Traineeships in Advanced Computing for High Energy Physics (TAC-HEP)

FPGA module training

Week-6

Lecture-11: 04/03/2025

Varun Sharma

University of Wisconsin – Madison, USA

Content

- Vivado/Vitis HLS Setup
  - First project

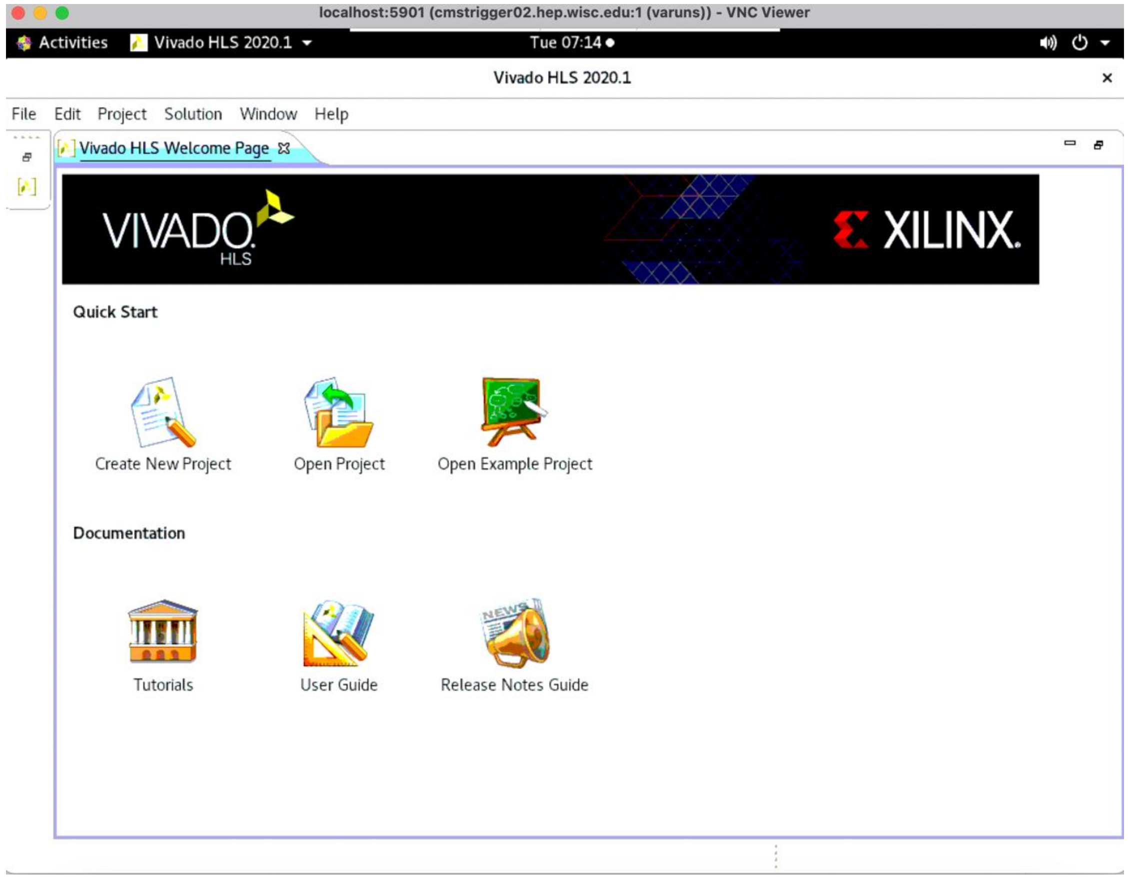
Reminder: HLS Setup

- ssh <username>@cmstrigger02-via-login -L5901:localhost:5901
  - Or whatever ***:1*** display number
  - Sometimes you may need to run vncserver -localhost -geometry 1024x768 again to start new vnc server

- **Connect to VNC server (remote desktop) client**
- **Open terminal**
  - source /opt/Xilinx/Vivado/2020.1/settings64.sh
  - cd /scratch/`whoami`
  - vivado_hls

OR

  - Source /opt/Xilinx/Vitis/2020.1/settings64.sh
  - Cd /scratch/`whoami`
  - vitis_hls

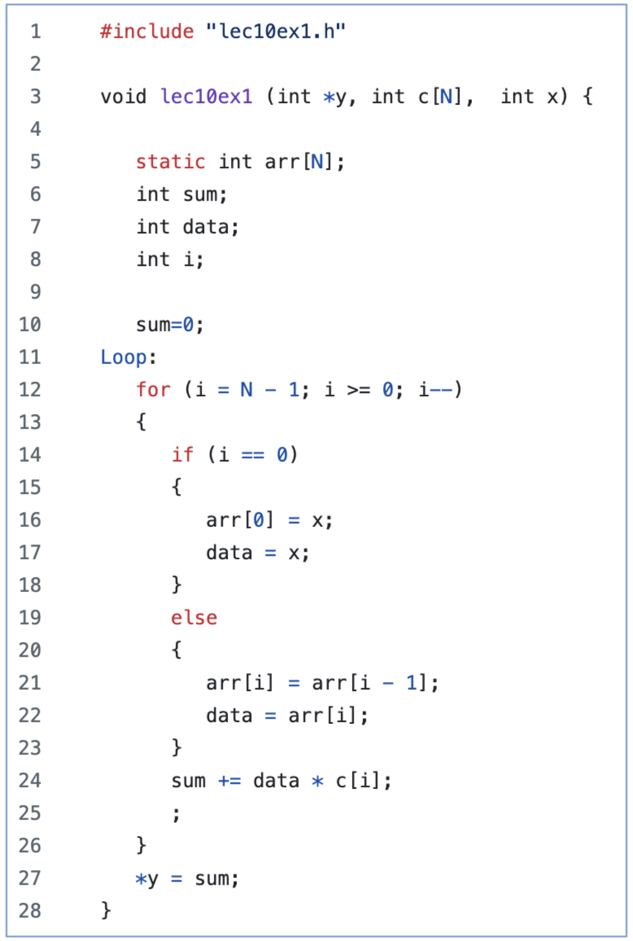


TAC-HEP 2025

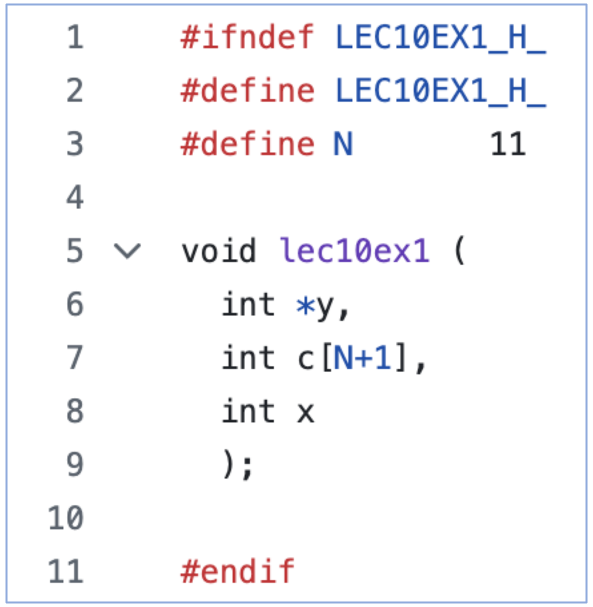
Example

https://github.com/varuns23/TAC-HEP-FPGA/tree/main/tutorial/wk5lec10

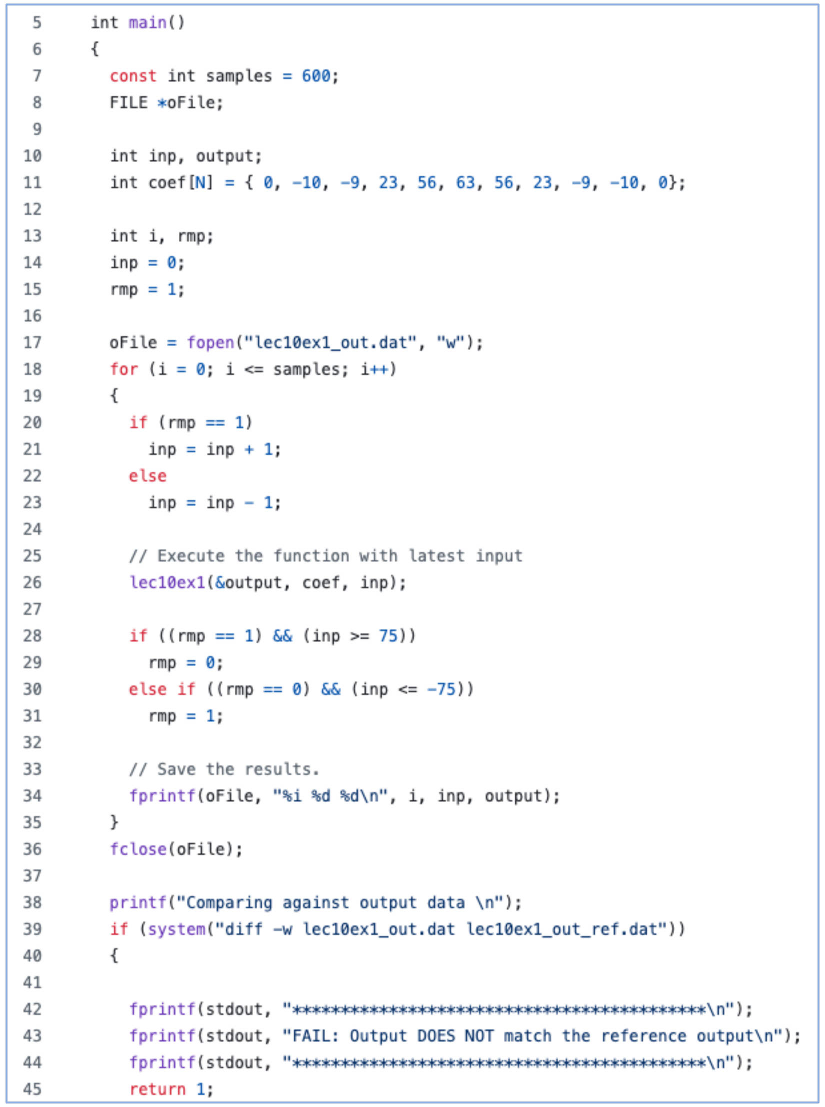
lec10ex1



```c
#include "lec10ex1.h"

void lec10ex1 (int *y, int c[N],  int x) {

   static int arr[N];
   int sum;
   int data;
   int i;

   sum=0;
Loop:
   for (i = N - 1; i >= 0; i--)
   {
      if (i == 0)
      {
         arr[0] = x;
         data = x;
      }
      else
      {
         arr[i] = arr[i - 1];
         data = arr[i];
      }
      sum += data * c[i];
      ;
   }
   *y = sum;
}
```

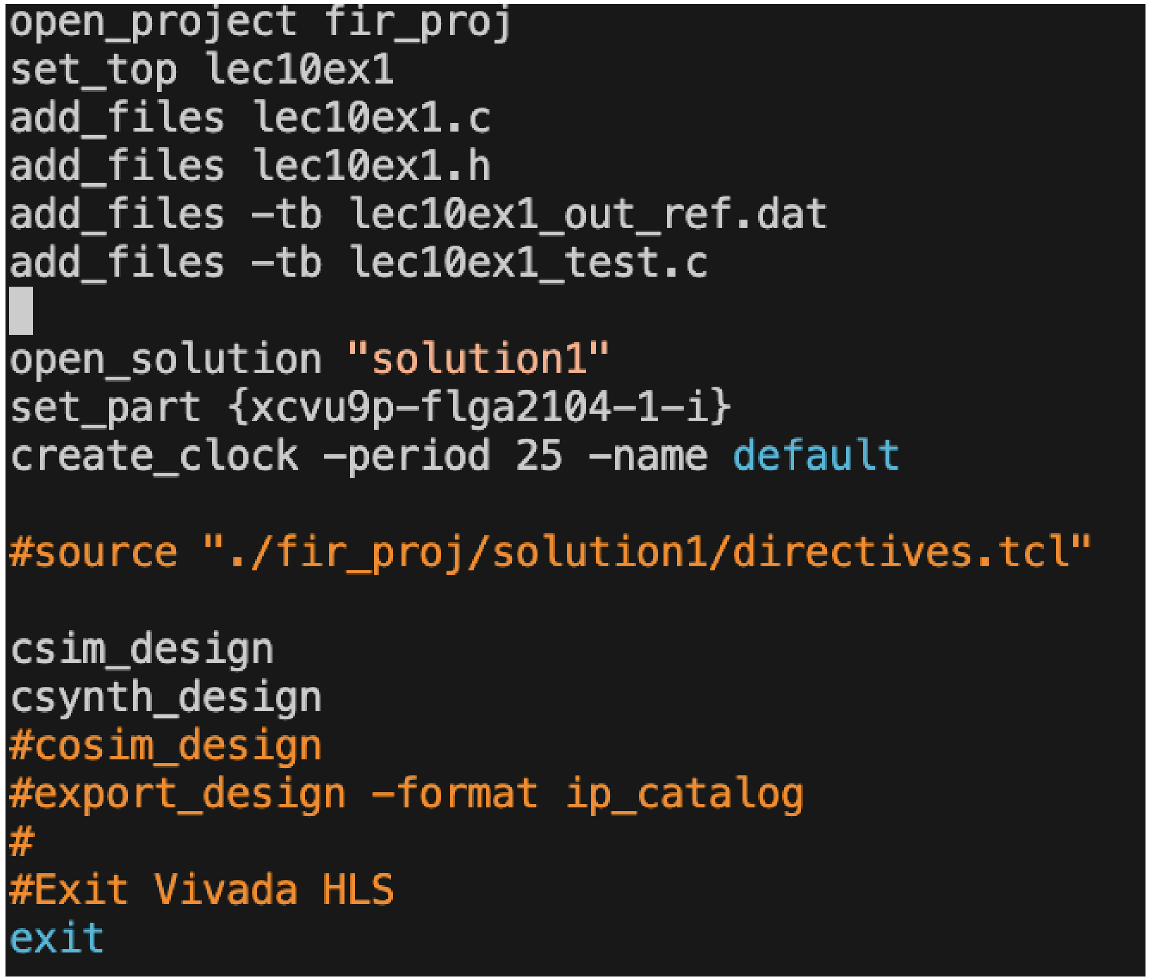


```c
#ifndef LEC10EX1_H_
#define LEC10EX1_H_
#define N       11

void lec10ex1 (
   int *y,
   int c[N+1],
   int x
   );

#endif
```



```c
int main()
{
  const int samples = 600;
  FILE *oFile;

  int inp, output;
  int coef[N] = { 0, -10, -9, 23, 56, 63, 56, 23, -9, -10, 0};

  int i, rmp;
  inp = 0;
  rmp = 1;

  oFile = fopen("lec10ex1_out.dat", "w");
  for (i = 0; i <= samples; i++)
  {
    if (rmp == 1)
      inp = inp + 1;
    else
      inp = inp - 1;

    // Execute the function with latest input
    lec10ex1(&output, coef, inp);

    if ((rmp == 1) && (inp >= 75))
      rmp = 0;
    else if ((rmp == 0) && (inp <= -75))
      rmp = 1;

    // Save the results.
    fprintf(oFile, "%i %d %d\n", i, inp, output);
  }
  fclose(oFile);

  printf("Comparing against output data \n");
  if (system("diff -w lec10ex1_out.dat lec10ex1_out_ref.dat"))
  {

    fprintf(stdout, "********************************************\n");
    fprintf(stdout, "FAIL: Output DOES NOT match the reference output\n");
    fprintf(stdout, "********************************************\n");
    return 1;
```

Run via CLI



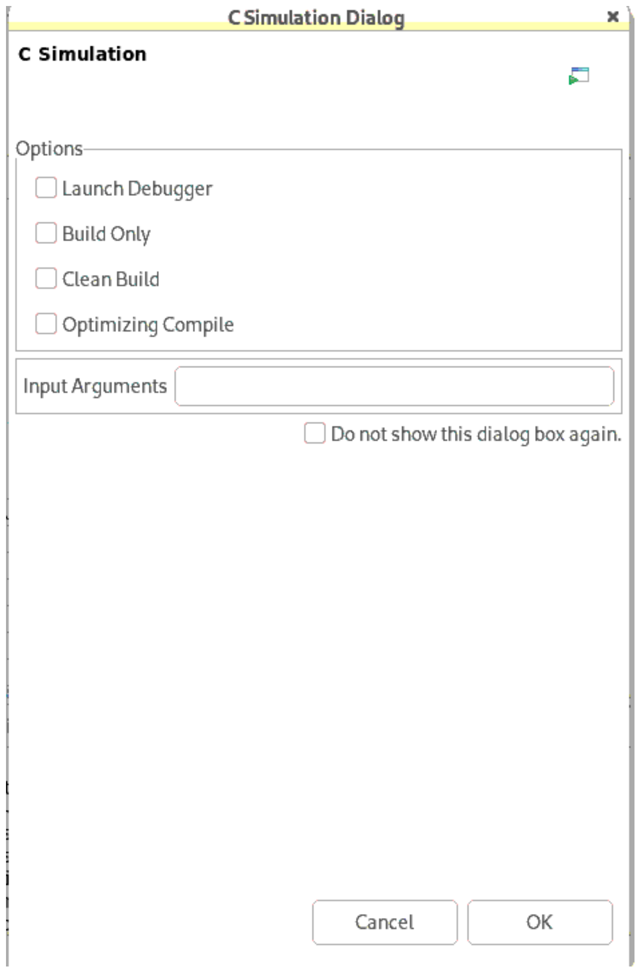
```
open_project fir_proj
set_top lec10ex1
add_files lec10ex1.c
add_files lec10ex1.h
add_files -tb lec10ex1_out_ref.dat
add_files -tb lec10ex1_test.c

open_solution "solution1"
set_part {xcvu9p-flga2104-1-i}
create_clock -period 25 -name default

#source "./fir_proj/solution1/directives.tcl"

csim_design
csynth_design
#cosim_design
#export_design -format ip_catalog
#
#Exit Vivada HLS
exit
```

Execute: *vivado_hls file.tcl*

TAC-HEP 2025

Review Reports

C – Simulation


C Simulation Dialog

C Simulation

Options
- [ ] Launch Debugger
- [ ] Build Only
- [ ] Clean Build
- [ ] Optimizing Compile

Input Arguments

- [ ] Do not show this dialog box again.

Launch Debugger: Compiles the C code & opens the debug perspective

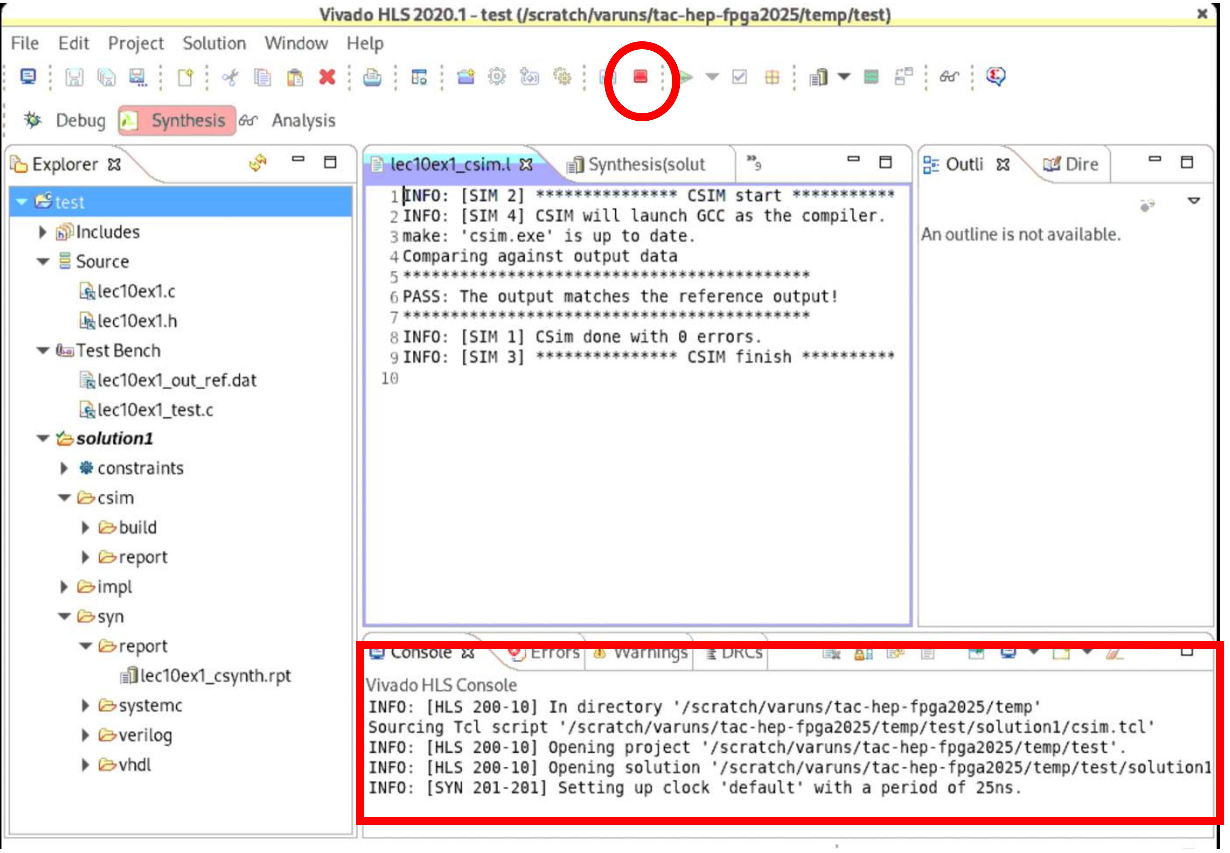
Build Only: C code compiles, but the simulation does not run

Clean Build: Remove any existing executable and object files from the project before compiling the code

Optimized Compile: Uses a higher level of optimization effort when compiling the design but removes all information required by the debugger. This increases the compile time but should reduce the simulation run time

C – Simulation

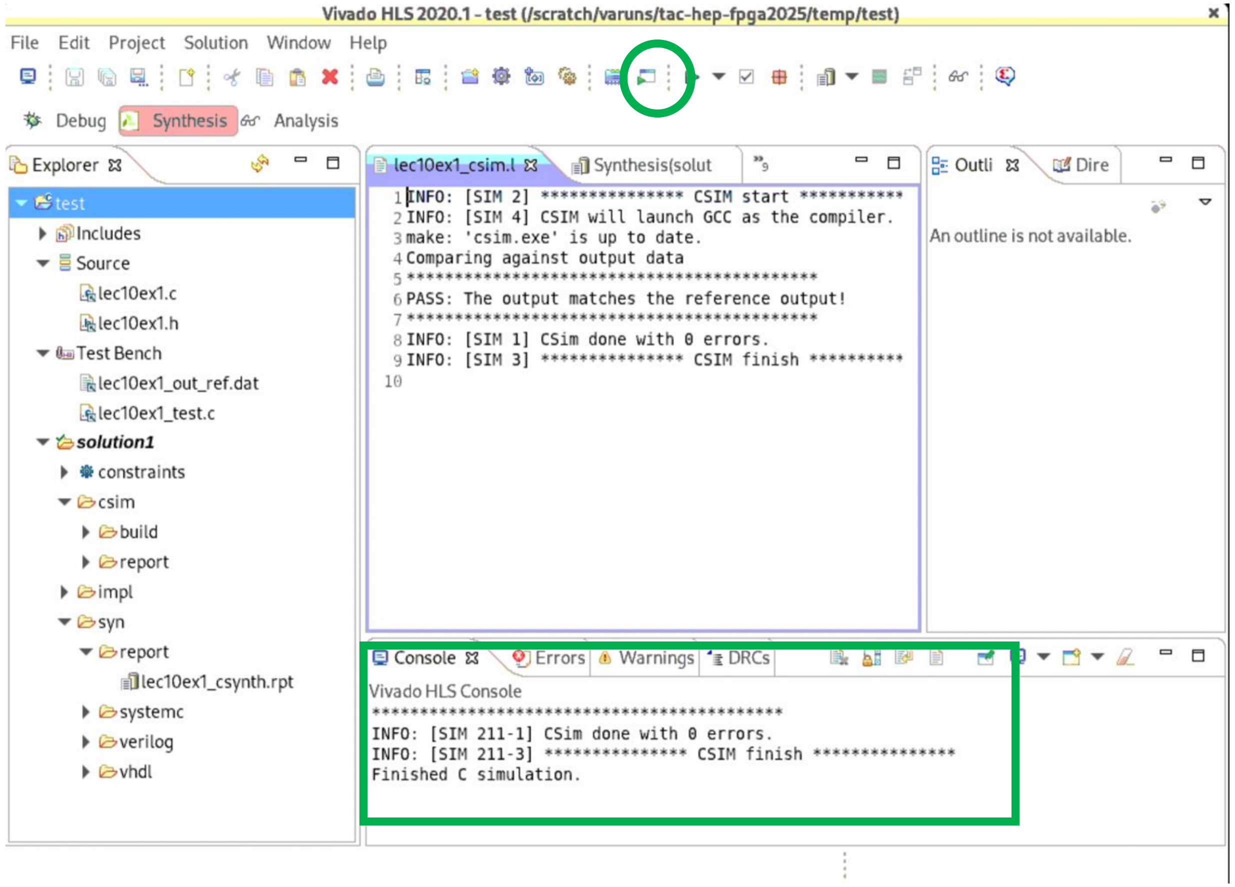
On-going…



```
INFO: [SIM 2] *************** CSIM start ***********
INFO: [SIM 4] CSIM will launch GCC as the compiler.
make: 'csim.exe' is up to date.
Comparing against output data
*******************************************
PASS: The output matches the reference output!
*******************************************
INFO: [SIM 1] CSim done with 0 errors.
INFO: [SIM 3] *************** CSIM finish **********
```

```
INFO: [HLS 200-10] In directory '/scratch/varuns/tac-hep-fpga2025/temp'
Sourcing Tcl script '/scratch/varuns/tac-hep-fpga2025/temp/test/solution1/csim.tcl'
INFO: [HLS 200-10] Opening project '/scratch/varuns/tac-hep-fpga2025/temp/test'.
INFO: [HLS 200-10] Opening solution '/scratch/varuns/tac-hep-fpga2025/temp/test/solution1
INFO: [SYN 201-201] Setting up clock 'default' with a period of 25ns.
```

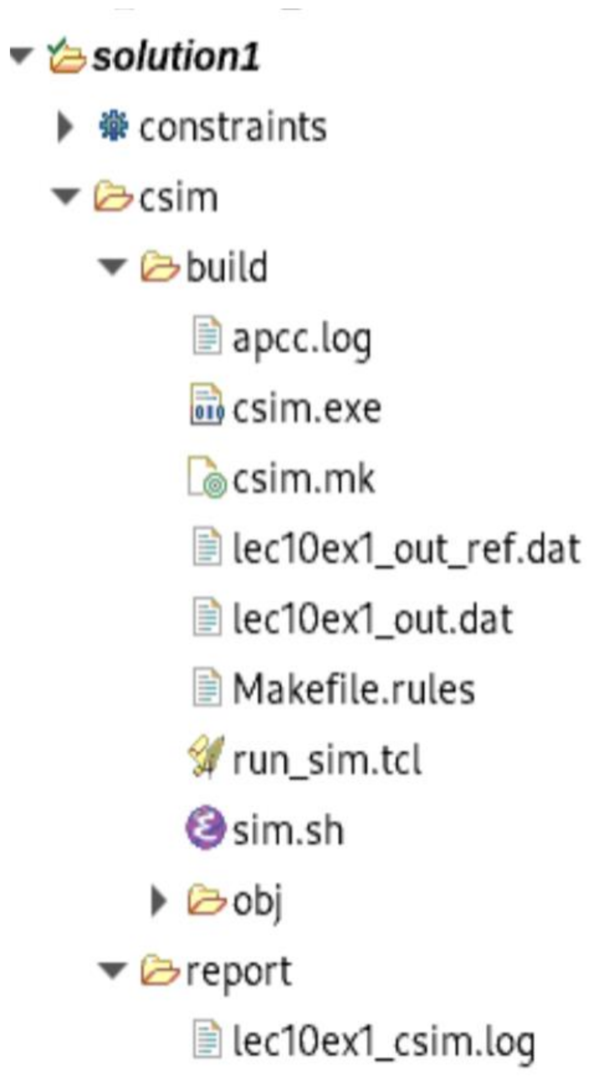
Finished…



```
INFO: [SIM 2] *************** CSIM start ***********
INFO: [SIM 4] CSIM will launch GCC as the compiler.
make: 'csim.exe' is up to date.
Comparing against output data
*******************************************
PASS: The output matches the reference output!
*******************************************
INFO: [SIM 1] CSim done with 0 errors.
INFO: [SIM 3] *************** CSIM finish **********
```

```
*******************************************
INFO: [SIM 211-1] CSim done with 0 errors.
INFO: [SIM 211-3] *************** CSIM finish ***************
Finished C simulation.
```

C – simulation output


- solution1
  - constraints
  - csim
    - build
      - apcc.log
      - csim.exe
      - csim.mk
      - lec10ex1_out_ref.dat
      - lec10ex1_out.dat
      - Makefile.rules
      - run_sim.tcl
      - sim.sh
      - obj
    - report
      - lec10ex1_csim.log

csim/build is the primary location for all files related to the C-simulation

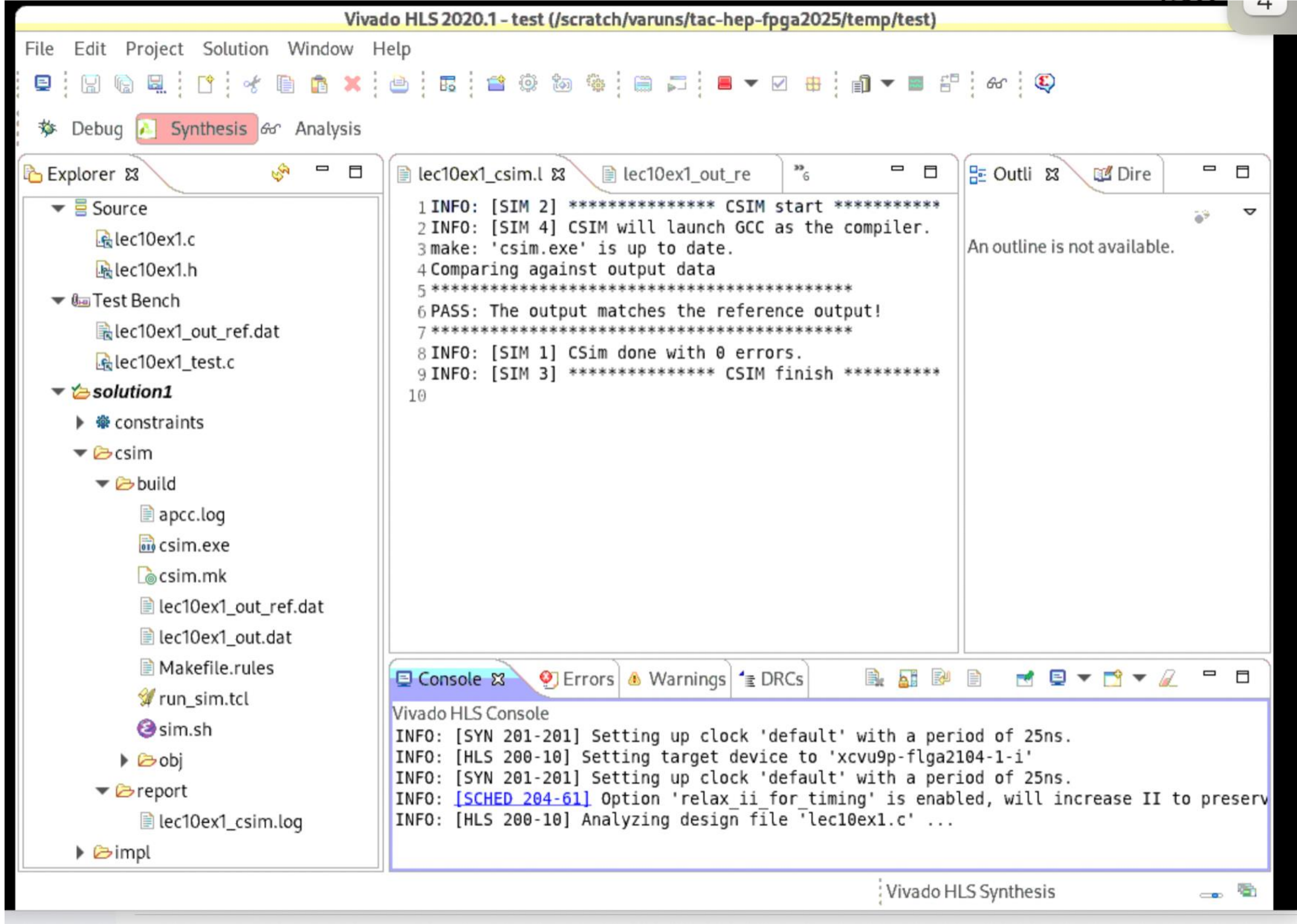
- Any files read by the test bench are copied to this folder
- The C executable file *csim.exe* is created and run in this folder
- Any files written by the test bench are created in this folder
  - Build Only option: exe file is not executed

Folder *csim/report* contains a log file of the C simulation

Next: execute synthesis

C-Synthesis

- Creating an Initial Solution
- Reviewing the Output of C Synthesis
- Analyzing the Results of Synthesis
- Creating a New Solution
- Applying Optimization Directives

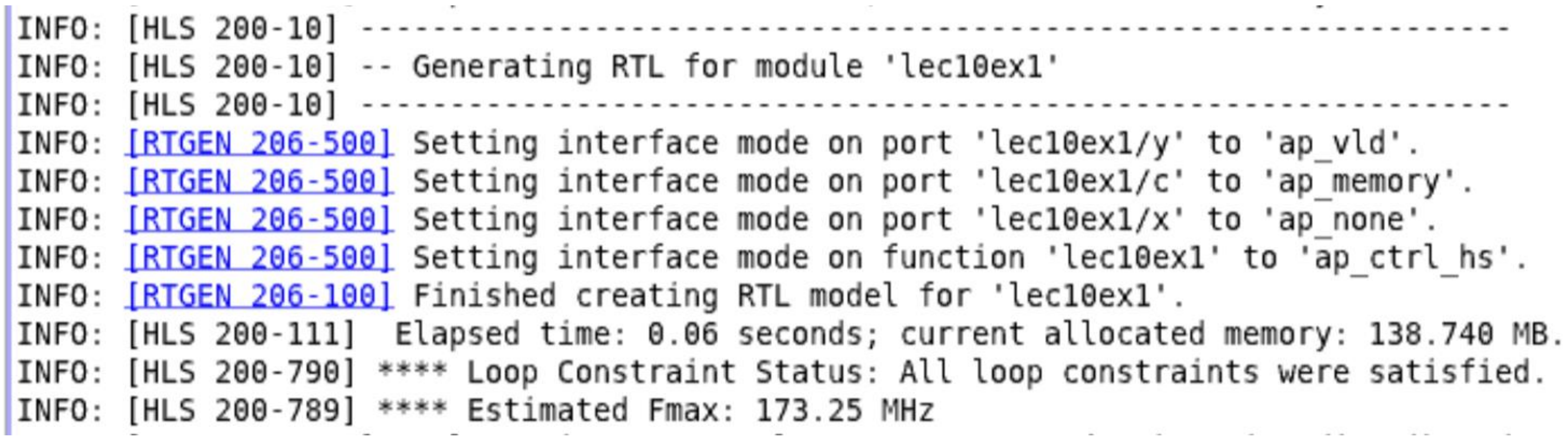


```
INFO: [SIM 2] *************** CSIM start ***********
INFO: [SIM 4] CSIM will launch GCC as the compiler.
make: 'csim.exe' is up to date.
Comparing against output data
*******************************************
PASS: The output matches the reference output!
*******************************************
INFO: [SIM 1] CSim done with 0 errors.
INFO: [SIM 3] *************** CSIM finish **********
```

```
INFO: [SYN 201-201] Setting up clock 'default' with a period of 25ns.
INFO: [HLS 200-10] Setting target device to 'xcvu9p-flga2104-1-i'
INFO: [SYN 201-201] Setting up clock 'default' with a period of 25ns.
INFO: [SCHED 204-61] Option 'relax_ii_for_timing' is enabled, will increase II to preserv
INFO: [HLS 200-10] Analyzing design file 'lec10ex1.c' ...
```

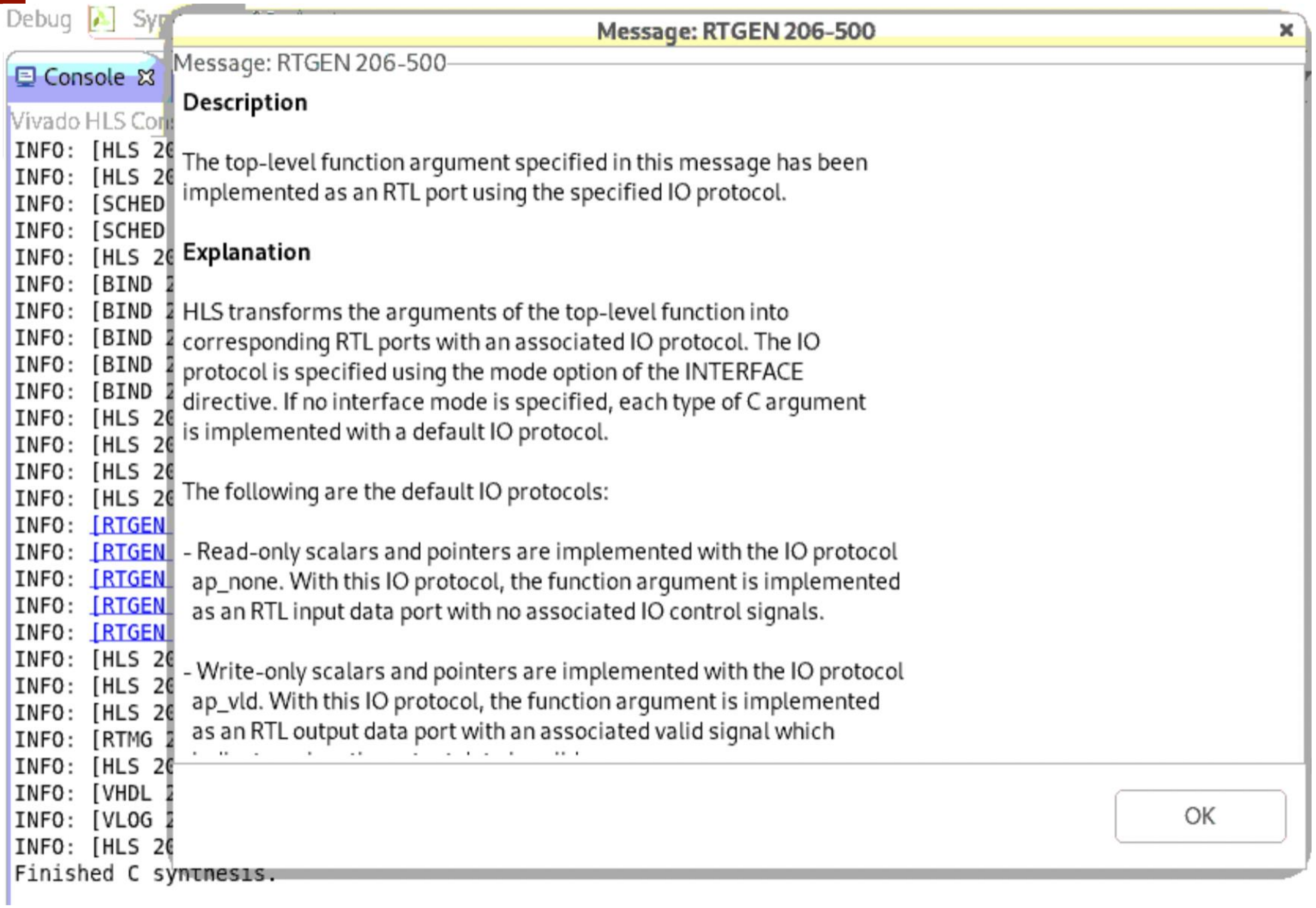
Console messages

- During the synthesis process messages are echoed to the console window
- The message include information messages showing how the synthesis process is proceeding



```
INFO: [HLS 200-10] ----------------------------------------------------------------
INFO: [HLS 200-10] -- Generating RTL for module 'lec10ex1'
INFO: [HLS 200-10] ----------------------------------------------------------------
INFO: [RTGEN 206-500] Setting interface mode on port 'lec10ex1/y' to 'ap_vld'.
INFO: [RTGEN 206-500] Setting interface mode on port 'lec10ex1/c' to 'ap_memory'.
INFO: [RTGEN 206-500] Setting interface mode on port 'lec10ex1/x' to 'ap_none'.
INFO: [RTGEN 206-500] Setting interface mode on function 'lec10ex1' to 'ap_ctrl_hs'.
INFO: [RTGEN 206-100] Finished creating RTL model for 'lec10ex1'.
INFO: [HLS 200-111]  Elapsed time: 0.06 seconds; current allocated memory: 138.740 MB.
INFO: [HLS 200-790] **** Loop Constraint Status: All loop constraints were satisfied.
INFO: [HLS 200-789] **** Estimated Fmax: 173.25 MHz
```

Console messages



```
INFO: [HLS 20
INFO: [HLS 20
INFO: [SCHED
INFO: [SCHED
INFO: [HLS 20
INFO: [BIND 2
INFO: [BIND 2
INFO: [BIND 2
INFO: [BIND 2
INFO: [BIND 2
INFO: [HLS 20
INFO: [HLS 20
INFO: [HLS 20
INFO: [HLS 20
INFO: [RTGEN
INFO: [RTGEN
INFO: [RTGEN
INFO: [RTGEN
INFO: [RTGEN
INFO: [HLS 20
INFO: [HLS 20
INFO: [HLS 20
INFO: [RTMG 2
INFO: [HLS 20
INFO: [VHDL 2
INFO: [VLOG 2
INFO: [HLS 20
Finished C synthesis.
```

Message: RTGEN 206-500

Description

The top-level function argument specified in this message has been implemented as an RTL port using the specified IO protocol.

Explanation

HLS transforms the arguments of the top-level function into corresponding RTL ports with an associated IO protocol. The IO protocol is specified using the mode option of the INTERFACE directive. If no interface mode is specified, each type of C argument is implemented with a default IO protocol.

The following are the default IO protocols:

- Read-only scalars and pointers are implemented with the IO protocol ap_none. With this IO protocol, the function argument is implemented as an RTL input data port with no associated IO control signals.

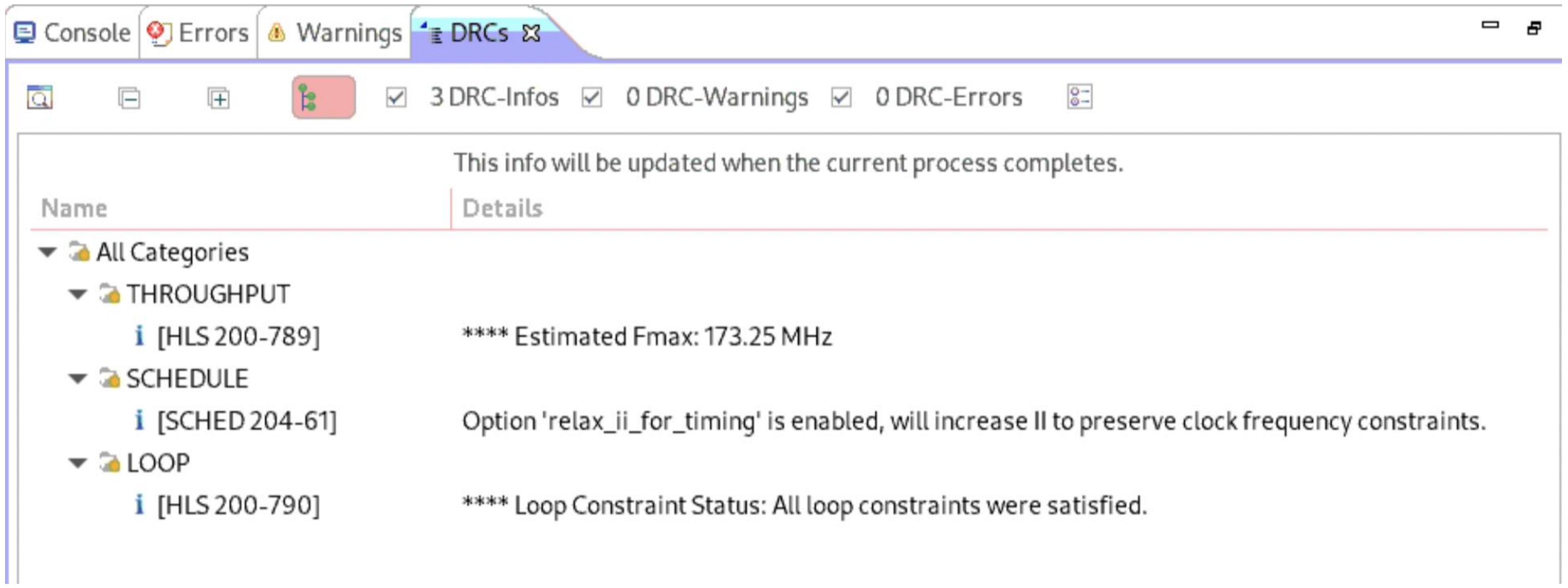
- Write-only scalars and pointers are implemented with the IO protocol ap_vld. With this IO protocol, the function argument is implemented as an RTL output data port with an associated valid signal which

Some messages may contain links to enhanced information

Clicking on this message provides more details on why the message was issued and possible resolutions

C – Synthesis finished

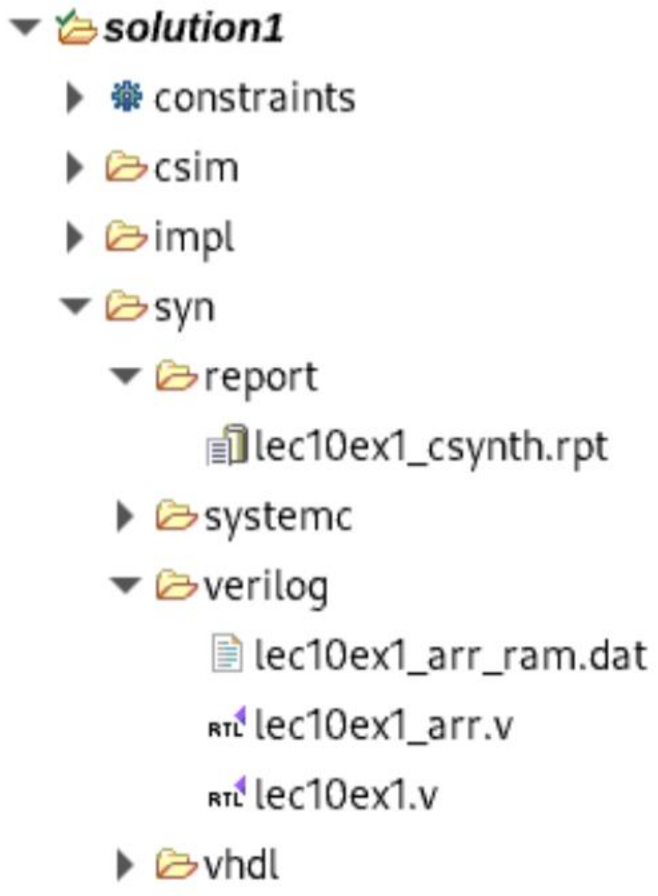
Design Rule Checking



This info will be updated when the current process completes.

Name	Details
All Categories	
THROUGHPUT	
[HLS 200-789]	**** Estimated Fmax: 173.25 MHz
SCHEDULE	
[SCHED 204-61]	Option 'relax_ii_for_timing' is enabled, will increase II to preserve clock frequency constraints.
LOOP	
[HLS 200-790]	**** Loop Constraint Status: All loop constraints were satisfied.

C-Synthesis: *Review the output*



```
▼ solution1
  ▶ constraints
  ▶ csim
  ▶ impl
  ▼ syn
    ▼ report
        lec10ex1_csynth.rpt
    ▶ systemc
    ▼ verilog
        lec10ex1_arr_ram.dat
        lec10ex1_arr.v
        lec10ex1.v
    ▶ vhdl
```

Folder *syn* is now available in the solution folder

report folder contains a report file for the top-level function and one for every sub-function in the design

verilog, *vhdl*, and *systemc* folders contain the output RTL files

The top-level file has the same name as the top-level function for synthesis

One RTL file for each function

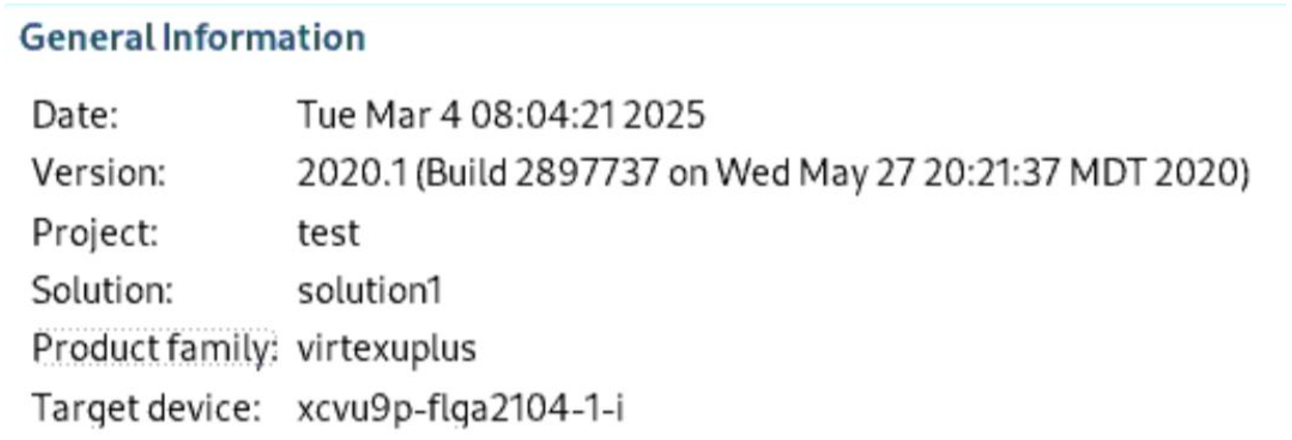
Might be additional RTL files to implement sub-blocks (block RAM, pipelined multipliers, etc)

Results of C Synthesis

- Two primary features provided to analyze the RTL design
  - **Synthesis reports**
    - The report provides details on both the performance and area of the RTL design
  - **Analysis Perspective**
    - Analysis Perspective provides both a tabular and graphical view of the design performance and resources and supports cross-referencing between both views

Synthesis Report

- General Information
  - When results were generated
  - Software version
  - Project name
  - Solution name
  - Technology details



General Information

Date:	Tue Mar 4 08:04:21 2025
Version:	2020.1 (Build 2897737 on Wed May 27 20:21:37 MDT 2020)
Project:	test
Solution:	solution1
Product family:	virtexuplus
Target device:	xcvu9p-flga2104-1-i

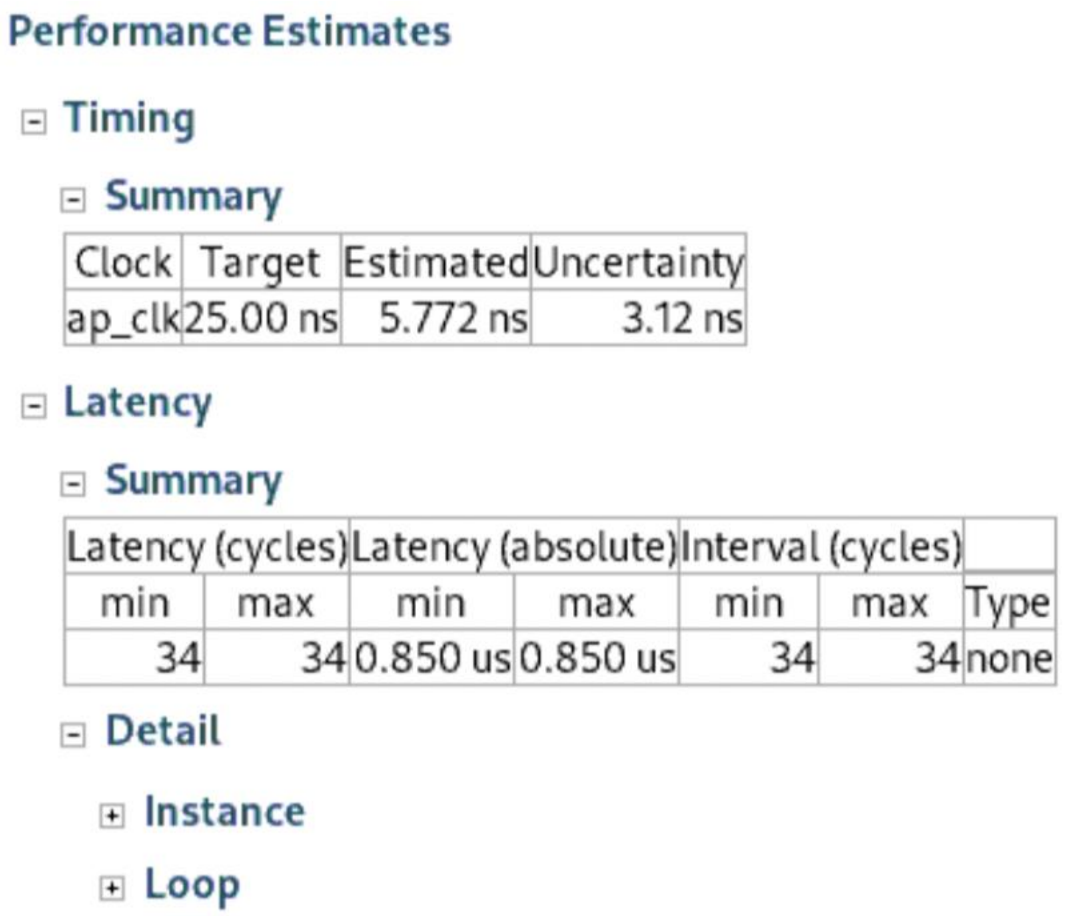
- Performance Estimates
- Utilization Estimates
- Interface Summary

Synthesis Report: Performance Estimates

Performance Estimates

Timing:

- Target clock frequency
- Clock uncertainty
- Estimate of the fastest achievable clock



Performance Estimates

Timing

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	25.00 ns	5.772 ns	3.12 ns

Latency

Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
34	34	0.850 us	0.850 us	34	34	none

Detail

Instance

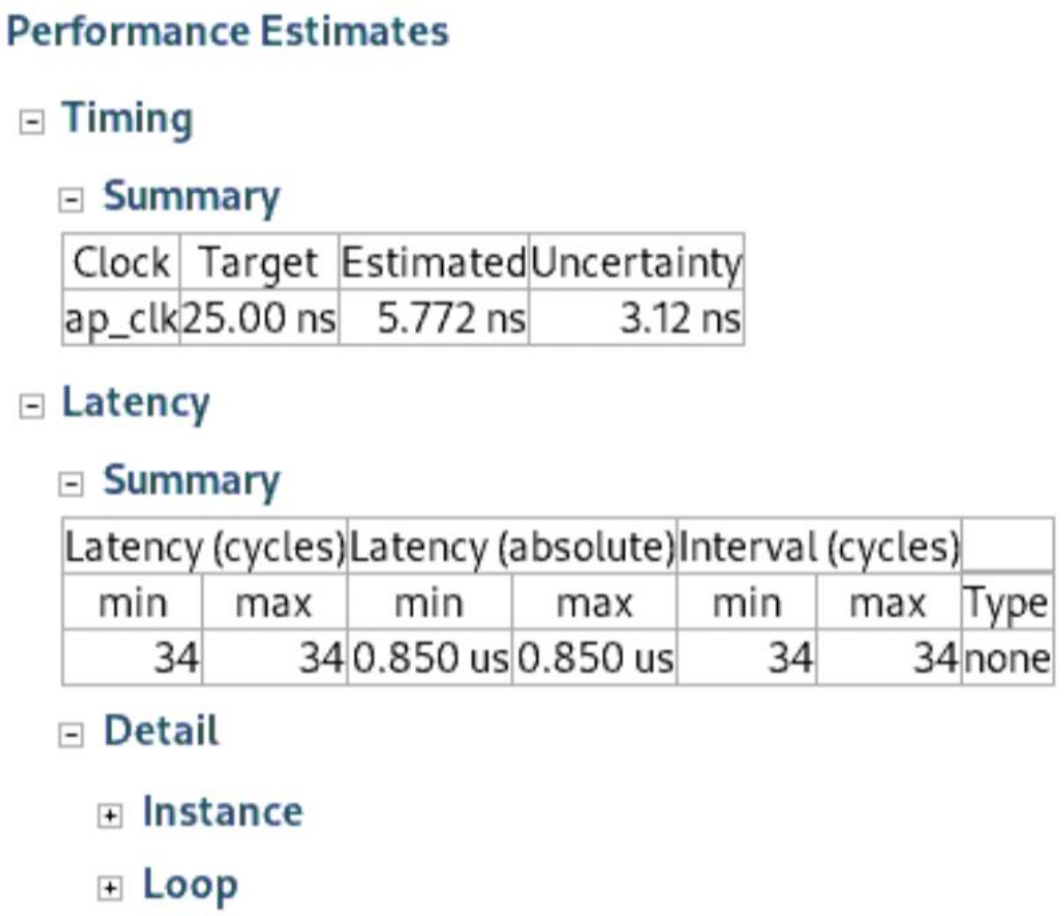
Loop

Synthesis Report: Performance Estimates

Performance Estimates

Latency (Summary):

- Reports the latency and initiation interval for this block and any sub blocks instantiated in this block
- Each sub function called at this level in the C source is an instance in this RTL block unless it was inlined
- **Latency** is the number of cycles it takes to produce the output
- **Initiation interval** is the number of clock cycles before new inputs can be applied
- In the absence of any PIPELINE directives the **latency** is one cycle less than the initiation interval
- Next input is read when the final output is written



Performance Estimates

⊟ Timing

⊟ Summary

Clock	Target	Estimated	Uncertainty
ap_clk	25.00 ns	5.772 ns	3.12 ns

⊟ Latency

⊟ Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
34	34	0.850 us	0.850 us	34	34	none

⊟ Detail

⊞ Instance

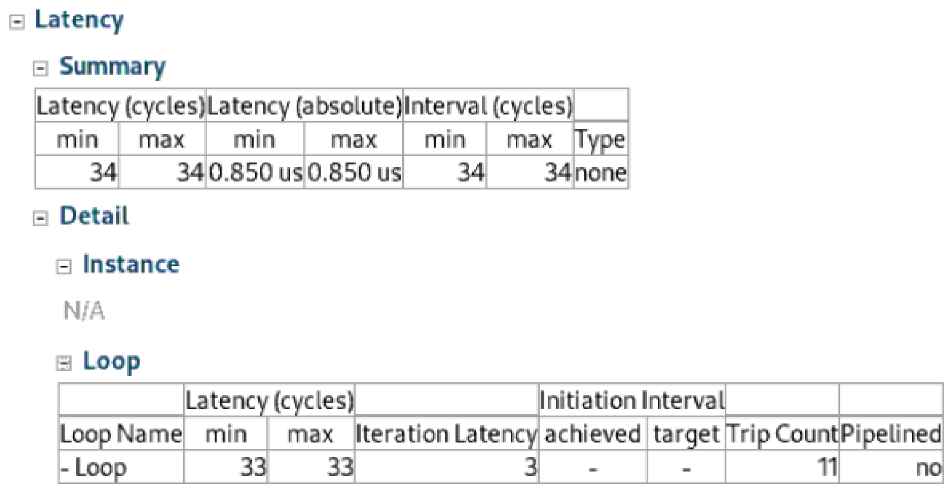
⊞ Loop

Synthesis Report: Performance Estimates

Performance Estimates

Latency (Details):

- The latency and initiation interval for the instances sub functions and loops in this block
  - If any loops contain sub loops the loop hierarchy is shown
- The min and max latency values indicate the latency to execute all iterations of the loop.
  - Presence of conditional branches in the code might make the min and max different
- The Iteration Latency is the latency for a single iteration of the loop
- If the loop has a variable latency, the latency values cannot be determined and are shown as a question mark (?)
- Any specified target initiation interval is shown beside the actual initiation interval achieved
- The tripcount shows the total number of loop iterations



⊟ Latency

⊟ Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
34	34	0.850 us	0.850 us	34	34	none

⊟ Detail

⊟ Instance

N/A

⊟ Loop

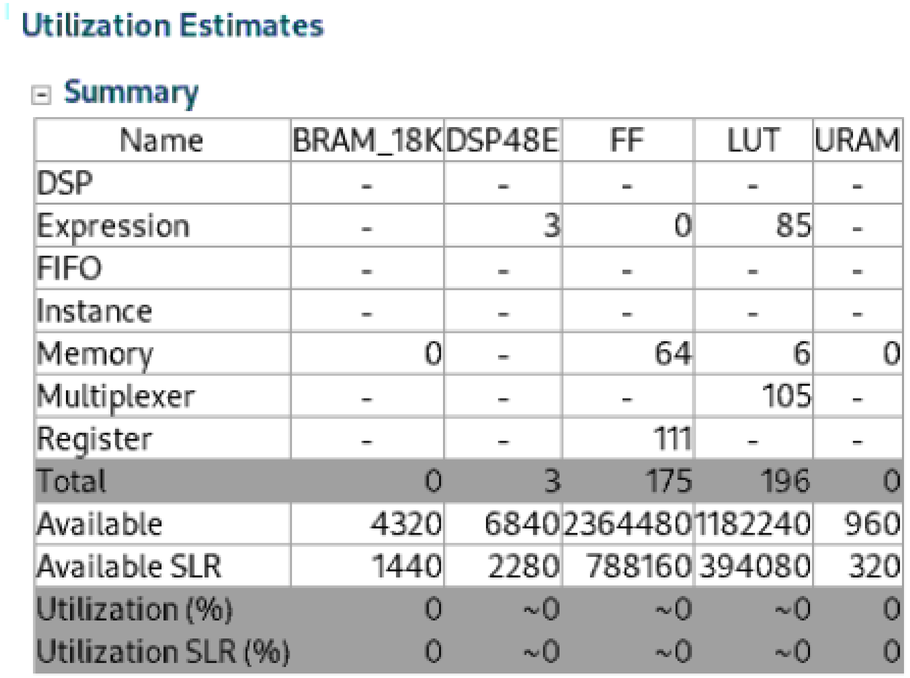
	Latency (cycles)			Initiation Interval			
Loop Name	min	max	Iteration Latency	achieved	target	Trip Count	Pipelined
- Loop	33	33	3	-	-	11	no

Synthesis Report: Utilization Estimates

Utilization Estimates:

Summary

This part of the report shows the resources: LUTS, Flip-Flops, DSPs used to implement the design



Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT	URAM
DSP	-	-	-	-	-
Expression	-	3	0	85	-
FIFO	-	-	-	-	-
Instance	-	-	-	-	-
Memory	0	-	64	6	0
Multiplexer	-	-	-	105	-
Register	-	-	111	-	-
Total	0	3	175	196	0
Available	4320	6840	2364480	1182240	960
Available SLR	1440	2280	788160	394080	320
Utilization (%)	0	~0	~0	~0	0
Utilization SLR (%)	0	~0	~0	~0	0

Synthesis Report: Utilization Estimates

Utilization Estimates: Details

Instance:

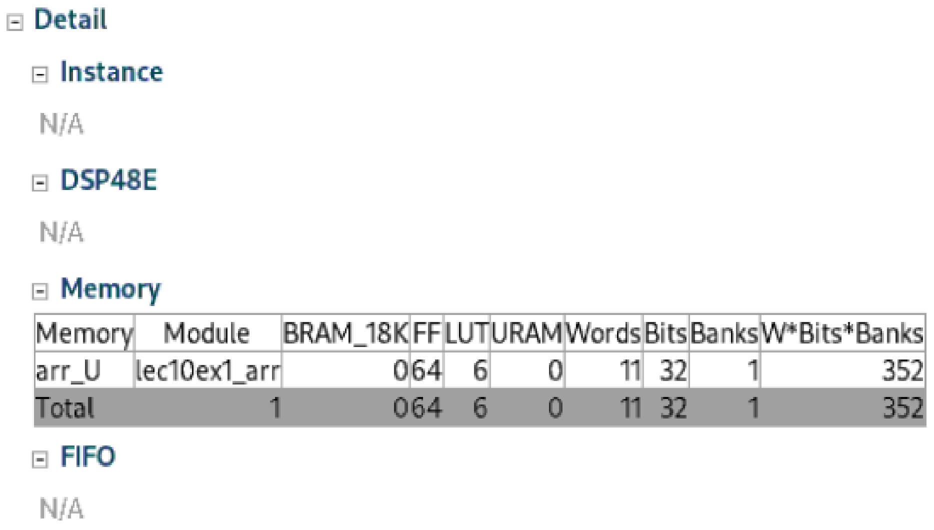
- The resources specified here are used by the sub blocks instantiated at this level of the hierarchy
- If the design only has no RTL hierarchy there are no instances reported
- If any instances are present clicking on the name of the instance opens the synthesis report for that instance

Memory

- The resources listed here are those used in the implementation of memories at this level of the hierarchy
- Vivado HLS reports a single port BRAM as using one bank of memory and reports a dual port BRAM as using two banks of memory

FIFO

- The resources listed here are those used in the implementation of any FIFOs implemented at this level of the hierarchy



Detail

Instance

N/A

DSP48E

N/A

Memory

Memory	Module	BRAM_18K	FF	LUT	URAM	Words	Bits	Banks	W*Bits*Banks
arr_U	lec10ex1_arr	0	64	6	0	11	32	1	352
Total	1	0	64	6	0	11	32	1	352

FIFO

N/A

Synthesis Report: Utilization Estimates

Utilization Estimates: Details

Expression:

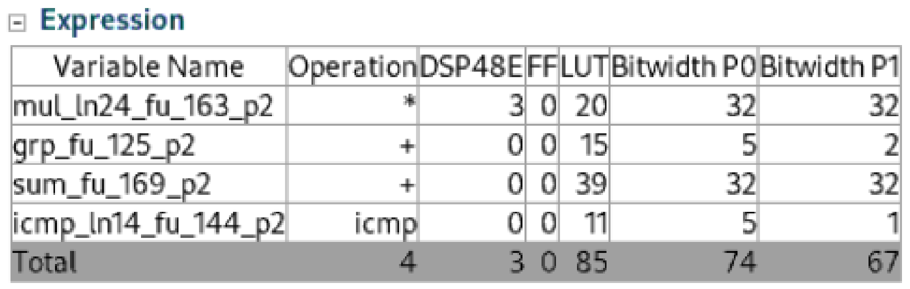
- Resources used by any expressions such as multipliers adders and comparators at the current level of hierarchy
- Bit widths of the input ports to the expressions are shown

Multiplexors:

- Resources used to implement multiplexors at this level of hierarchy
- Input widths of the multiplexors are shown

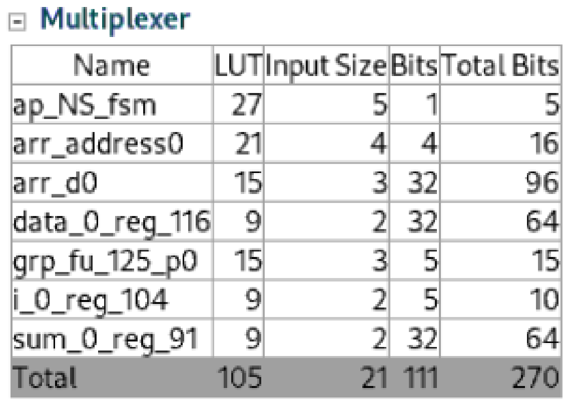
Register:

- List of all registers at this level of hierarchy
- The report includes the register bit widths



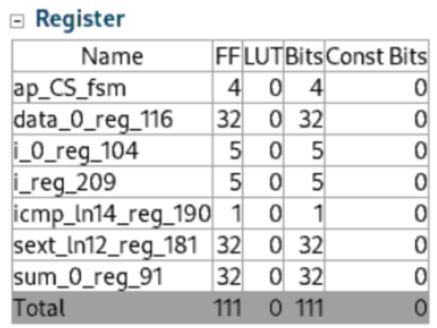
Expression

Variable Name	Operation	DSP48E	FF	LUT	Bitwidth P0	Bitwidth P1
mul_ln24_fu_163_p2	*	3	0	20	32	32
grp_fu_125_p2	+	0	0	15	5	2
sum_fu_169_p2	+	0	0	39	32	32
icmp_ln14_fu_144_p2	icmp	0	0	11	5	1
Total	4	3	0	85	74	67



Multiplexer

Name	LUT	Input Size	Bits	Total Bits
ap_NS_fsm	27	5	1	5
arr_address0	21	4	4	16
arr_d0	15	3	32	96
data_0_reg_116	9	2	32	64
grp_fu_125_p0	15	3	5	15
i_0_reg_104	9	2	5	10
sum_0_reg_91	9	2	32	64
Total	105	21	111	270



Register

Name	FF	LUT	Bits	Const Bits
ap_CS_fsm	4	0	4	0
data_0_reg_116	32	0	32	0
i_0_reg_104	5	0	5	0
i_reg_209	5	0	5	0
icmp_ln14_reg_190	1	0	1	0
sext_ln12_reg_181	32	0	32	0
sum_0_reg_91	32	0	32	0
Total	111	0	111	0

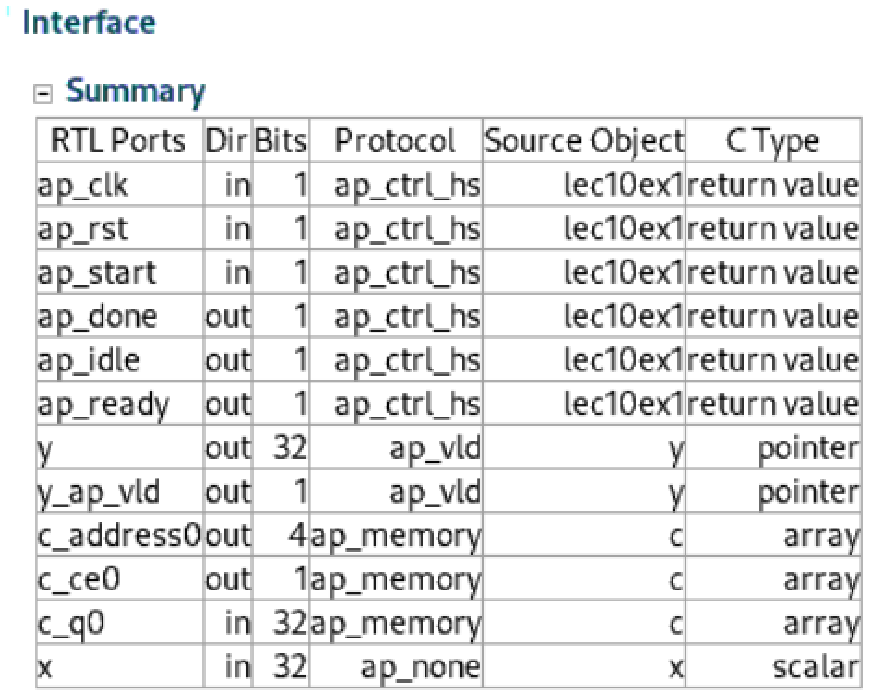
Synthesis Report: Interface Summary

Interface

- This section shows how the function arguments have been synthesized into RTL ports
- The RTL port names are grouped with their protocol
- **Source object:** RTL ports created when that source object is synthesized with the stated I/O protocol

- The design has a clock and reset port

- Synthesis has automatically added some block level control ports : *ap_start*, *ap_done*, *ap_idle* and *ap_ready*

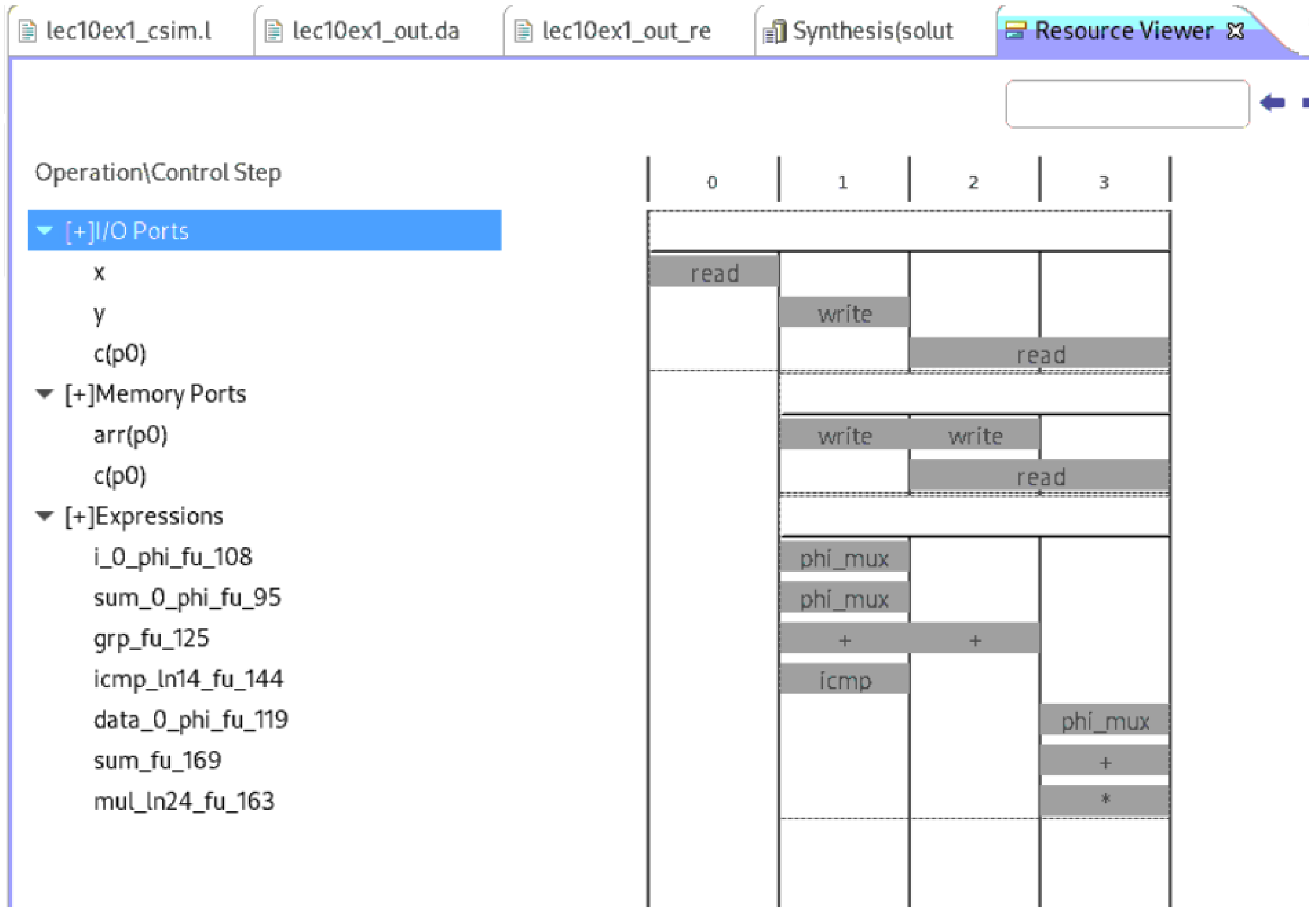


Interface

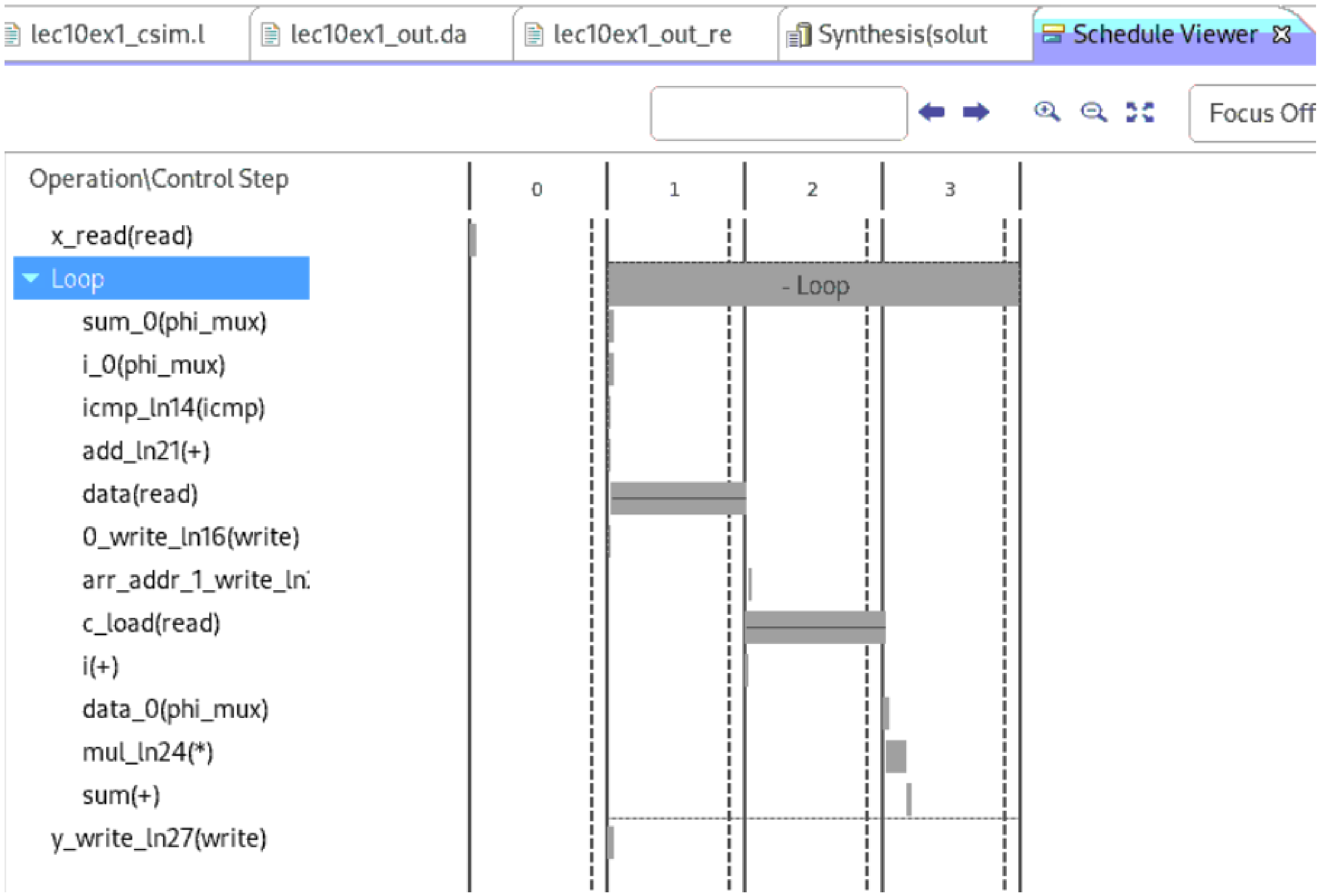
Summary

RTL Ports	Dir	Bits	Protocol	Source Object	C Type
ap_clk	in	1	ap_ctrl_hs	lec10ex1	return value
ap_rst	in	1	ap_ctrl_hs	lec10ex1	return value
ap_start	in	1	ap_ctrl_hs	lec10ex1	return value
ap_done	out	1	ap_ctrl_hs	lec10ex1	return value
ap_idle	out	1	ap_ctrl_hs	lec10ex1	return value
ap_ready	out	1	ap_ctrl_hs	lec10ex1	return value
y	out	32	ap_vld	y	pointer
y_ap_vld	out	1	ap_vld	y	pointer
c_address0	out	4	ap_memory	c	array
c_ce0	out	1	ap_memory	c	array
c_q0	in	32	ap_memory	c	array
x	in	32	ap_none	x	scalar

Analysis Perspective: Resource view



Analysis Perspective: Schedule view



Reminder: Assignments

- Assignment-1 (13-02-2025)
- Assignment-2 (18-02-2025)
- Assignment-3 (27-02-2025)

Uploaded to cernbox: https://cernbox.cern.ch/s/gmUqRDHTxDLqx4M

Send via email: **varun.sharma@cern.ch**

Submit in 2 weeks from date of assignment

TAC-HEP 2025

Questions?

Acknowledgements:
- Some of these slides are from Isobel Ojalvo

Jargons

- **ICs - Integrated chip:** assembly of hundreds of millions of transistors on a minor chip
- **PCB:** Printed Circuit Board
- **LUT - Look Up Table aka 'logic'** - generic functions on small bitwidth inputs. Combine many to build the algorithm
- **FF - Flip Flops** - control the flow of data with the clock pulse. Used to build the pipeline and achieve high throughput
- **DSP - Digital Signal Processor** - performs multiplication and other arithmetic in the FPGA
- **BRAM - Block RAM** - hardened RAM resource. More efficient memories than using LUTs for more than a few elements
- **PCIe or PCI-E - Peripheral Component Interconnect Express**: is a serial expansion bus standard for connecting a computer to one or more peripheral devices
- **InfiniBand** is a computer networking communications standard used in high-performance computing that features very high throughput and very low latency
- **HLS** - High Level Synthesis - compiler for C, C++, SystemC into FPGA IP cores
- **HDL** - Hardware Description Language - low level language for describing circuits
- **RTL** - Register Transfer Level - the very low level description of the function and connection of logic gates
- **FIFO** – First In First Out memory
- **Latency** - time between starting processing and receiving the result
  - Measured in clock cycles or seconds
- **II - Initiation Interval** - time from accepting first input to accepting next input