V.GOKUL KUMAR
FUTURE WIRELESS COMMUNICATIONS (FWC)
Indian Institute of Technology Hyderabad
Email: velicharla@outlook.com

Date:10-04-2023

Hls ASSIGNMENT-7

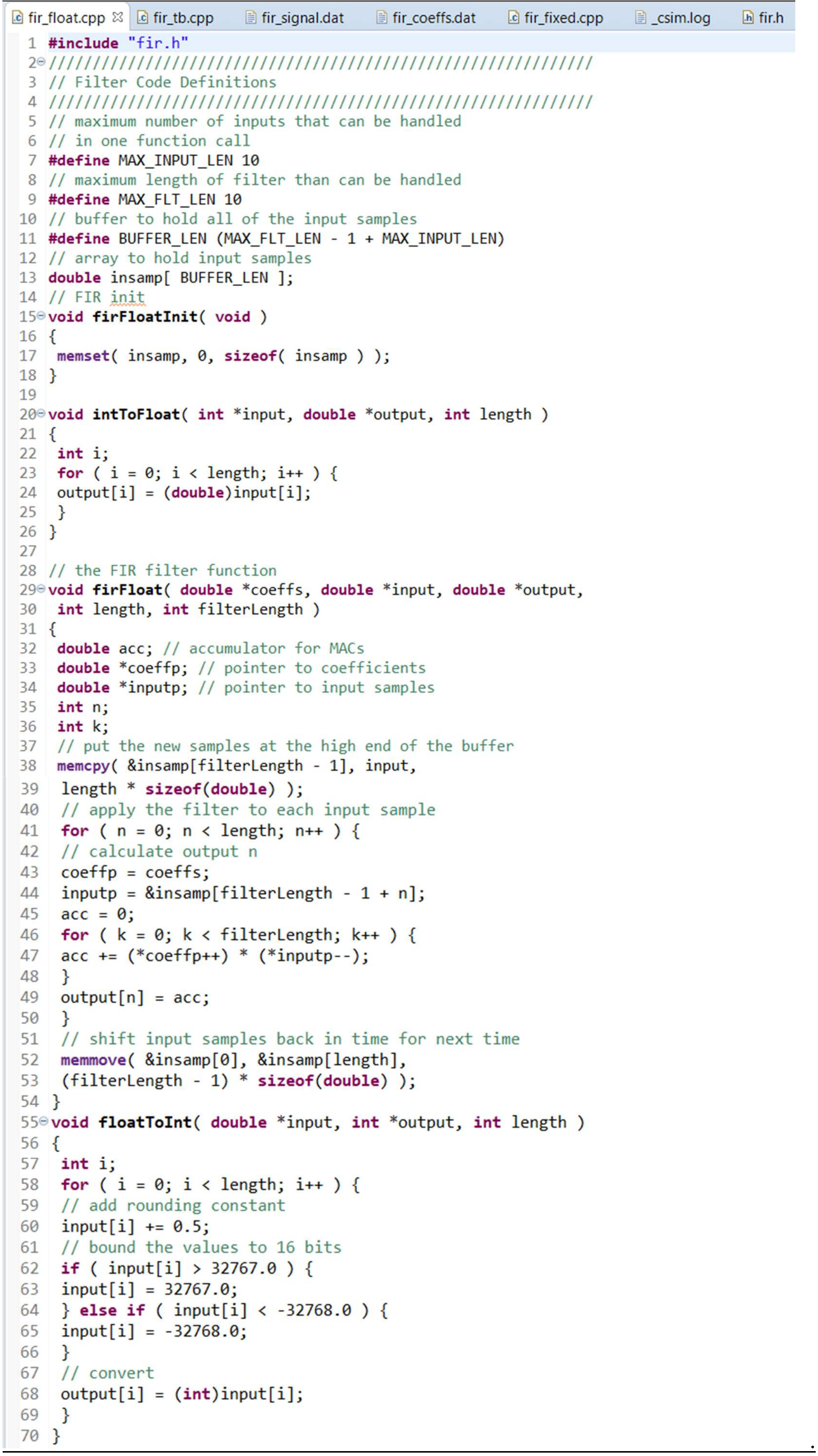
Implement a module in HLS that models a basic FIR filter as specified on this web page: https://sestevenson.wordpress.com/implementation-of-fir-filtering-in-c-part-1/. The web page also has C code which you should use as reference for your HLS design. Your task is to model the design in the most efficient manner possible for the hardware (e.g., small clock period, lower initiation interval, less resource consumption, etc.). After designing, you should compare your output against the reference output generated by the C code for the same set of input vectors, using a self-checking testbench that allows for a 5% difference in output values generated by the C code and the HLS code. Use at least two different input vectors. Also, the HLS design should use appropriate fixed-point format instead of floating-point format wherever applicable. The C code from the website can be integrated as part of your HLS testbench if you name both the design modules differently, for e.g., firFloat for C module and firFixed for HLS module. This will enable you to pass the input vectors to both the modules and compare the outputs, in a single testbench.

Note that a good C code will probably never be good for the hardware HLS design. So don't copy the C design and expect it to work optimally and even correctly on the hardware.

The purpose of this assignment is to make you use your learning till now to design a basic module. Report all the source code files, self-checking testbench files, input and output files, and "ALL" the HLS reports.

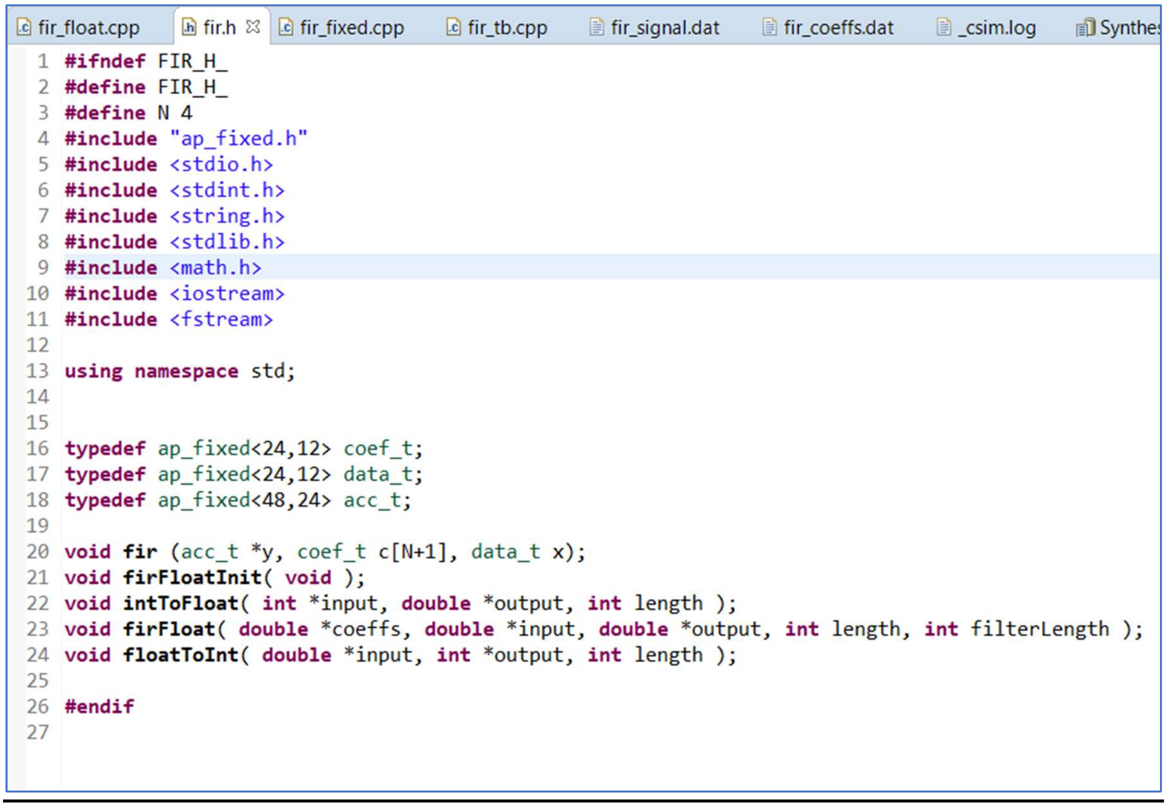
Design:

Main function: (C-CODE)



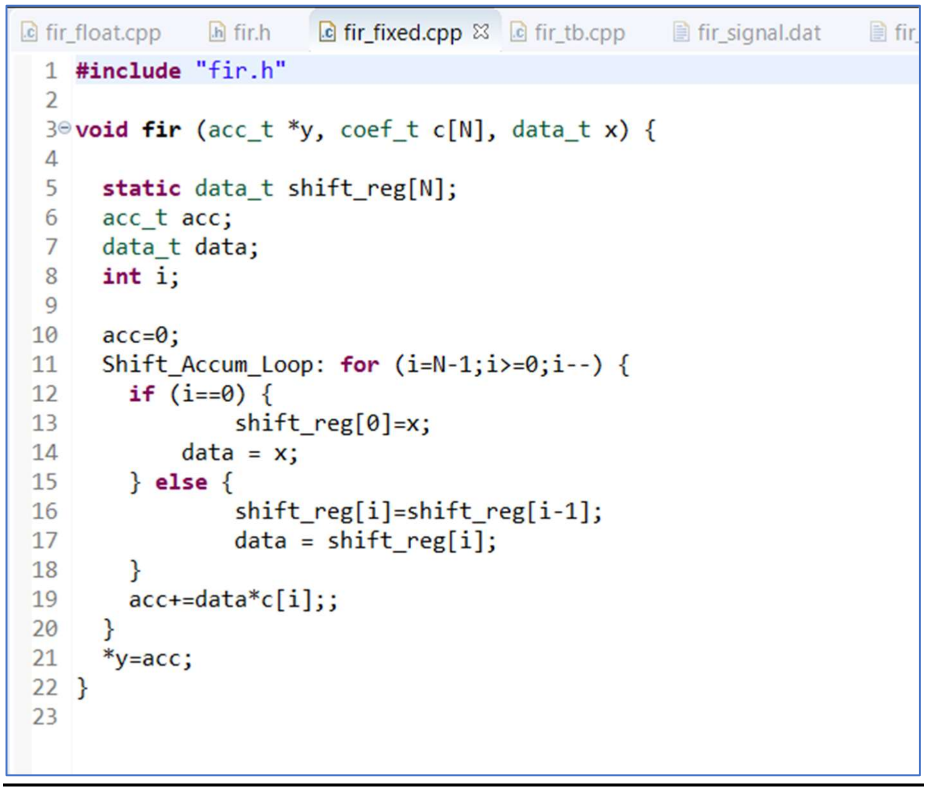
```
#include "fir.h"
////////////////////////////////////////////////////////////
// Filter Code Definitions
////////////////////////////////////////////////////////////
// maximum number of inputs that can be handled
// in one function call
#define MAX_INPUT_LEN 10
// maximum length of filter than can be handled
#define MAX_FLT_LEN 10
// buffer to hold all of the input samples
#define BUFFER_LEN (MAX_FLT_LEN - 1 + MAX_INPUT_LEN)
// array to hold input samples
double insamp[ BUFFER_LEN ];
// FIR init
void firFloatInit( void )
{
 memset( insamp, 0, sizeof( insamp ) );
}

void intToFloat( int *input, double *output, int length )
{
 int i;
 for ( i = 0; i < length; i++ ) {
 output[i] = (double)input[i];
 }
}

// the FIR filter function
void firFloat( double *coeffs, double *input, double *output,
 int length, int filterLength )
{
 double acc; // accumulator for MACs
 double *coeffp; // pointer to coefficients
 double *inputp; // pointer to input samples
 int n;
 int k;
 // put the new samples at the high end of the buffer
 memcpy( &insamp[filterLength - 1], input,
 length * sizeof(double) );
 // apply the filter to each input sample
 for ( n = 0; n < length; n++ ) {
 // calculate output n
 coeffp = coeffs;
 inputp = &insamp[filterLength - 1 + n];
 acc = 0;
 for ( k = 0; k < filterLength; k++ ) {
 acc += (*coeffp++) * (*inputp--);
 }
 output[n] = acc;
 }
 // shift input samples back in time for next time
 memmove( &insamp[0], &insamp[length],
 (filterLength - 1) * sizeof(double) );
}
void floatToInt( double *input, int *output, int length )
{
 int i;
 for ( i = 0; i < length; i++ ) {
 // add rounding constant
 input[i] += 0.5;
 // bound the values to 16 bits
 if ( input[i] > 32767.0 ) {
 input[i] = 32767.0;
 } else if ( input[i] < -32768.0 ) {
 input[i] = -32768.0;
 }
 // convert
 output[i] = (int)input[i];
 }
}
```

Header File:



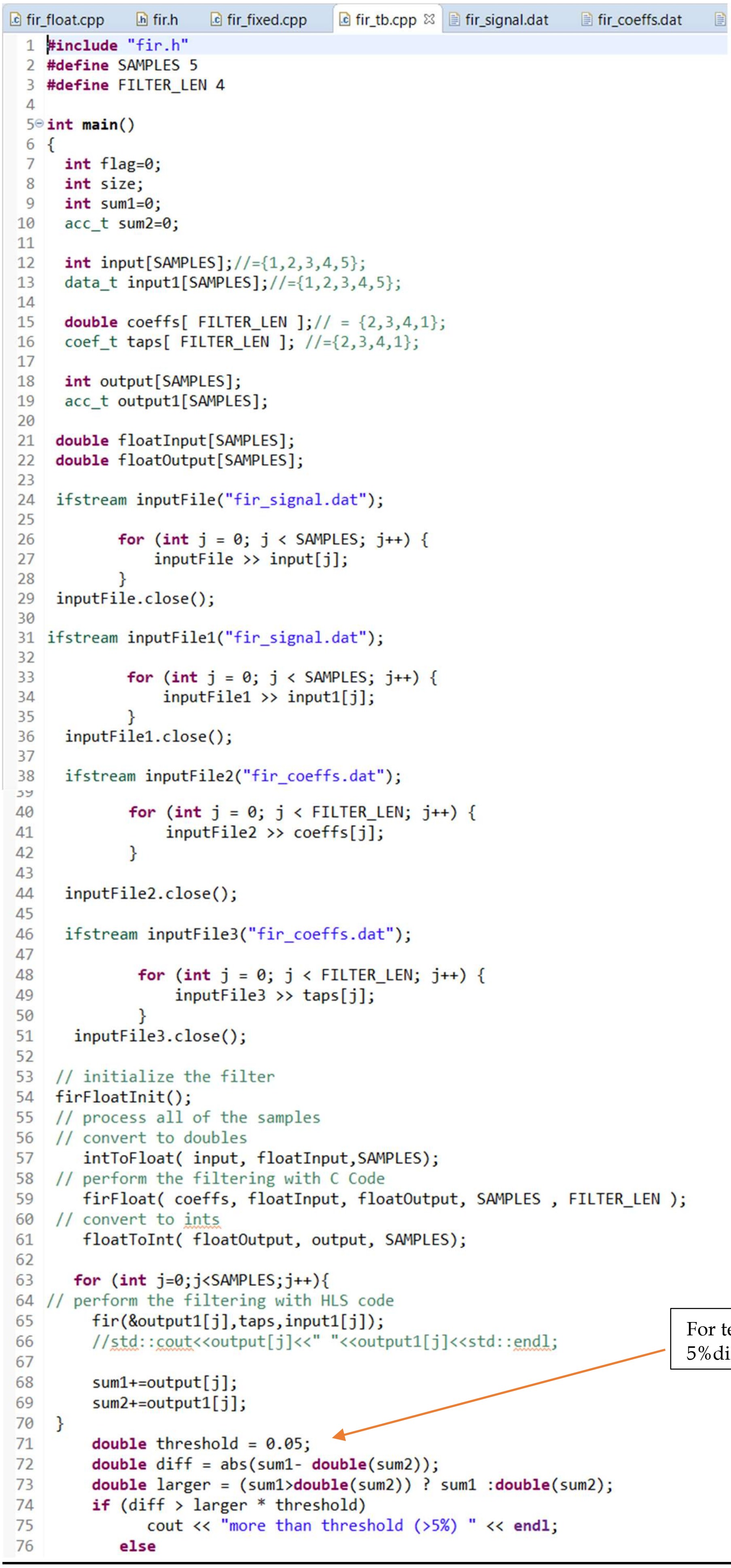
```
#ifndef FIR_H_
#define FIR_H_
#define N 4
#include "ap_fixed.h"
#include <stdio.h>
#include <stdint.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <iostream>
#include <fstream>

using namespace std;


typedef ap_fixed<24,12> coef_t;
typedef ap_fixed<24,12> data_t;
typedef ap_fixed<48,24> acc_t;

void fir (acc_t *y, coef_t c[N+1], data_t x);
void firFloatInit( void );
void intToFloat( int *input, double *output, int length );
void firFloat( double *coeffs, double *input, double *output, int length, int filterLength );
void floatToInt( double *input, int *output, int length );

#endif
```

Main function: (HLS-CODE)



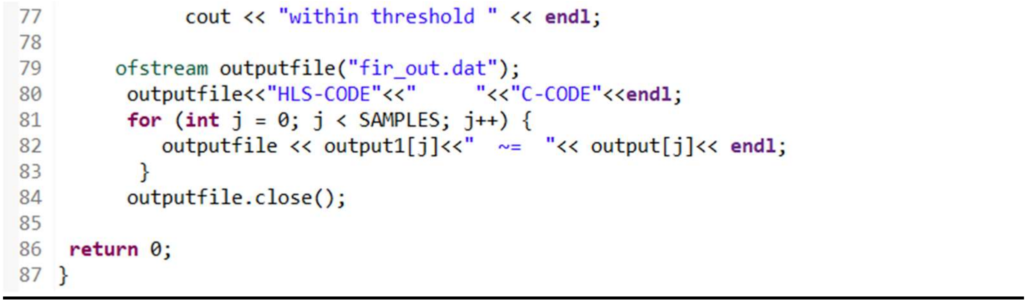
```
#include "fir.h"

void fir (acc_t *y, coef_t c[N], data_t x) {

  static data_t shift_reg[N];
  acc_t acc;
  data_t data;
  int i;

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

Self-checking Test Bench:



```
#include "fir.h"
#define SAMPLES 5
#define FILTER_LEN 4

int main()
{
  int flag=0;
  int size;
  int sum1=0;
  acc_t sum2=0;

  int input[SAMPLES];//={1,2,3,4,5};
  data_t input1[SAMPLES];//={1,2,3,4,5};

  double coeffs[ FILTER_LEN ];// = {2,3,4,1};
  coef_t taps[ FILTER_LEN ]; //={2,3,4,1};

  int output[SAMPLES];
  acc_t output1[SAMPLES];

 double floatInput[SAMPLES];
 double floatOutput[SAMPLES];

 ifstream inputFile("fir_signal.dat");

        for (int j = 0; j < SAMPLES; j++) {
            inputFile >> input[j];
        }
 inputFile.close();

ifstream inputFile1("fir_signal.dat");

         for (int j = 0; j < SAMPLES; j++) {
             inputFile1 >> input1[j];
         }
  inputFile1.close();

  ifstream inputFile2("fir_coeffs.dat");

         for (int j = 0; j < FILTER_LEN; j++) {
             inputFile2 >> coeffs[j];
         }

  inputFile2.close();

  ifstream inputFile3("fir_coeffs.dat");

          for (int j = 0; j < FILTER_LEN; j++) {
              inputFile3 >> taps[j];
          }
   inputFile3.close();

 // initialize the filter
 firFloatInit();
 // process all of the samples
 // convert to doubles
    intToFloat( input, floatInput,SAMPLES);
 // perform the filtering with C Code
    firFloat( coeffs, floatInput, floatOutput, SAMPLES , FILTER_LEN );
 // convert to ints
    floatToInt( floatOutput, output, SAMPLES);

   for (int j=0;j<SAMPLES;j++){
// perform the filtering with HLS code
     fir(&output1[j],taps,input1[j]);
     //std::cout<<output[j]<<" "<<output1[j]<<std::endl;

     sum1+=output[j];
     sum2+=output1[j];
 }
     double threshold = 0.05;
     double diff = abs(sum1- double(sum2));
     double larger = (sum1>double(sum2)) ? sum1 :double(sum2);
     if (diff > larger * threshold)
           cout << "more than threshold (>5%) " << endl;
        else
```

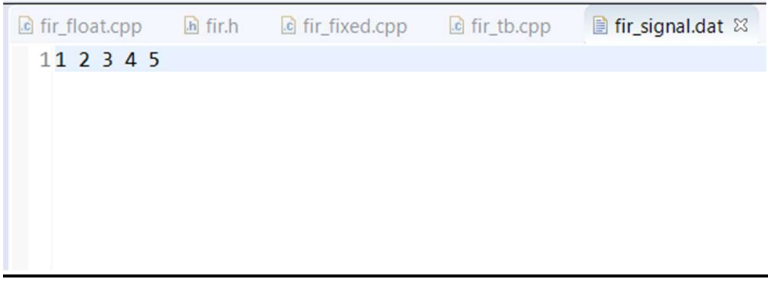
For testing 5%difference



```
            cout << "within threshold " << endl;

      ofstream outputfile("fir_out.dat");
       outputfile<<"HLS-CODE"<<"     "<<"C-CODE"<<endl;
       for (int j = 0; j < SAMPLES; j++) {
          outputfile << output1[j]<<"  ~=  "<< output[j]<< endl;
        }
       outputfile.close();

 return 0;
}
```

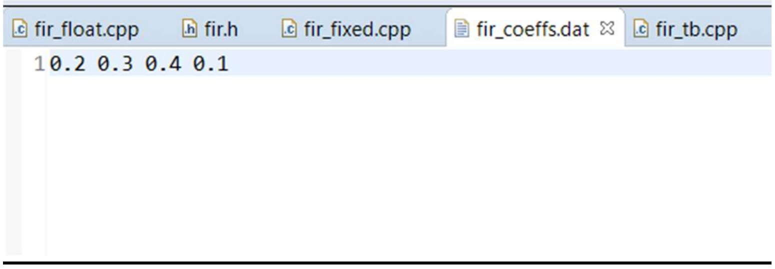
Input Files: (Test1)

Signal sample file:



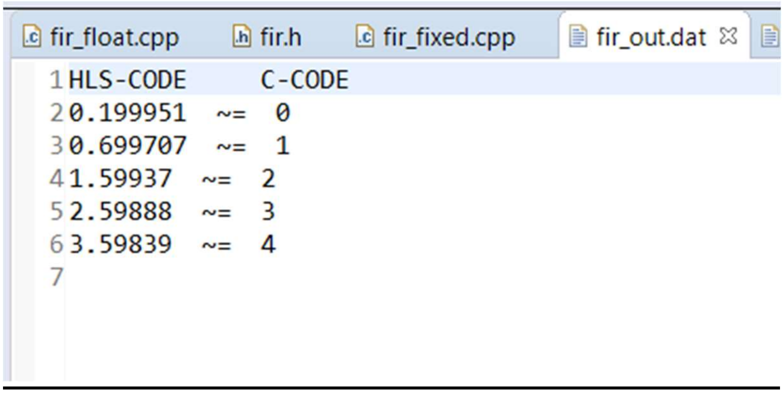
```
1 2 3 4 5
```

Filter coefficients file:



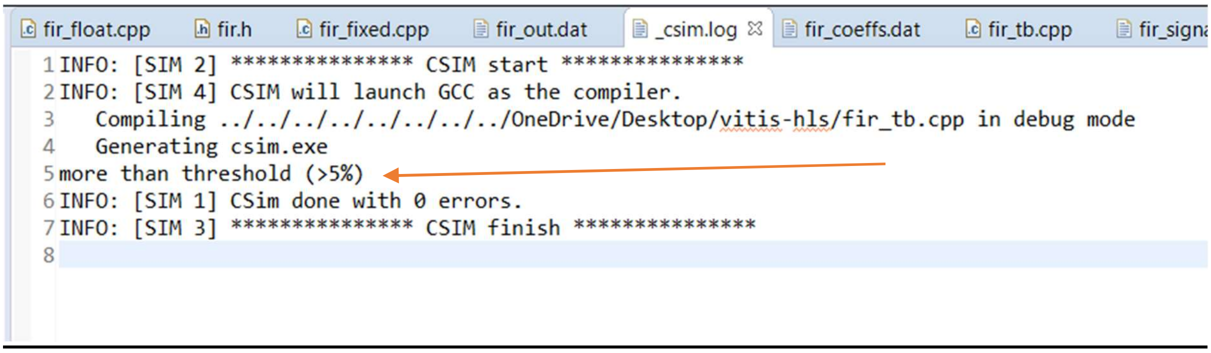
```
0.2 0.3 0.4 0.1
```

Output File:



```
HLS-CODE     C-CODE
0.199951   ~=  0
0.699707   ~=  1
1.59937  ~=  2
2.59888  ~=  3
3.59839  ~=  4

```

C simulation printed output:



```
INFO: [SIM 2] *************** CSIM start ***************
INFO: [SIM 4] CSIM will launch GCC as the compiler.
   Compiling ../../../../../../../../OneDrive/Desktop/vitis-hls/fir_tb.cpp in debug mode
   Generating csim.exe
more than threshold (>5%)
INFO: [SIM 1] CSim done with 0 errors.
INFO: [SIM 3] *************** CSIM finish ***************
```

Theoretical:

we have a 5-sample input signal: {1, 2, 3, 4, 5}
filter coefficients: {0.2, 0.3, 0.4, 0.1}. This filter has a length of 4 taps.

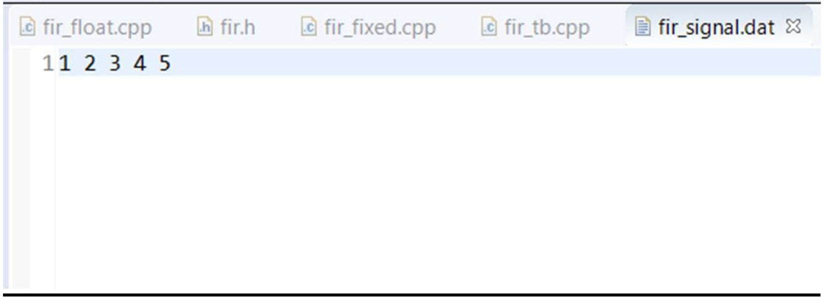
Output [0] = (0.2 * 1) + (0.3 * 0) + (0.4 * 0) + (0.1 * 0) = 0.2
Output [1] = (0.2 * 2) + (0.3 * 1) + (0.4 * 0) + (0.1 * 0) = 0.7
Output [2] = (0.2 * 3) + (0.3 * 2) + (0.4 * 1) + (0.1 * 0) = 1.6
Output [3] = (0.2 * 4) + (0.3 * 3) + (0.4 * 2) + (0.1 * 1) = 2.6
Output [4] = (0.2 * 5) + (0.3 * 4) + (0.4 * 3) + (0.1 * 2) = 3.6

Compare with above hls-code result in output file

Thus, the output of the FIR filter for the given input signal and filter coefficients is: {0.2, 0.7, 1.6, 2.6, 3.6}.

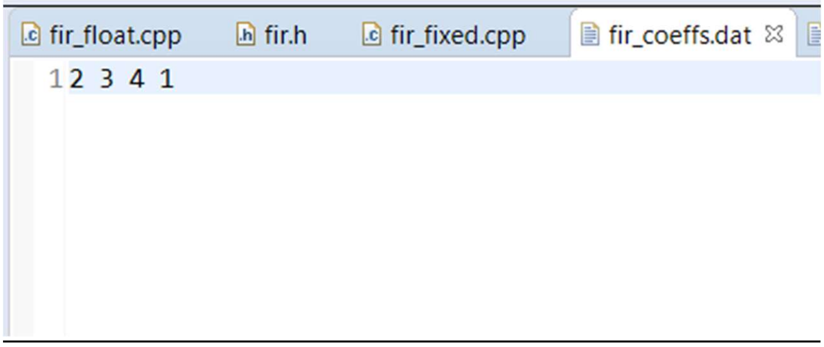
Input Files: (Test2)

Signal sample file:



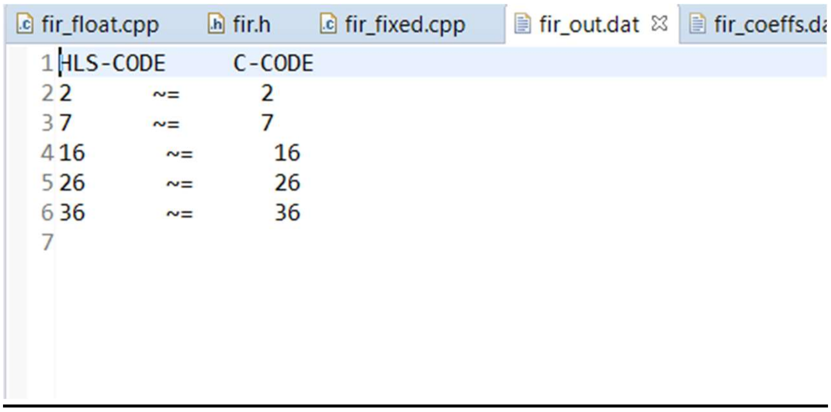
```
1 2 3 4 5
```

Filter coefficients file:



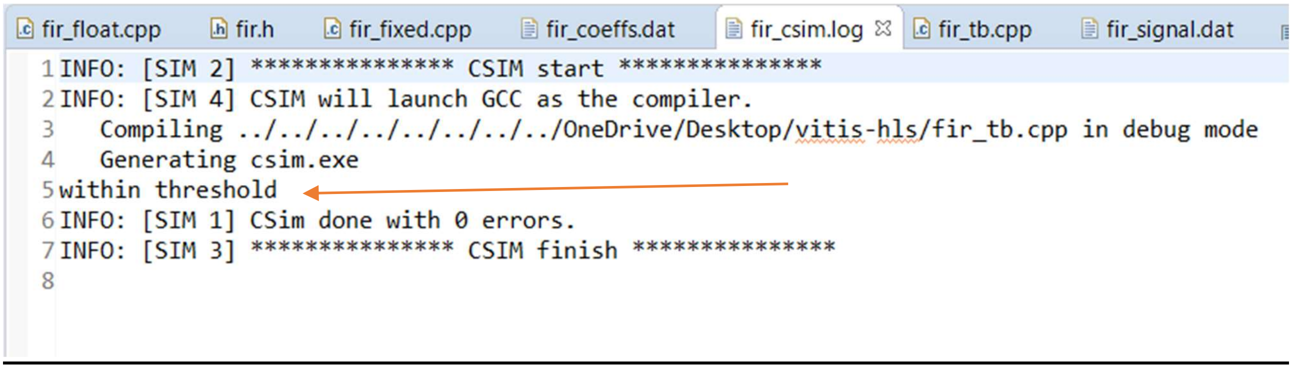
```
2 3 4 1
```

Output File:



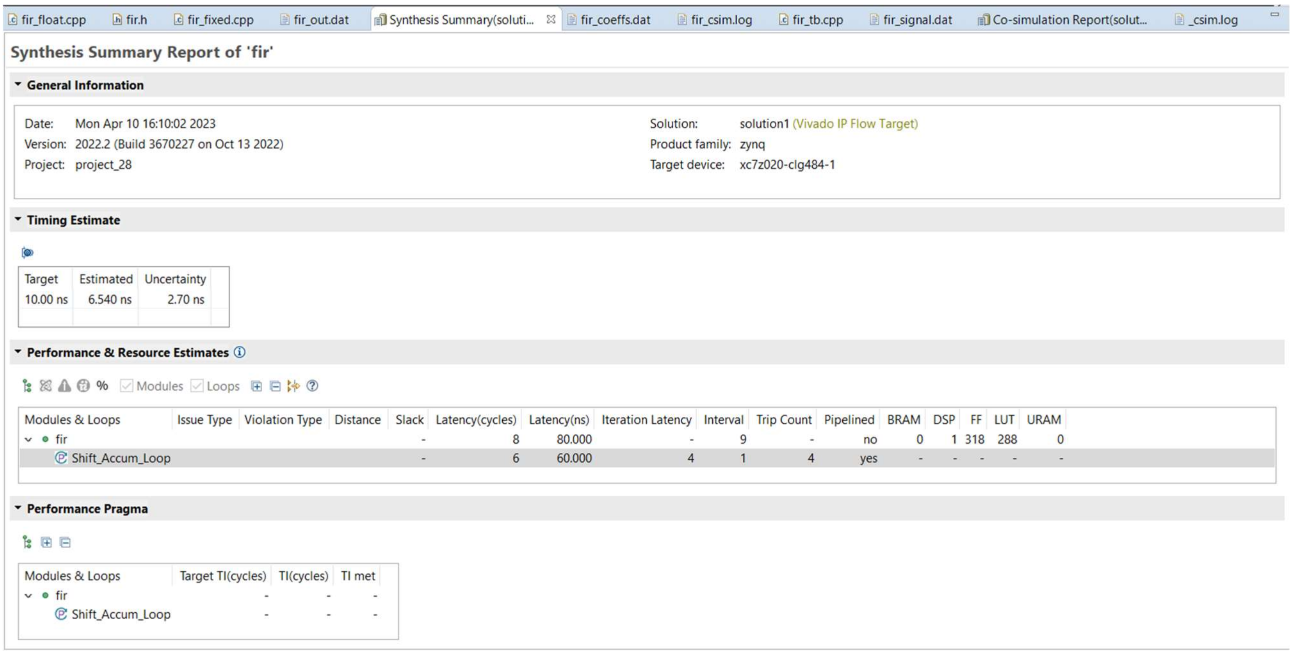
```
HLS-CODE      C-CODE
2       ~=      2
7       ~=      7
16       ~=      16
26       ~=      26
36       ~=      36
```

C simulation printed output:



```
INFO: [SIM 2] *************** CSIM start ***************
INFO: [SIM 4] CSIM will launch GCC as the compiler.
   Compiling ../../../../../../../../OneDrive/Desktop/vitis-hls/fir_tb.cpp in debug mode
   Generating csim.exe
within threshold
INFO: [SIM 1] CSim done with 0 errors.
INFO: [SIM 3] *************** CSIM finish ***************
```

Synthesis Report:



Synthesis Summary Report of 'fir'

General Information

Date: Mon Apr 10 16:10:02 2023
Version: 2022.2 (Build 3670227 on Oct 13 2022)
Project: project_28

Solution: solution1 (Vivado IP Flow Target)
Product family: zynq
Target device: xc7z020-clg484-1

Timing Estimate

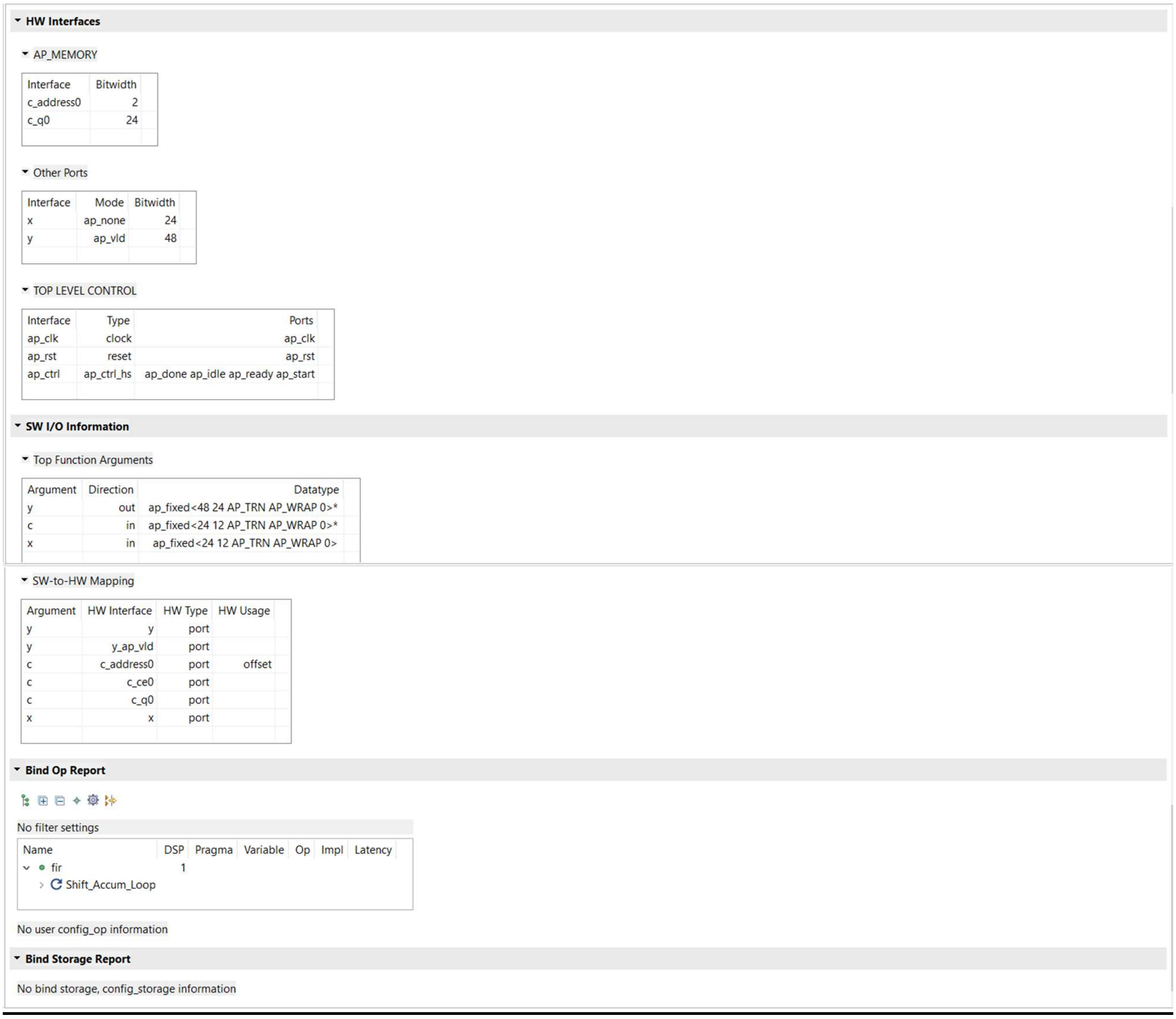
Target	Estimated	Uncertainty
10.00 ns	6.540 ns	2.70 ns

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
fir				-	8	80.000	-	9	-	no	0	1	318	288	0
Shift_Accum_Loop				-	6	60.000	4	1	4	yes	-	-	-	-	-

Performance Pragma

Modules & Loops	Target TI(cycles)	TI(cycles)	TI met
fir	-	-	-
Shift_Accum_Loop	-	-	-



HW Interfaces

AP_MEMORY

Interface	Bitwidth
c_address0	2
c_q0	24

Other Ports

Interface	Mode	Bitwidth
x	ap_none	24
y	ap_vld	48

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
y	out	ap_fixed<48 24 AP_TRN AP_WRAP 0>*
c	in	ap_fixed<24 12 AP_TRN AP_WRAP 0>*
x	in	ap_fixed<24 12 AP_TRN AP_WRAP 0>

SW-to-HW Mapping

Argument	HW Interface	HW Type	HW Usage
y	y	port	
y	y_ap_vld	port	
c	c_address0	port	offset
c	c_ce0	port	
c	c_q0	port	
x	x	port	

Bind Op Report

No filter settings

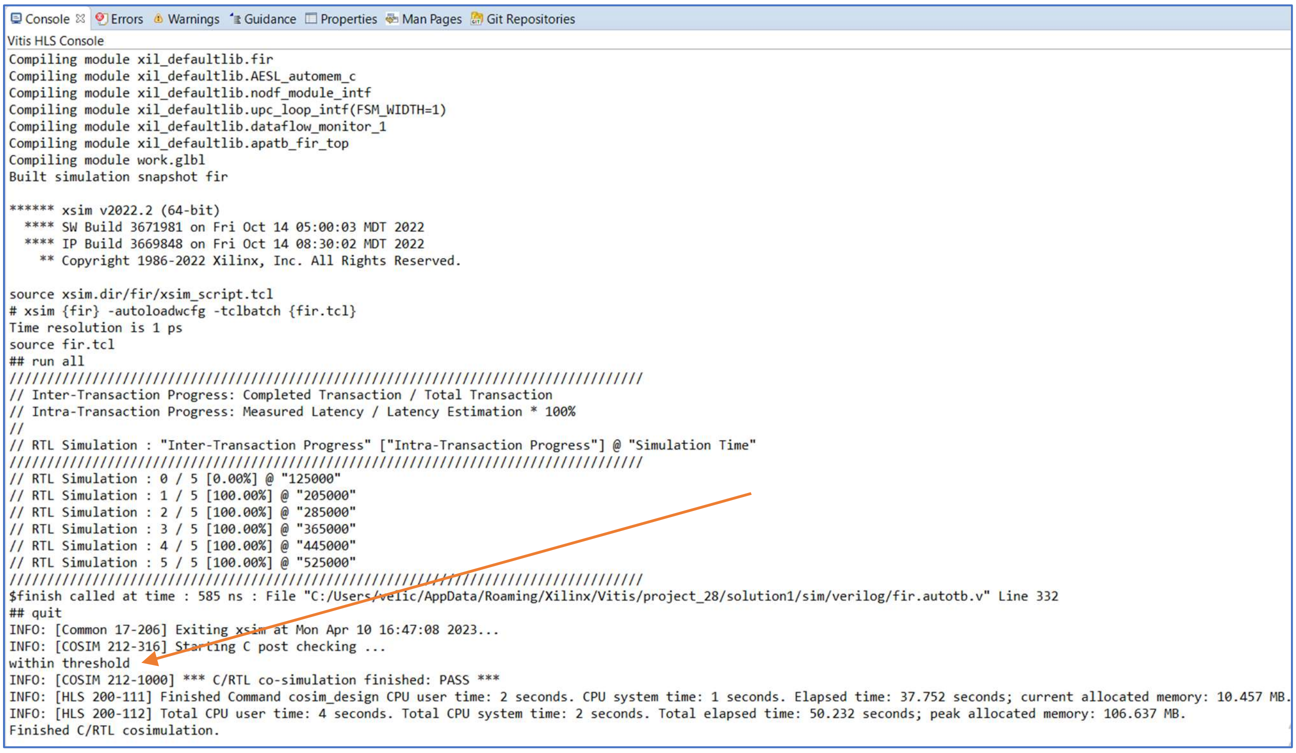
Name	DSP	Pragma	Variable	Op	Impl	Latency
fir	1					
Shift_Accum_Loop						

No user config_op information

Bind Storage Report

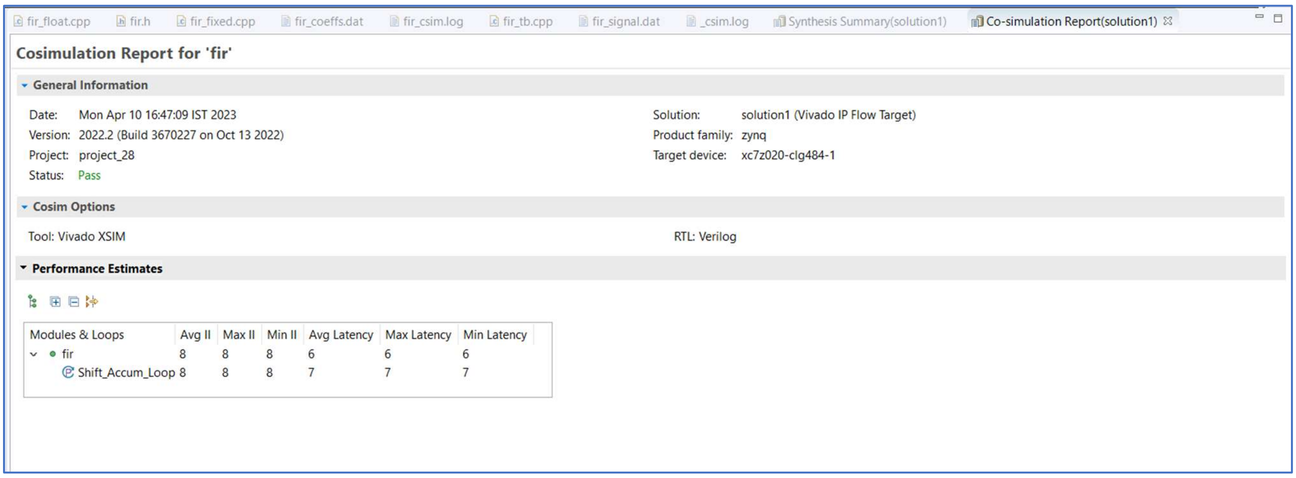
No bind storage, config_storage information

Cosimulation printed output:



```
Compiling module xil_defaultlib.fir
Compiling module xil_defaultlib.AESL_automem_c
Compiling module xil_defaultlib.nodf_module_intf
Compiling module xil_defaultlib.upc_loop_intf(FSM_WIDTH=1)
Compiling module xil_defaultlib.dataflow_monitor_1
Compiling module xil_defaultlib.apatb_fir_top
Compiling module work.glbl
Built simulation snapshot fir

****** xsim v2022.2 (64-bit)
  **** SW Build 3671981 on Fri Oct 14 05:00:03 MDT 2022
  **** IP Build 3669848 on Fri Oct 14 08:30:02 MDT 2022
    ** Copyright 1986-2022 Xilinx, Inc. All Rights Reserved.

source xsim.dir/fir/xsim_script.tcl
# xsim {fir} -autoloadwcfg -tclbatch {fir.tcl}
Time resolution is 1 ps
source fir.tcl
## run all
////////////////////////////////////////////////////////////////////////////////////
// Inter-Transaction Progress: Completed Transaction / Total Transaction
// Intra-Transaction Progress: Measured Latency / Latency Estimation * 100%
//
// RTL Simulation : "Inter-Transaction Progress" ["Intra-Transaction Progress"] @ "Simulation Time"
////////////////////////////////////////////////////////////////////////////////////
// RTL Simulation : 0 / 5 [0.00%] @ "125000"
// RTL Simulation : 1 / 5 [100.00%] @ "205000"
// RTL Simulation : 2 / 5 [100.00%] @ "285000"
// RTL Simulation : 3 / 5 [100.00%] @ "365000"
// RTL Simulation : 4 / 5 [100.00%] @ "445000"
// RTL Simulation : 5 / 5 [100.00%] @ "525000"
////////////////////////////////////////////////////////////////////////////////////
$finish called at time : 585 ns : File "C:/Users/velic/AppData/Roaming/Xilinx/Vitis/project_28/solution1/sim/verilog/fir.autotb.v" Line 332
## quit
INFO: [Common 17-206] Exiting xsim at Mon Apr 10 16:47:08 2023...
INFO: [COSIM 212-316] Starting C post checking ...
within threshold
INFO: [COSIM 212-1000] *** C/RTL co-simulation finished: PASS ***
INFO: [HLS 200-111] Finished Command cosim_design CPU user time: 2 seconds. CPU system time: 1 seconds. Elapsed time: 37.752 seconds; current allocated memory: 10.457 MB.
INFO: [HLS 200-112] Total CPU user time: 4 seconds. Total CPU system time: 2 seconds. Total elapsed time: 50.232 seconds; peak allocated memory: 106.637 MB.
Finished C/RTL cosimulation.
```

Cosimulation Report:



Cosimulation Report for 'fir'

General Information

Date: Mon Apr 10 16:47:09 IST 2023
Version: 2022.2 (Build 3670227 on Oct 13 2022)
Project: project_28
Status: Pass

Solution: solution1 (Vivado IP Flow Target)
Product family: zynq
Target device: xc7z020-clg484-1

Cosim Options

Tool: Vivado XSIM

RTL: Verilog

Performance Estimates

Modules & Loops	Avg II	Max II	Min II	Avg Latency	Max Latency	Min Latency
fir	8	8	8	6	6	6
Shift_Accum_Loop	8	8	8	7	7	7

Github:

https://github.com/velicharlagokulkumar/vitis-hls/tree/main/project_28