------------
INFO: [SCHED 204-11] Starting scheduling ...
INFO: [SCHED 204-11] Finished scheduling.
```

Apply TRIPCOUNT Pragma

- Explorer view – Syn – Report - Yuv_filter_csynth.rpt – Latency – Detail - Loop

➔ 하위 레벨 모듈 loop latency 확인

- YUV_SCALE_LOOP_Y loop latency – Trip Count 간 의미?



⊟ Detail

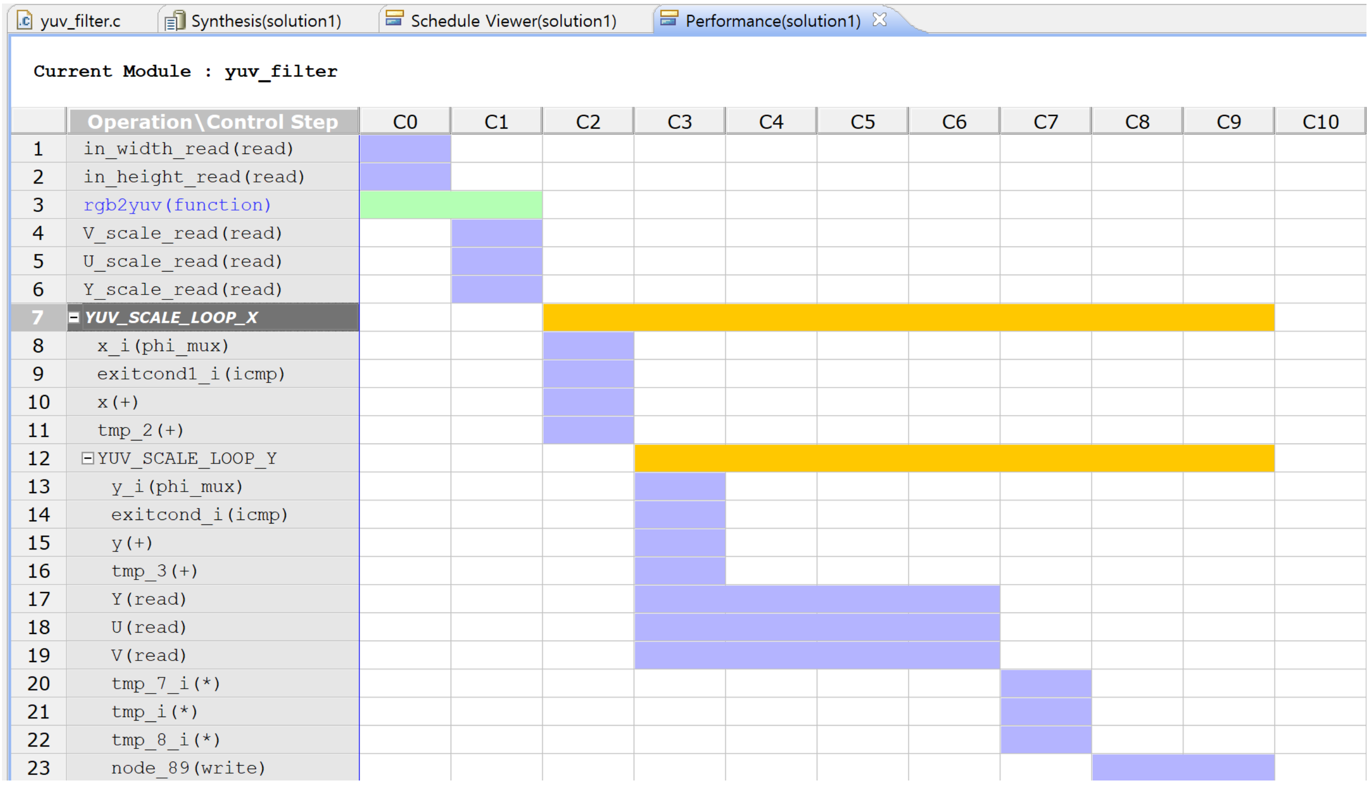
⊞ Instance

⊟ Loop

	Latency			Initiation Interval			
Loop Name	min	max	Iteration Latency	achieved	target	Trip Count	Pipelined
- YUV_SCALE_LOOP_X	280400	17207040	1402 ~ 8962	-	-	200 ~ 1920	no
+ YUV_SCALE_LOOP_Y	1400	8960	7	-	-	200 ~ 1280	no

Apply TRIPCOUNT Pragma

- analysis perspective view 클릭 Analysis
- Module Hierarchy - Yuv_filter 오른쪽 버튼 클릭 – Performance view 열기
- YUV_SCALE_LOOP_X, YUV_SCALE_LOOP_Y 확장



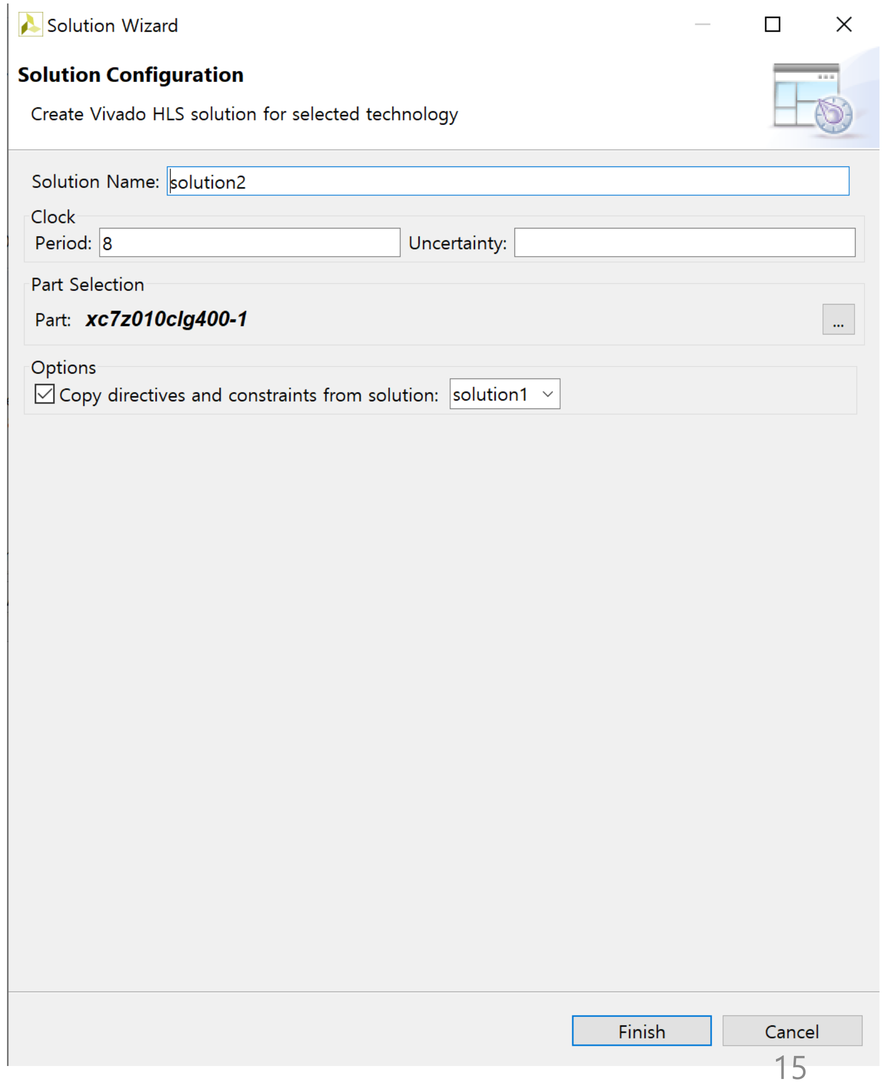
yuv_filter.c | Synthesis(solution1) | Schedule Viewer(solution1) | Performance(solution1)

Current Module : yuv_filter

	Operation\Control Step	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	in_width_read(read)	🟦										
2	in_height_read(read)	🟦										
3	rgb2yuv(function)	🟩	🟩									
4	V_scale_read(read)		🟦									
5	U_scale_read(read)		🟦									
6	Y_scale_read(read)		🟦									
7	⊟ YUV_SCALE_LOOP_X			🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	
8	x_i(phi_mux)			🟦								
9	exitcond1_i(icmp)			🟦								
10	x(+)			🟦								
11	tmp_2(+)			🟦								
12	⊟ YUV_SCALE_LOOP_Y				🟨	🟨	🟨	🟨	🟨	🟨	🟨	
13	y_i(phi_mux)				🟦							
14	exitcond_i(icmp)				🟦							
15	y(+)				🟦							
16	tmp_3(+)				🟦							
17	Y(read)				🟦	🟦	🟦	🟦				
18	U(read)				🟦	🟦	🟦	🟦				
19	V(read)				🟦	🟦	🟦	🟦				
20	tmp_7_i(*)								🟦			
21	tmp_i(*)								🟦			
22	tmp_8_i(*)								🟦			
23	node_89(write)									🟦	🟦	

Turn OFF INLINE and Apply PIPELINE Directive

- Project - New Solution
- Project name: solution2
- Clock period: 8
- Copy directives and constraints from solution: solution1



Solution Wizard

Solution Configuration

Create Vivado HLS solution for selected technology

Solution Name: solution2

Clock

Period: 8 Uncertainty:

Part Selection

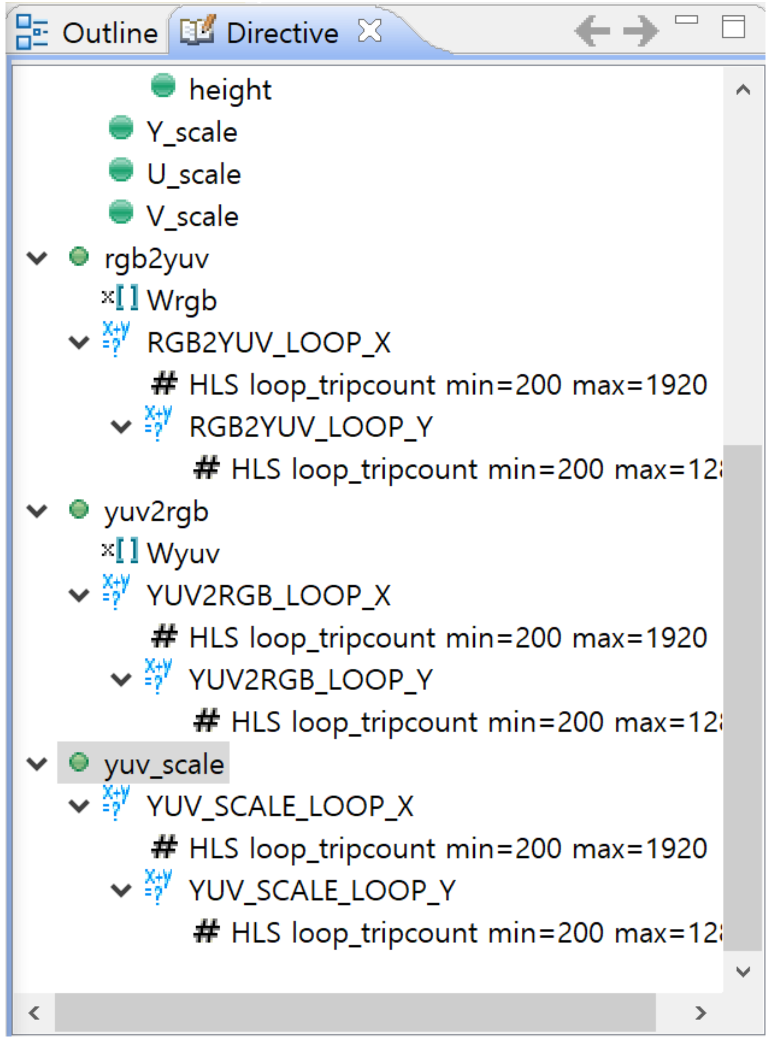
Part: ***xc7z010clg400-1***

Options

[✓] Copy directives and constraints from solution: solution1

Turn OFF INLINE and Apply PIPELINE Directive

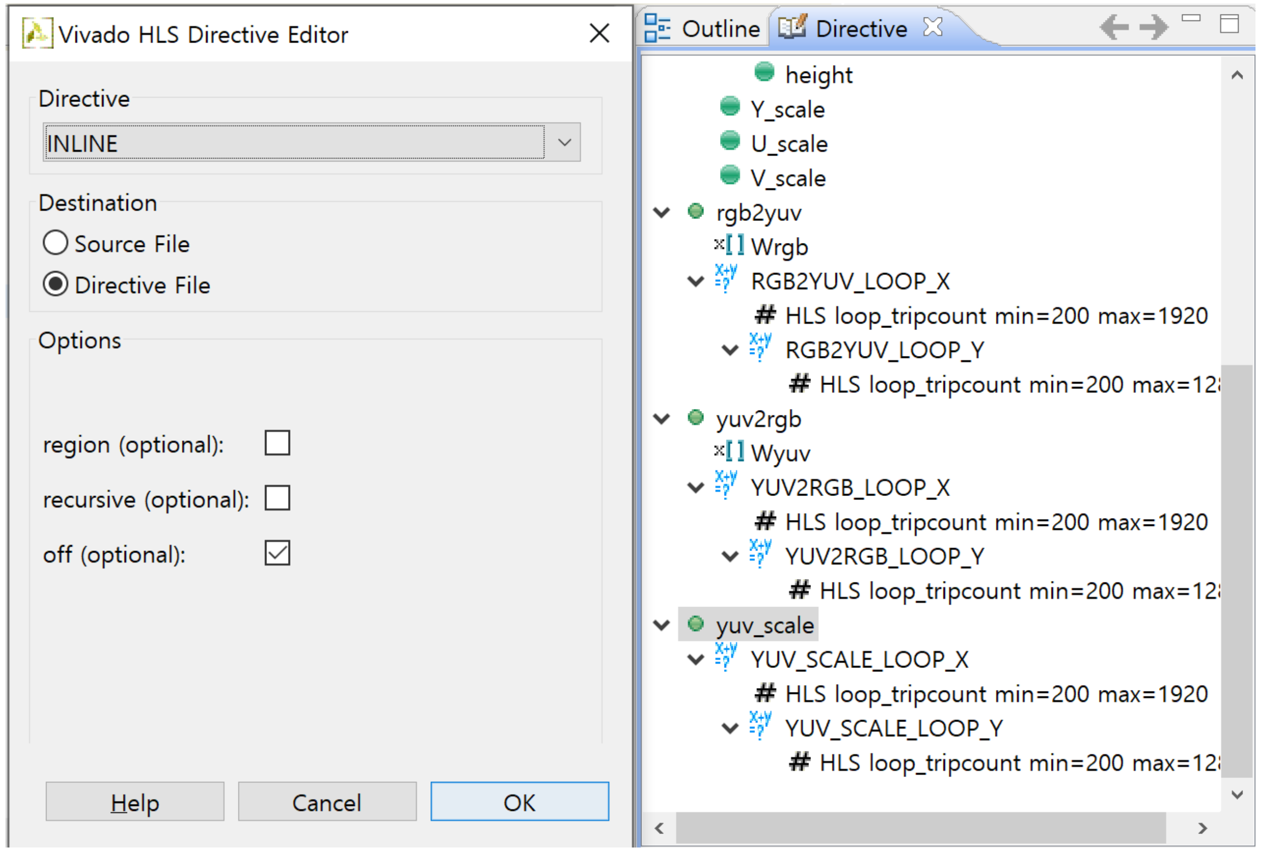
- Explorer – source – yuv_filter.c 열기
- Directive 클릭
- Yuv_scale 함수가 Directive 창에 있음을 확인



```
        height
      Y_scale
      U_scale
      V_scale
    rgb2yuv
      Wrgb
      RGB2YUV_LOOP_X
        # HLS loop_tripcount min=200 max=1920
        RGB2YUV_LOOP_Y
          # HLS loop_tripcount min=200 max=12
    yuv2rgb
      Wyuv
      YUV2RGB_LOOP_X
        # HLS loop_tripcount min=200 max=1920
        YUV2RGB_LOOP_Y
          # HLS loop_tripcount min=200 max=12
    yuv_scale
      YUV_SCALE_LOOP_X
        # HLS loop_tripcount min=200 max=1920
        YUV_SCALE_LOOP_Y
          # HLS loop_tripcount min=200 max=12
```

Turn OFF INLINE and Apply PIPELINE Directive

- Yuv_scale 함수 오른쪽 클릭 – insert directive
- Directive: INLINE
- Destination: Directive File
- Options – off 체크


Vivado HLS Directive Editor

Directive: INLINE

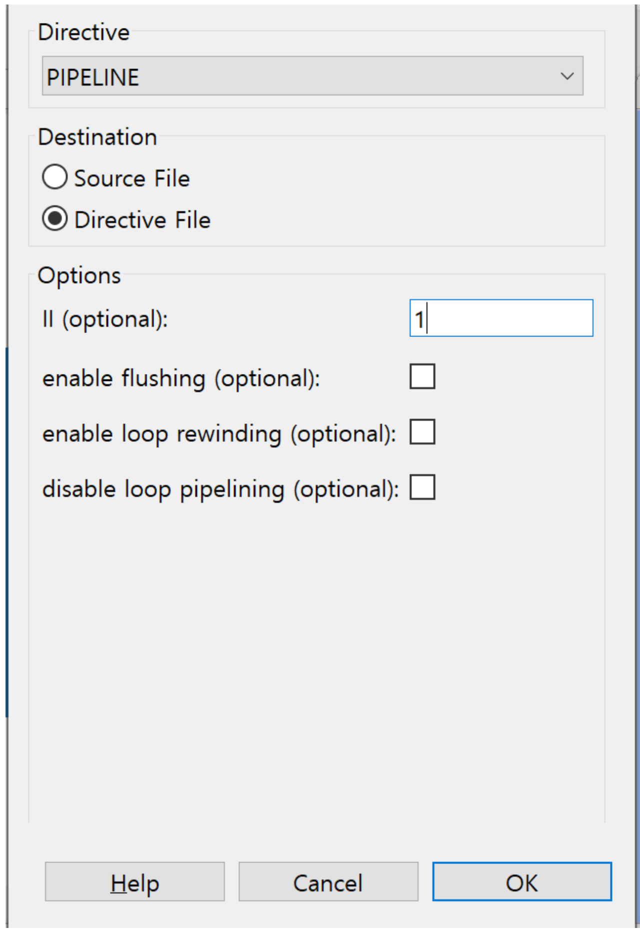
Destination
( ) Source File
(●) Directive File

Options
region (optional): ☐
recursive (optional): ☐
off (optional): ☑

```
height
Y_scale
U_scale
V_scale
rgb2yuv
  Wrgb
  RGB2YUV_LOOP_X
    # HLS loop_tripcount min=200 max=1920
    RGB2YUV_LOOP_Y
      # HLS loop_tripcount min=200 max=12
yuv2rgb
  Wyuv
  YUV2RGB_LOOP_X
    # HLS loop_tripcount min=200 max=1920
    YUV2RGB_LOOP_Y
      # HLS loop_tripcount min=200 max=12
yuv_scale
  YUV_SCALE_LOOP_X
    # HLS loop_tripcount min=200 max=1920
    YUV_SCALE_LOOP_Y
      # HLS loop_tripcount min=200 max=12
```

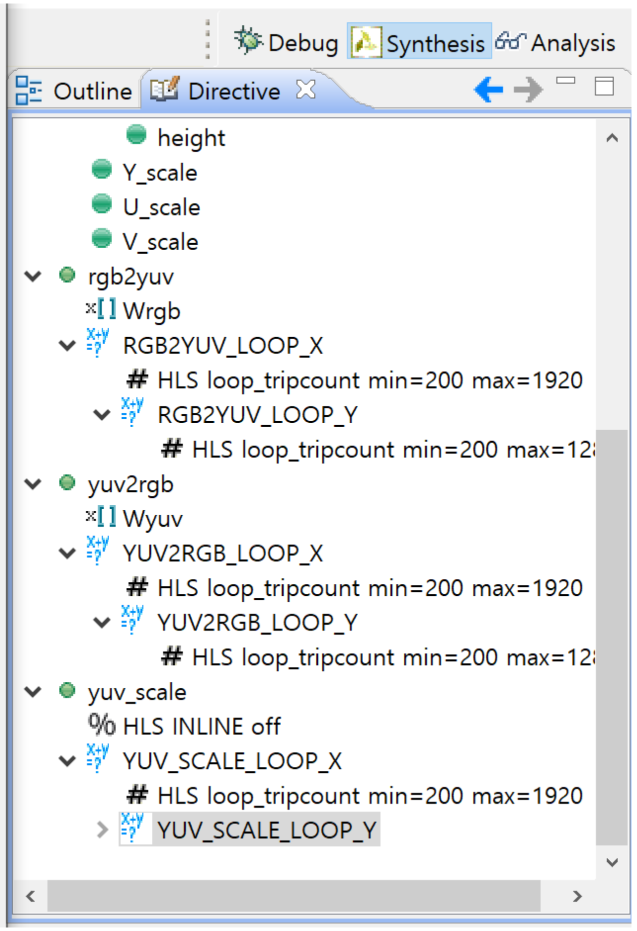
Turn OFF INLINE and Apply PIPELINE Directive

- YUV_SCALE_LOOP_Y 함수 오른쪽 클릭 – insert directive
- Directive: PIPELINE
- Destination: Directive File
- II : 1
- YUV2RGB_LOOP_Y, RGB2YUV_LOOP_Y 동일하게 PIPELINE 설정


Directive
PIPELINE

Destination
( ) Source File
(●) Directive File

Options
II (optional): 1
enable flushing (optional): ☐
enable loop rewinding (optional): ☐
disable loop pipelining (optional): ☐

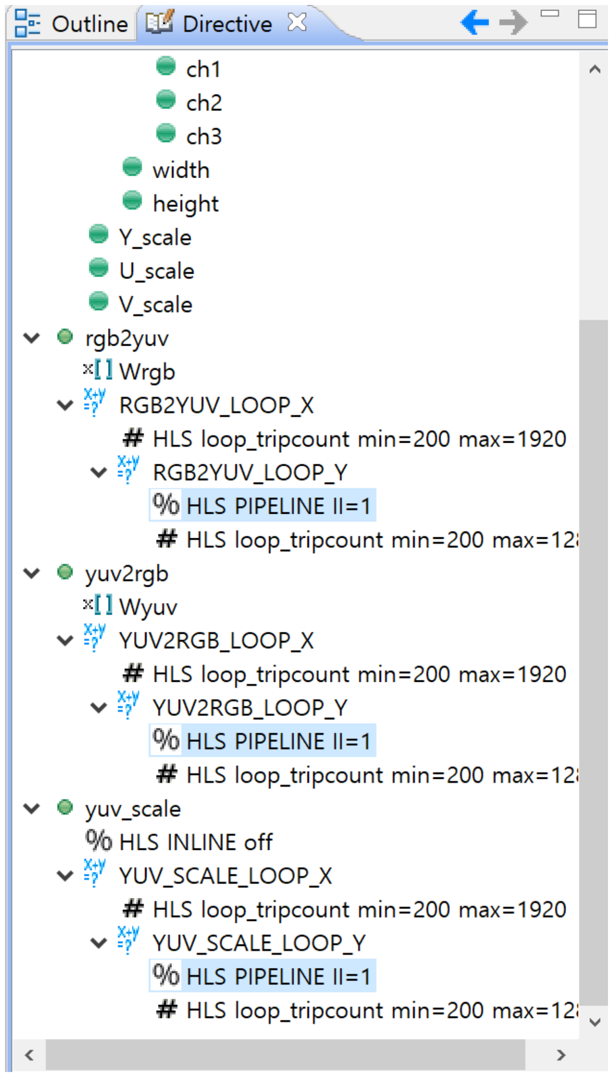


```
height
Y_scale
U_scale
V_scale
rgb2yuv
  Wrgb
  RGB2YUV_LOOP_X
    # HLS loop_tripcount min=200 max=1920
    RGB2YUV_LOOP_Y
      # HLS loop_tripcount min=200 max=12
yuv2rgb
  Wyuv
  YUV2RGB_LOOP_X
    # HLS loop_tripcount min=200 max=1920
    YUV2RGB_LOOP_Y
      # HLS loop_tripcount min=200 max=12
yuv_scale
  % HLS INLINE off
  YUV_SCALE_LOOP_X
    # HLS loop_tripcount min=200 max=1920
    YUV_SCALE_LOOP_Y
```

Turn OFF INLINE and Apply PIPELINE Directive

- **PIPELINE 설정한 함수에 다음과 같은 표시 생성**

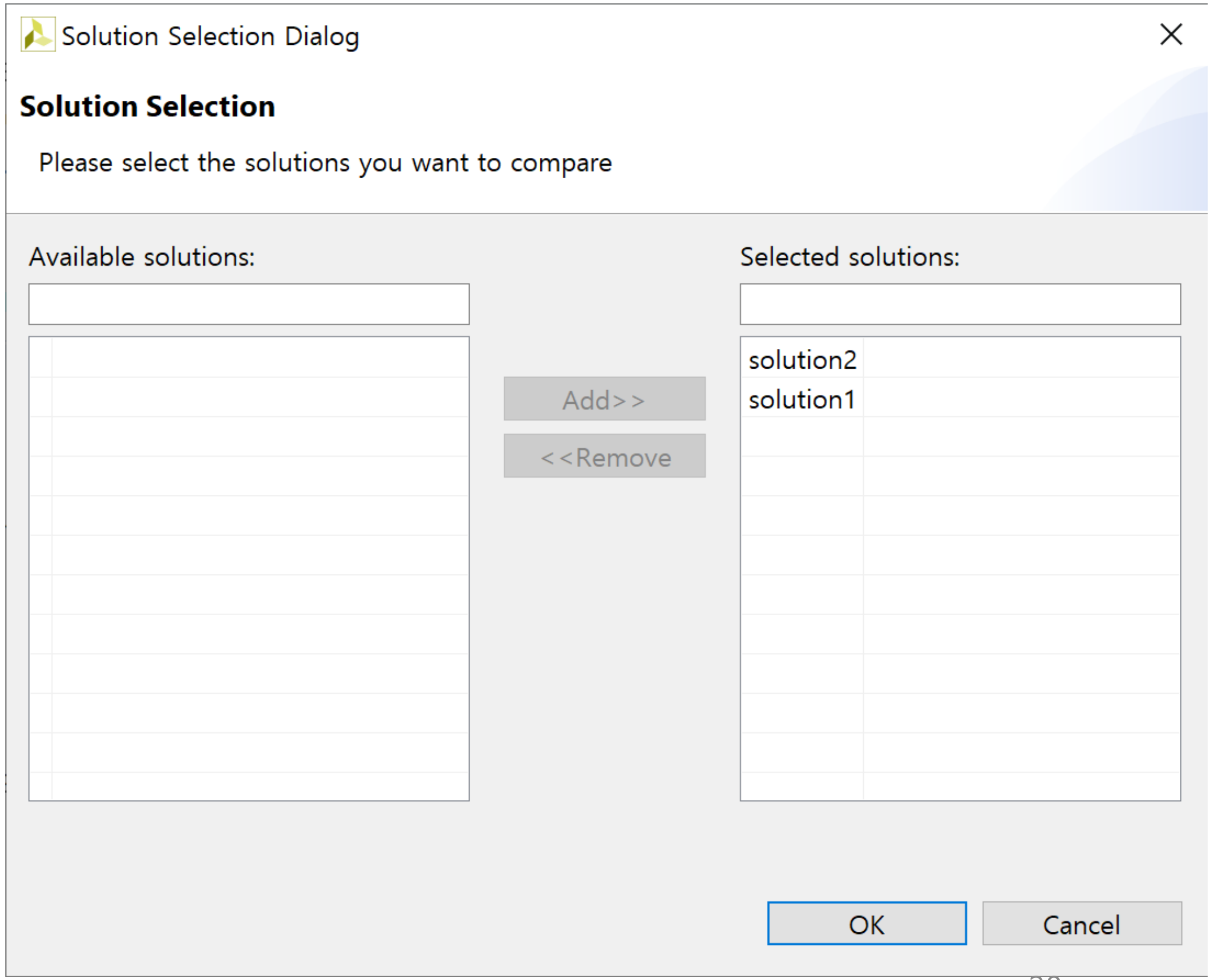
- **HLS PIPELINE II = 1**



```
Outline   Directive
        ch1
        ch2
        ch3
      width
      height
    Y_scale
    U_scale
    V_scale
  rgb2yuv
    Wrgb
    RGB2YUV_LOOP_X
      # HLS loop_tripcount min=200 max=1920
      RGB2YUV_LOOP_Y
        % HLS PIPELINE II=1
        # HLS loop_tripcount min=200 max=12
  yuv2rgb
    Wyuv
    YUV2RGB_LOOP_X
      # HLS loop_tripcount min=200 max=1920
      YUV2RGB_LOOP_Y
        % HLS PIPELINE II=1
        # HLS loop_tripcount min=200 max=12
  yuv_scale
    % HLS INLINE off
    YUV_SCALE_LOOP_X
      # HLS loop_tripcount min=200 max=1920
      YUV_SCALE_LOOP_Y
        % HLS PIPELINE II=1
        # HLS loop_tripcount min=200 max=12
```

Turn OFF INLINE and Apply PIPELINE Directive

- **Synthesis 클릭**
- **Project - Compare Reports**
- **Solution1, solution2 선택**
- **Add>> 클릭**


Solution Selection Dialog

Solution Selection

Please select the solutions you want to compare

Available solutions:

Selected solutions:

solution2
solution1

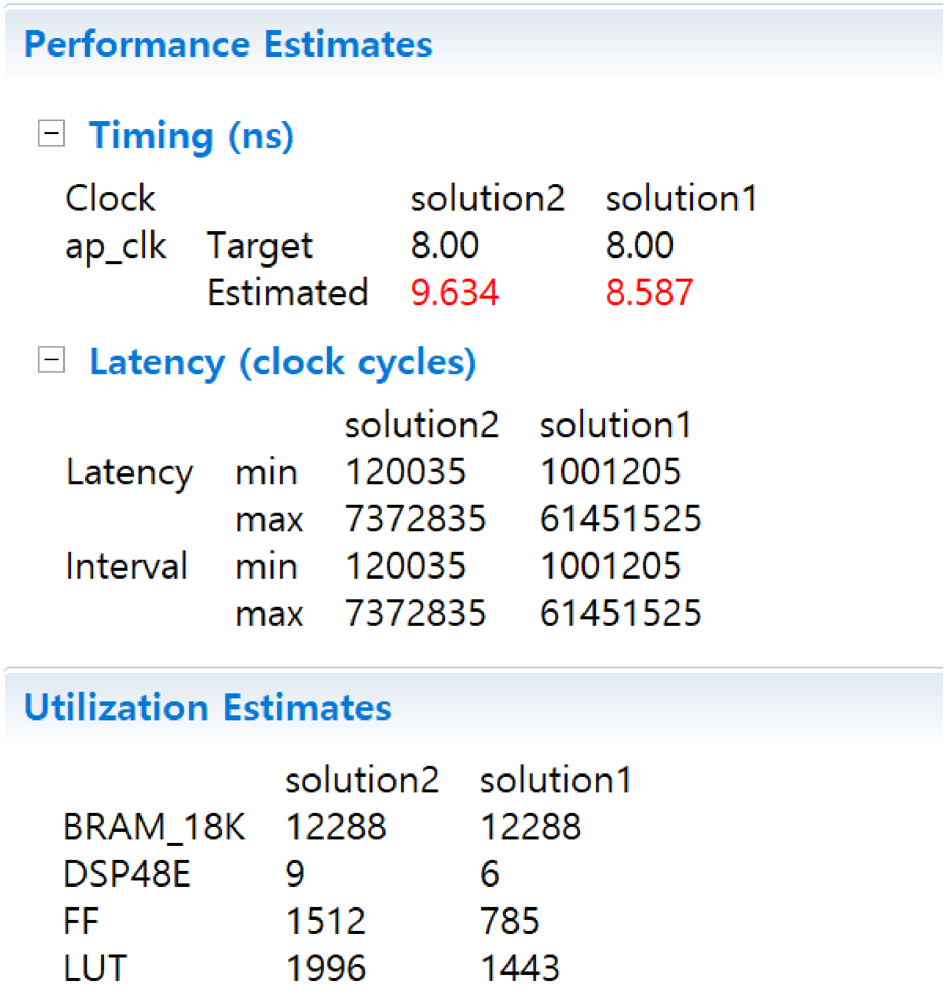
Turn OFF INLINE and Apply PIPELINE Directive

- Performance Estimates - Latency

- Solution1 (max): 61451525
- Solution2 (max): 7372835

- Utilization Estimates

- Solution1
  - DSP: 6, FF: 785, LUT: 1443
- Solution2
  - DSP: 9, FF: 1512, LUT: 1996



Performance Estimates

Timing (ns)

Clock		solution2	solution1
ap_clk	Target	8.00	8.00
	Estimated	9.634	8.587

Latency (clock cycles)

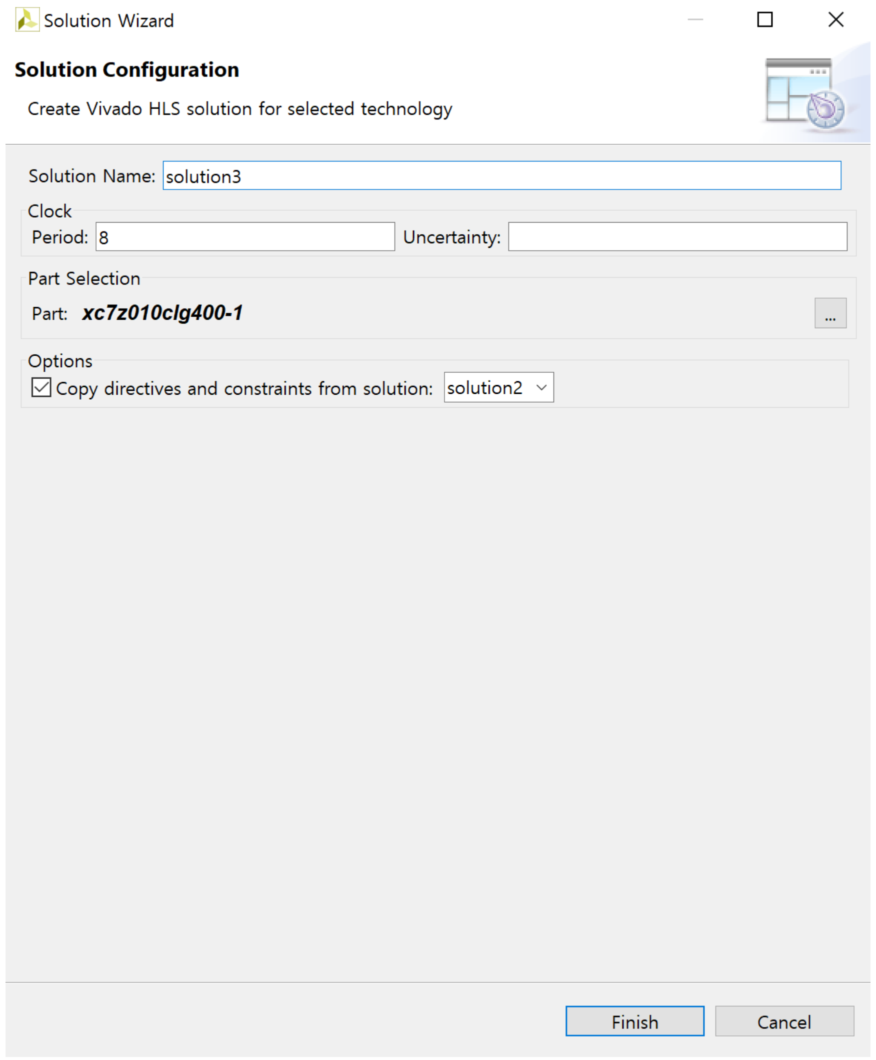
		solution2	solution1
Latency	min	120035	1001205
	max	7372835	61451525
Interval	min	120035	1001205
	max	7372835	61451525

Utilization Estimates

	solution2	solution1
BRAM_18K	12288	12288
DSP48E	9	6
FF	1512	785
LUT	1996	1443

Apply DATAFLOW Directive and Configuration Command

- Project - New Solution
- Project name: solution3
- Clock period: 8
- Copy directives and constraints from solution: solution2



Solution Configuration

Create Vivado HLS solution for selected technology

Solution Name: solution3

Clock

Period: 8 Uncertainty:

Part Selection

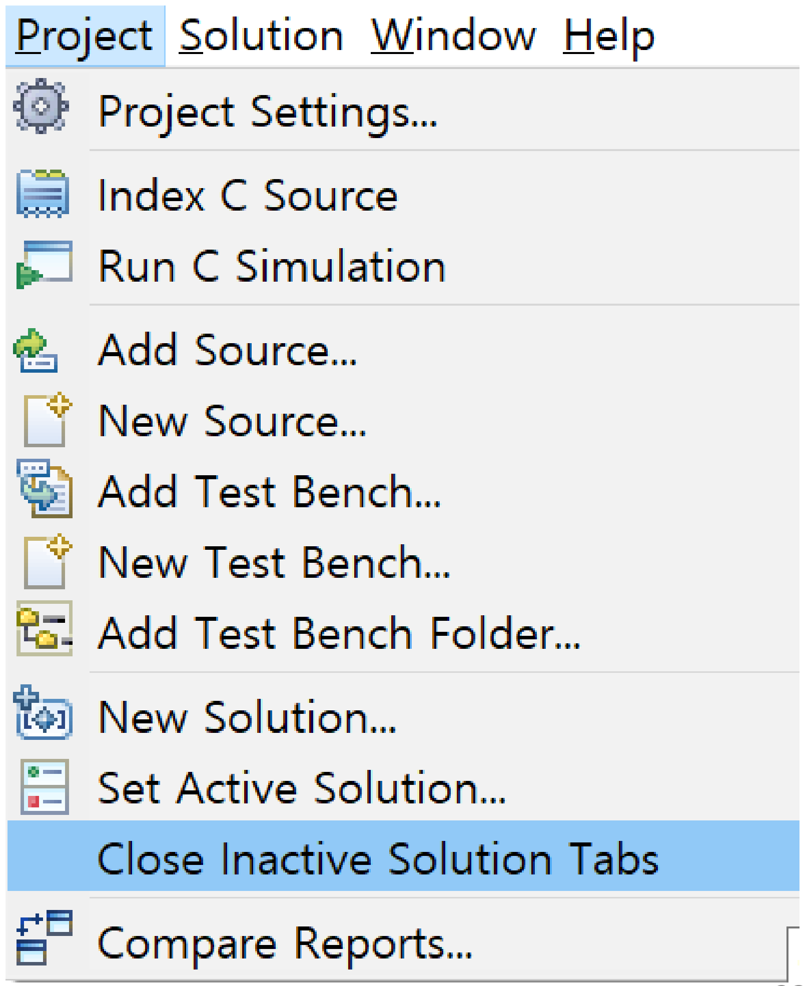
Part: ***xc7z010clg400-1***

Options

☑ Copy directives and constraints from solution: solution2

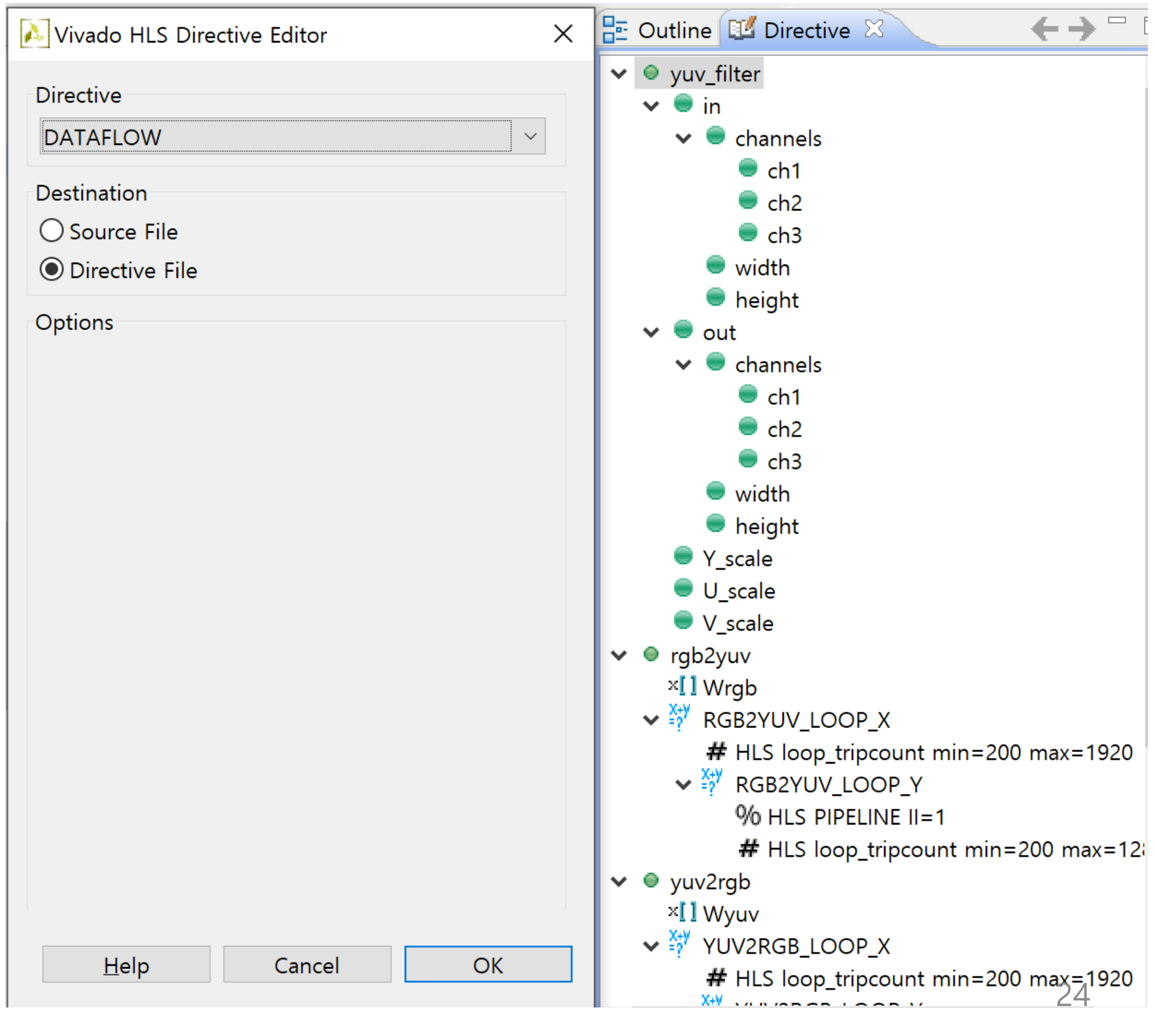
Apply DATAFLOW Directive and Configuration Command

- **Project - Close Inactive Solution Tabs**
- **Solution1, solution2 window 닫힘**



Apply DATAFLOW Directive and Configuration Command

- Yuv_filter.c 코드 open
- Yuv_filter 오른쪽 클릭 – insert directive
- Directive: DATAFLOW
- Destination: Directive File
- OK 클릭
- Synthesis 클릭



Vivado HLS Directive Editor

Directive: DATAFLOW

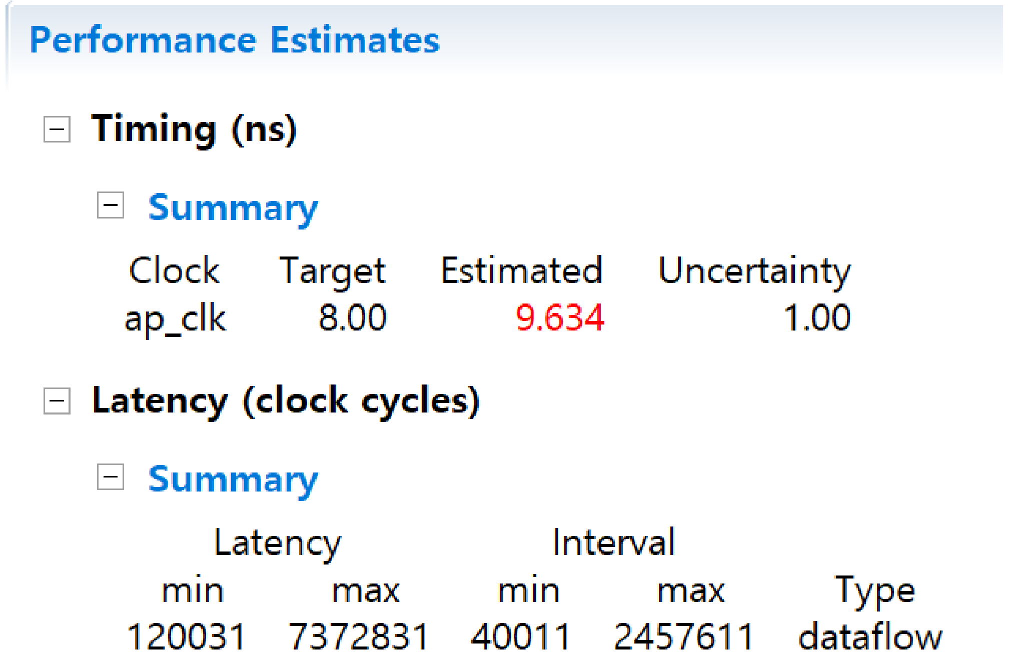
Destination
Source File [✓ Directive File]

Options

```
yuv_filter
  in
    channels
      ch1
      ch2
      ch3
    width
    height
  out
    channels
      ch1
      ch2
      ch3
    width
    height
  Y_scale
  U_scale
  V_scale
rgb2yuv
  Wrgb
  RGB2YUV_LOOP_X
    # HLS loop_tripcount min=200 max=1920
    RGB2YUV_LOOP_Y
      % HLS PIPELINE II=1
      # HLS loop_tripcount min=200 max=12
yuv2rgb
  Wyuv
  YUV2RGB_LOOP_X
    # HLS loop_tripcount min=200 max=1920
```

Apply DATAFLOW Directive and Configuration Command

- **Performance Estimates – Latency**
- **Type: dataflow**



Performance Estimates

Timing (ns)

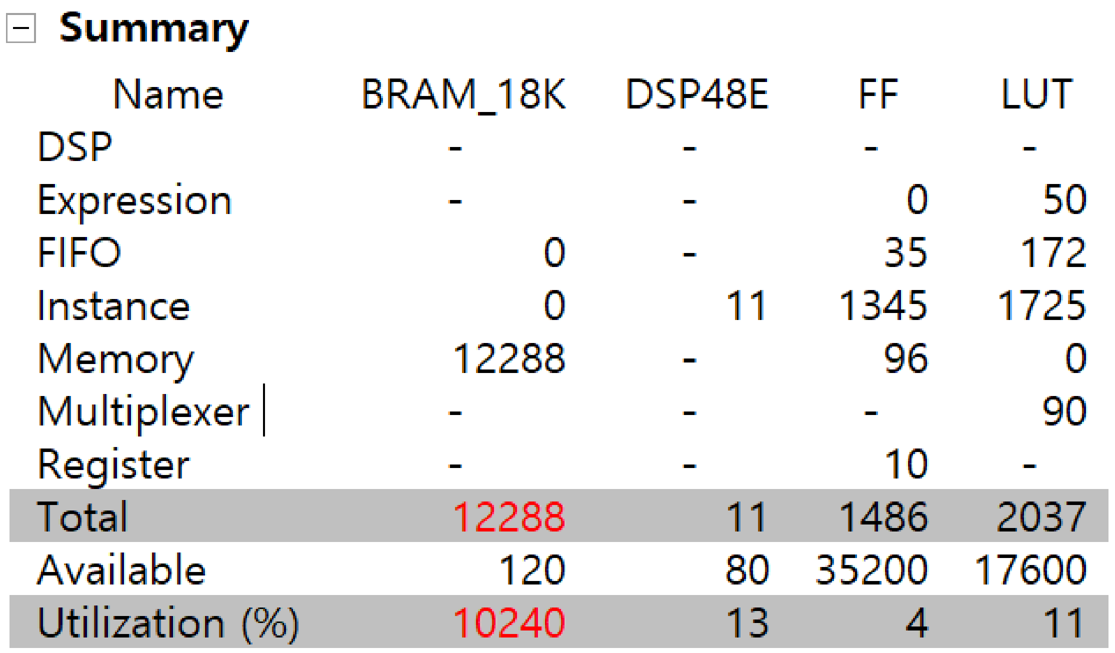
Summary

Clock	Target	Estimated	Uncertainty
ap_clk	8.00	9.634	1.00

Latency (clock cycles)

Summary

Latency		Interval		
min	max	min	max	Type
120031	7372831	40011	2457611	dataflow



Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	50
FIFO	0	-	35	172
Instance	0	11	1345	1725
Memory	12288	-	96	0
Multiplexer	-	-	-	90
Register	-	-	10	-
Total	12288	11	1486	2037
Available	120	80	35200	17600
Utilization (%)	10240	13	4	11

Apply Dataflow configuration command, generate the solution, and observe the improved resources utilization

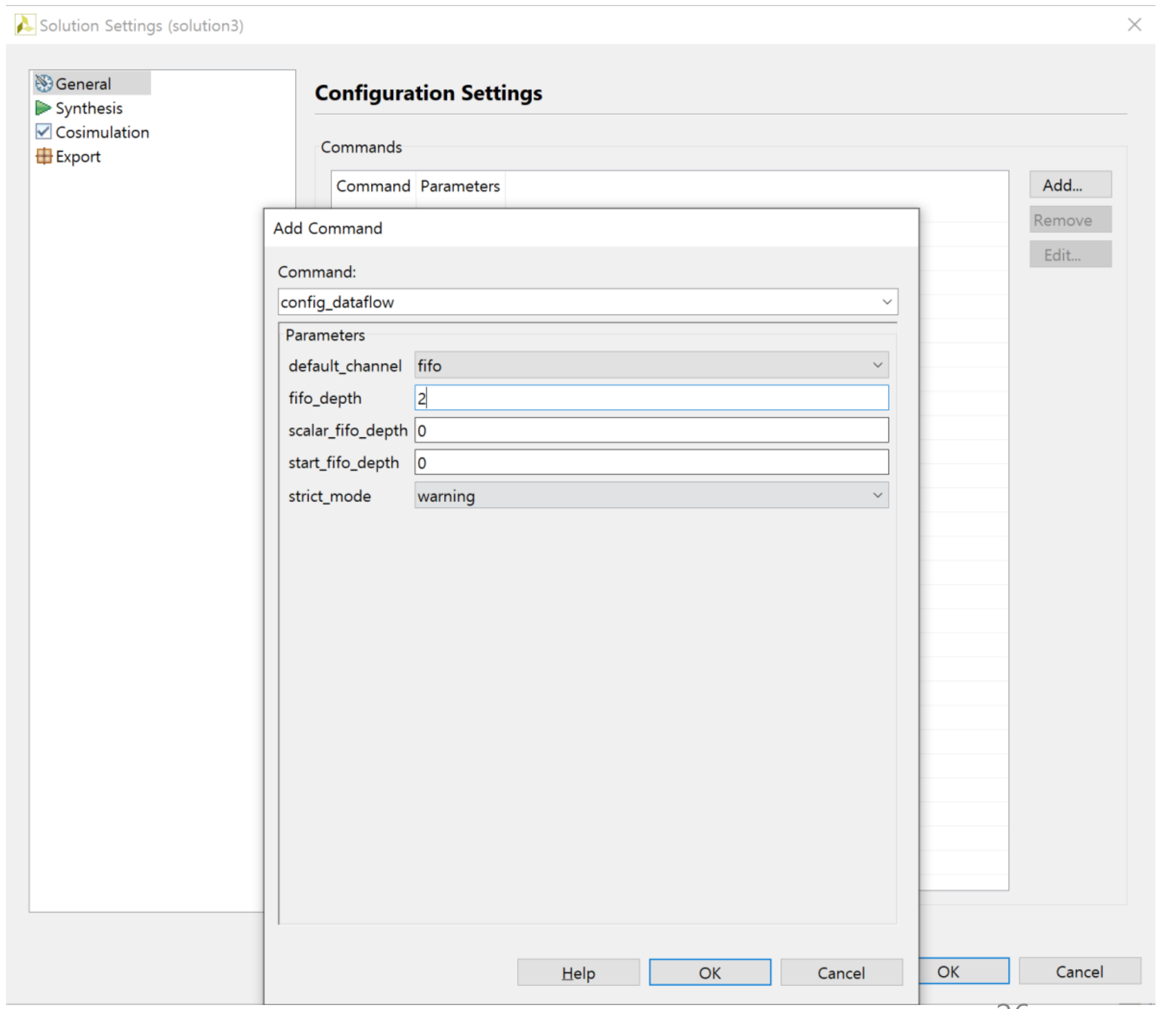
- Solution - Solution Settings

- General – Add

- Command: Config_dataflow
- Default_channel: fifo
- Fifo_depth: 2

- OK 클릭

- Synthesis 클릭



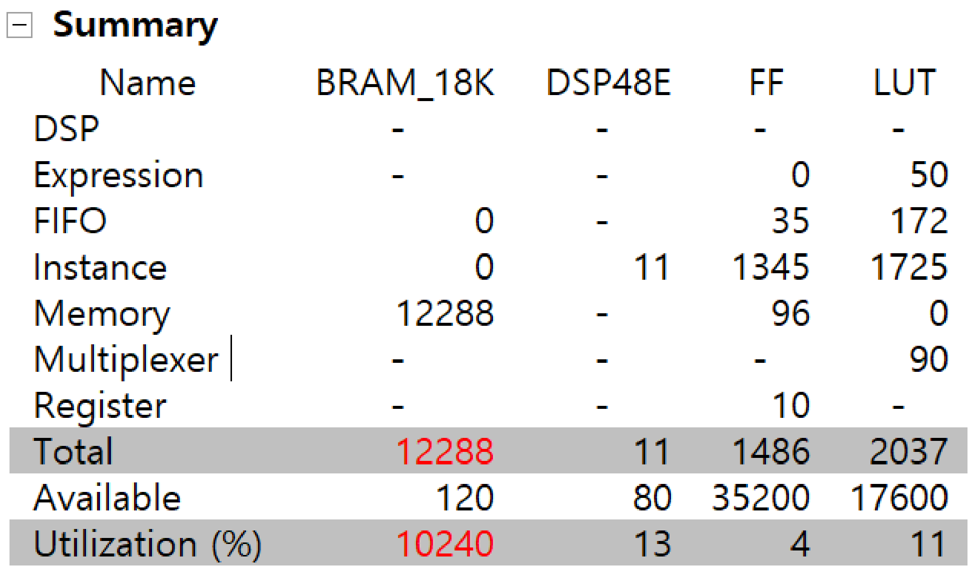
Command: config_dataflow

Parameters

default_channel	fifo
fifo_depth	2
scalar_fifo_depth	0
start_fifo_depth	0
strict_mode	warning

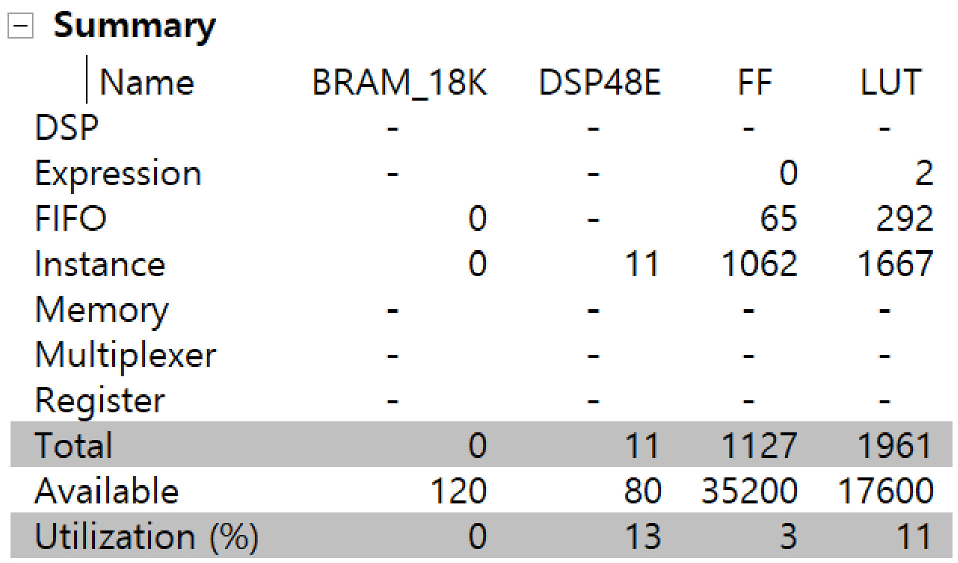
Apply Dataflow configuration command, generate the solution, and observe the improved resources utilization

- **Resource estimates: BRAM, FF, LUT**
- **Design이 사용안하는 resource 제외**



Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	50
FIFO	0	-	35	172
Instance	0	11	1345	1725
Memory	12288	-	96	0
Multiplexer	-	-	-	90
Register	-	-	10	-
Total	12288	11	1486	2037
Available	120	80	35200	17600
Utilization (%)	10240	13	4	11



Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	2
FIFO	0	-	65	292
Instance	0	11	1062	1667
Memory	-	-	-	-
Multiplexer	-	-	-	-
Register	-	-	-	-
Total	0	11	1127	1961
Available	120	80	35200	17600
Utilization (%)	0	13	3	11

감사합니다!

- **Q&A**