

| Step-By-Step Lab1: | FIR

Yuan-teng Chang, Catapult HLS AE
yuan-teng.chang@siemens.com

Objectives

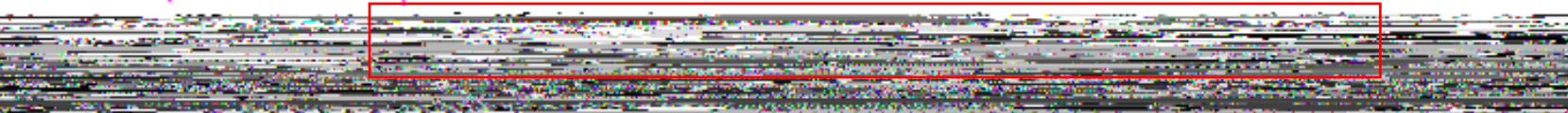
- Learn the Catapult HLS C++ flow
- Understand the loop unrolling and pipelining
- Understand the memory interface
- Learn how to design with multiple blocks

Environment Variables

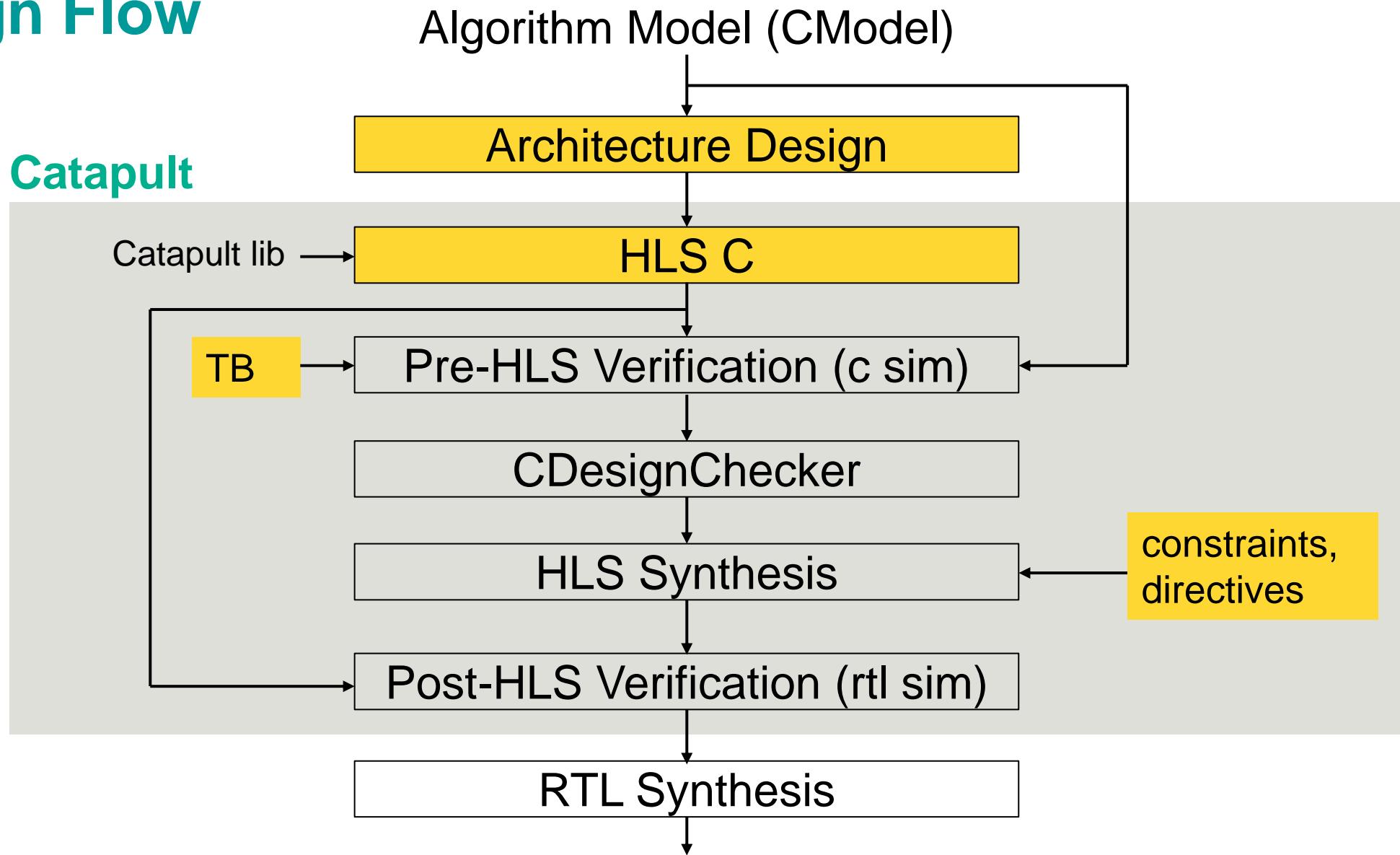
To run Catapult HLS and QuestaSim, a few environment variables are required to be defined

- Define them in a shell script like setup-catapult.csh and source it
- Modify the path depending on the tool install path in your workstation

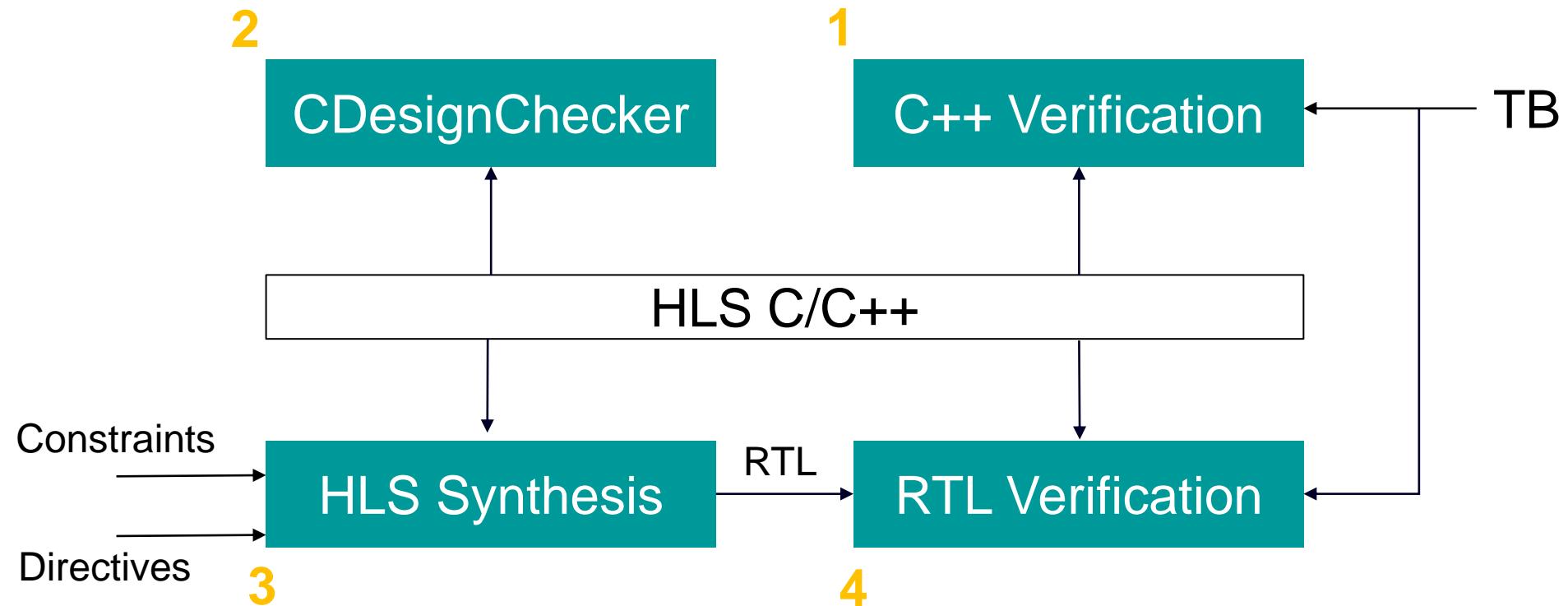
```
2 echo "Catapult setup"
3 setenv MGC_HOME /usr/local/bin/Siemens_EDA/Catapult_Synthesis_2022.2_1-1019737/Mgc_home
4
5 setenv CXX_HOME $MGC_HOME
6 setenv SYSTEMC_INCDIR $MGC_HOME/shared/include
7 setenv SYSTEMC_LIBDIR $MGC_HOME/shared/lib
8 set path = ($path ${MGC_HOME}/bin)
9
10 setenv LD_LIBRARY_PATH $SYSTEMC_LIBDIR
11 setenv LD_LIBRARY_PATH ${MGC_HOME}/lib:$LD_LIBRARY_PATH
12
13 echo "QuestaSim setup"
```



Design Flow



Catapult Flow



| Walkthrough of Catapult C++ Flow

An 8-tap FIR

fir_ref.h (Reference model)

$$\begin{aligned}y &= \sum_{i=0}^7 c_i \cdot r[i] \\&= c_0 r[0] + c_1 r[1] + \dots + c_7 r[7]\end{aligned}$$

```
class fir_ref {
private:
int regs[8];

public:
fir_ref() {
    for (int i = 7; i >= 0; i--)
        { regs[i] = 0; }
}
...
```

```
void run(int input, int coeffs[8], int &output)
{
    int temp = 0;

    for (int i = 7; i >= 0; i--) {
        if (i == 0) {
            regs[i] = input;
        } else {
            regs[i] = regs[i - 1];
        }
    }

    for (int i = 0; i < 8; i++)
        temp += regs[i] * coeffs[i];

    output = temp >> 11;
};
```

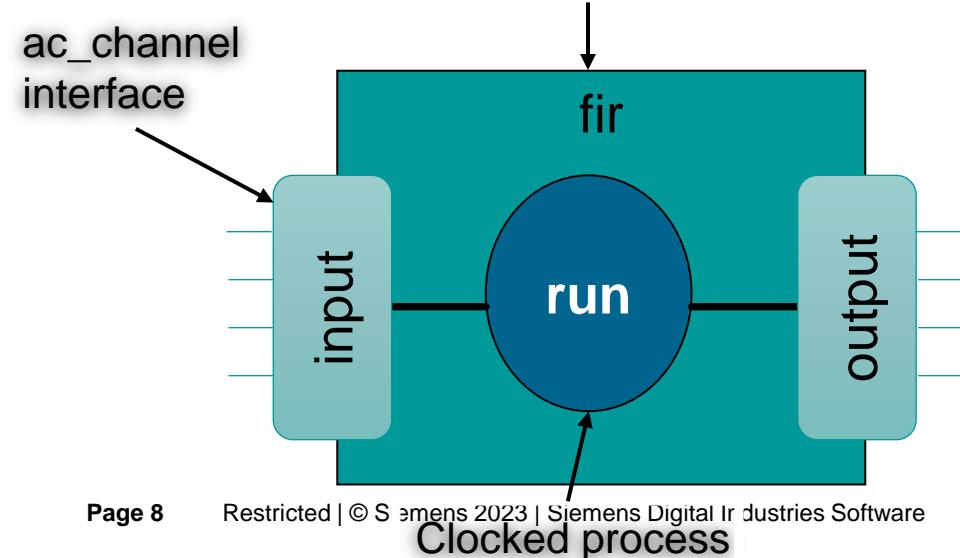
fir.h (HLS C)

```
#include <ac_int.h>
#include <ac_channel.h>
#include <mc_scverify.h>

class fir {
private:
    int8 regs[8];
public:
    fir() {
        for (int i = 7; i >= 0; i--)
            { regs[i] = 0; }
    }
}
```

Required

Design block



```
#pragma hls_design_interface
void CCS_BLOCK(run)(ac_channel<int8> &input,
                     int8 coeffs[8],
                     ac_channel<int8> &output)
{
    int19 temp = 0;
    SHIFT: for (int i = 7; i >= 0; i--) {
        if (i == 0) {
            regs[i] = input.read();
        } else {
            regs[i] = regs[i - 1];
        }
    }
    MAC: for (int i = 7; i >= 0; i--) {
        temp += coeffs[i] * regs[i];
    }
    output.write(temp >> 11);
};
```

Required

label

fir_tb.cpp (TB)

```
#include "fir_ref.h"
#include "fir.h"
CCS_MAIN(int argc, char *argv[])
{
    int8 coeffs[8];
    ac_channel<int8> in_chn;
    ac_channel<int8> out_chn;
    int8 din, dout;
    int coeffs_ref[8];
    int din_ref;
    int dout_ref;
    fir dut;
    fir_ref ref_model;
    int pass_cnt = 0;
    int fail_cnt = 0;
    // Test Impulse
    for (int i=0; i < 8; i++)
        coeffs_ref[i]=coeffs[i]=-128+rand()%256;
    ...
}
```

```
for (int i = 0; i < 100; i++) {
    din_ref = din = -128 + rand()%256;
    in_chn.write(din);
    dut.run(in_chn, coeffs, out_chn);
    ref_model.run(din_ref, coeffs_ref,
dout_ref);
    dout = out_chn.read();
    if (dout.to_int() != dout_ref) {
        printf("fail @ %3d: %4d != %4d \n", i,
dout.to_int(), dout_ref);
        fail_cnt++;
    } else {
        printf("pass @ %3d: %4d == %4d \n", i,
dout.to_int(), dout_ref);
        pass_cnt++;
    }
}
printf("\n");
printf("total pass count %d\n", pass_cnt);
printf("total fail count %d\n", fail_cnt);
CCS_RETURN(0);
}
```

Pre-HLS Verification by gcc (C Sim)

Build with gcc

- g++ -o3 -std=c++11 -I \$MGC_HOME/shared/include fir_tb.cpp -o SCVerify_fir.exe

```
[mentor@RHEL74 01_walkthrough_loops]$ make
/usr/local/bin/Siemens_EDA/Catapult_Synthesis_2023.2_1-1065141/Mgc_home/bin/g++ -g -O3 -std=c++11 -Wall -Wno-unknown-pragmas -Wno-unused-variable -Wno-unused-label -I./ -I/usr/local/bin/Siemens_EDA/Catapult_Synthesis_2023.2_1-1065141/Mgc_home/shared/include ./fir_tb.cpp -o bin/SCVerify_fir.exe
[mentor@RHEL74 01_walkthrough_loops]$
```

Run execution file

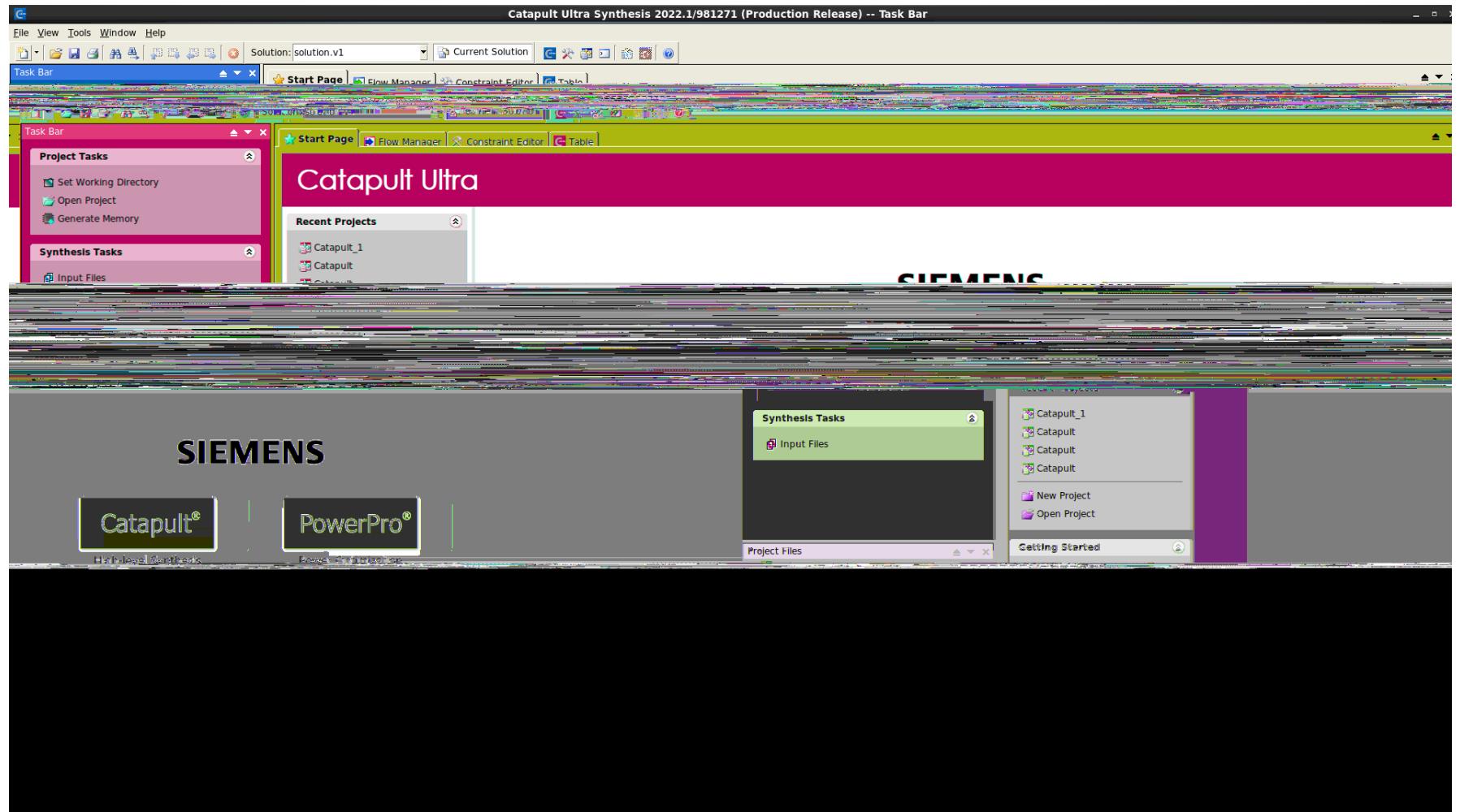
- ./SCVerify_fir.exe

```
pass @ 89:    3 ==    3
pass @ 90:   -18 ==  -18
pass @ 91:    21 ==   21
pass @ 92:   -12 ==  -12
pass @ 93:     2 ==    2
pass @ 94:    -6 ==   -6
pass @ 95:     5 ==    5
pass @ 96:   -21 ==  -21
pass @ 97:    18 ==   18
pass @ 98:    -5 ==   -5
pass @ 99:     9 ==    9

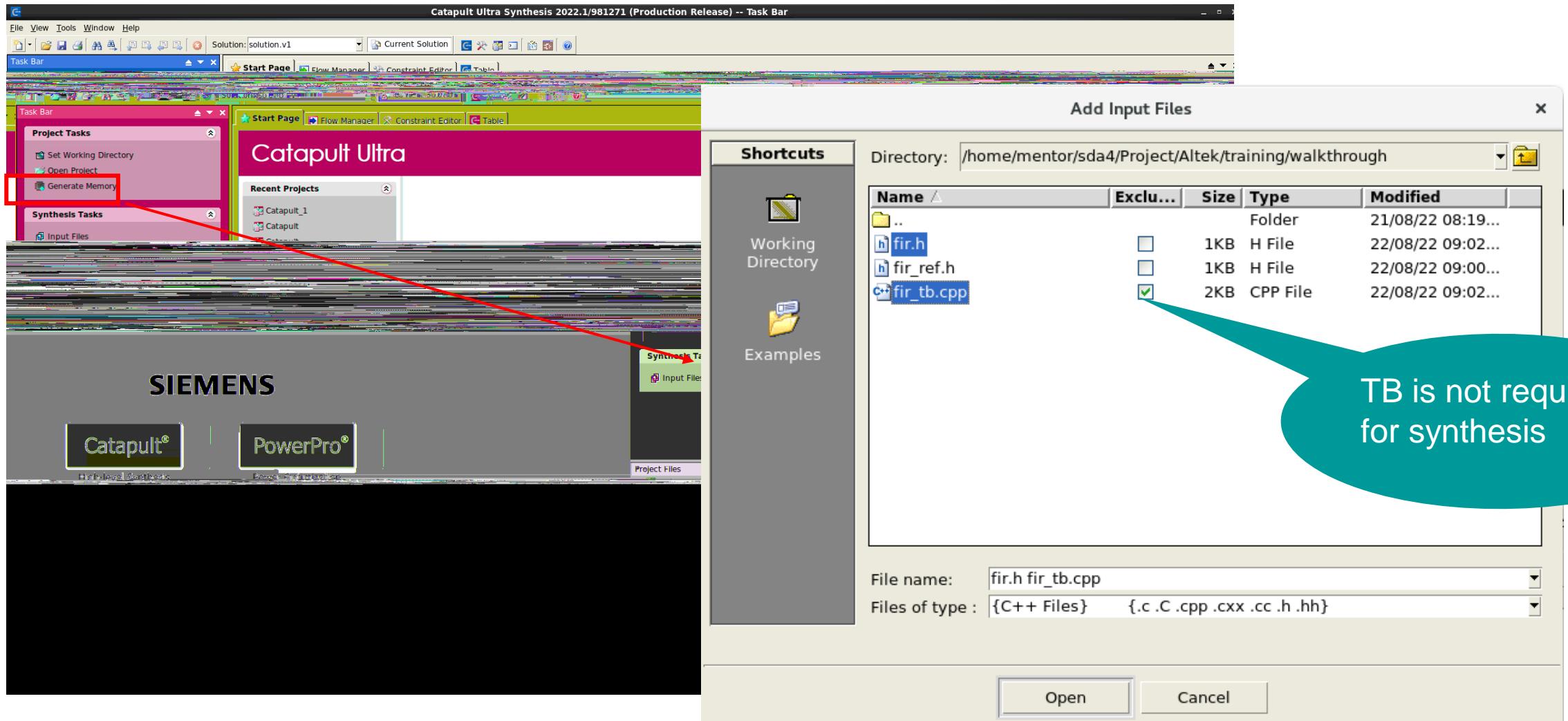
total pass count 100
total fail count 0
```

Launch Catapult

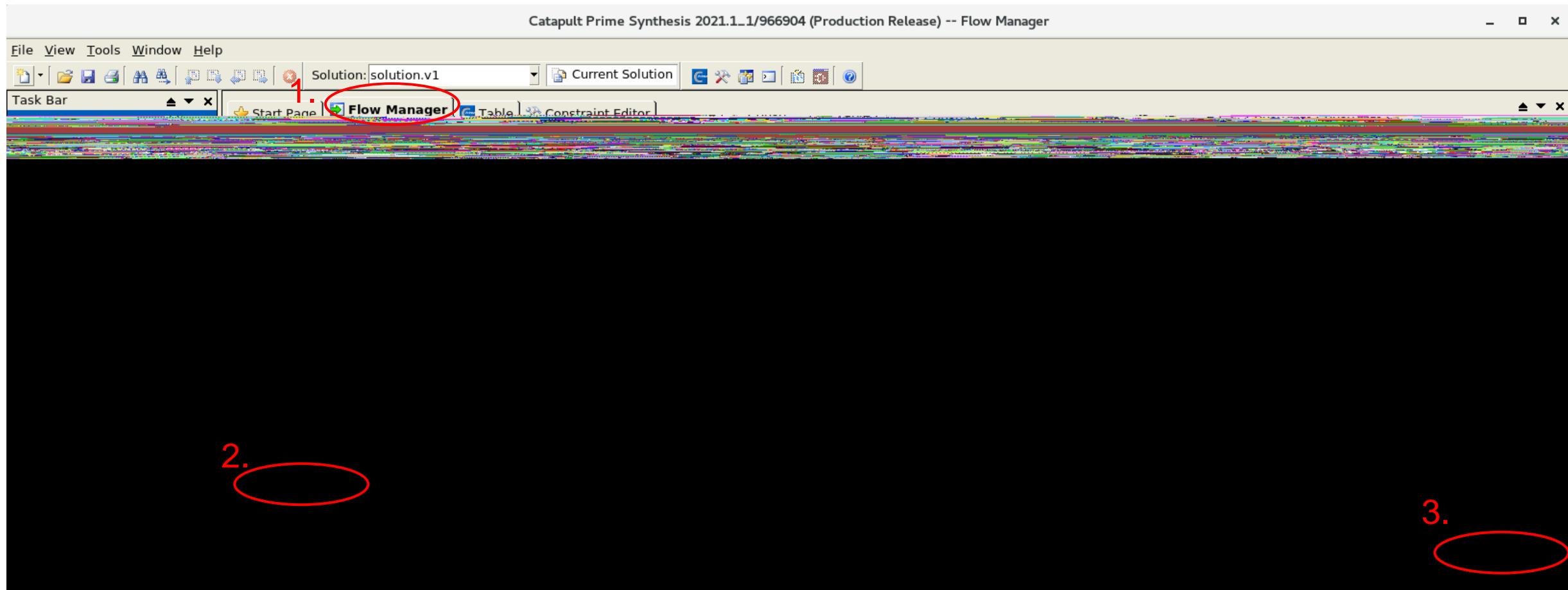
catapult &



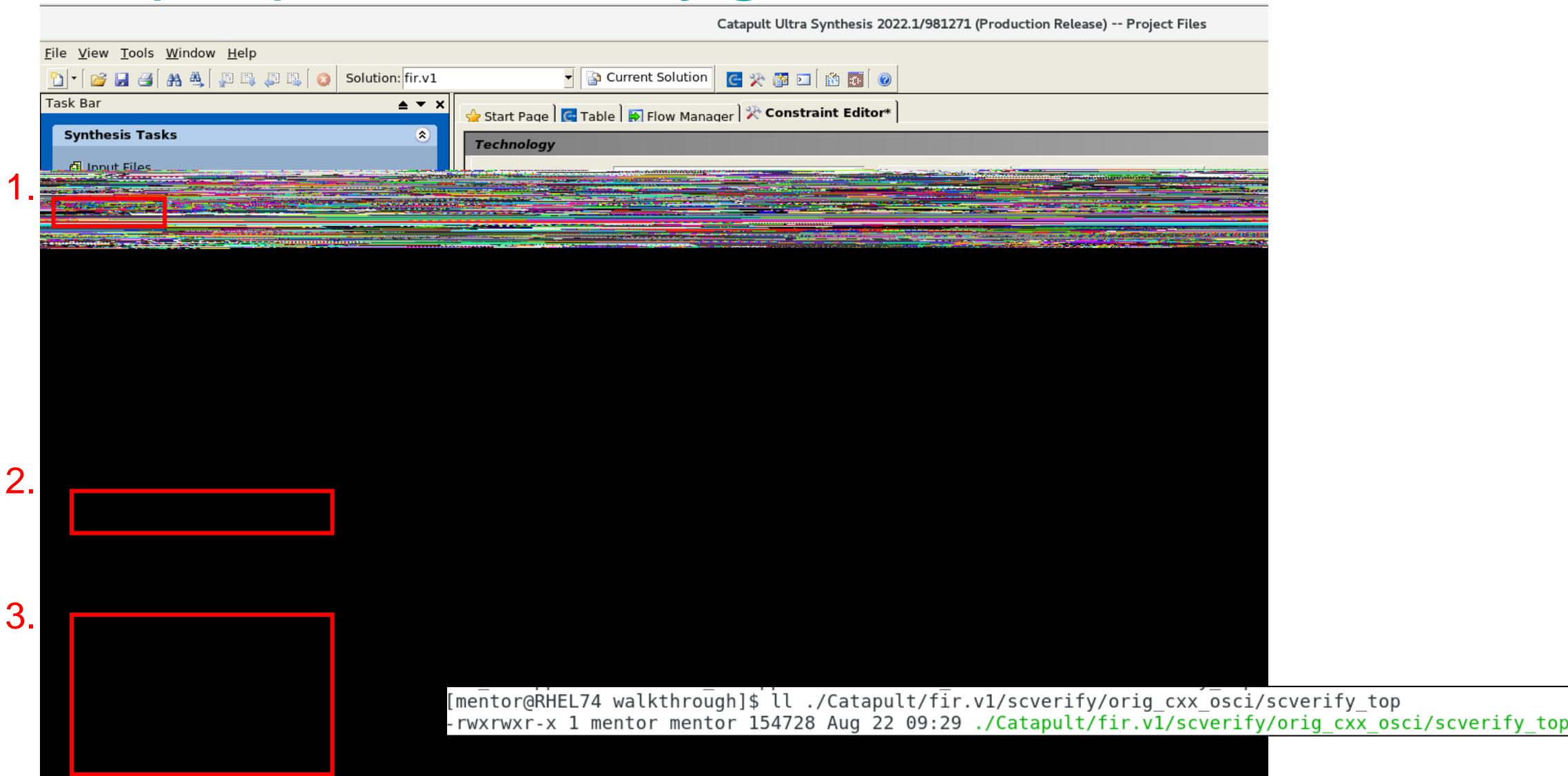
Add input file



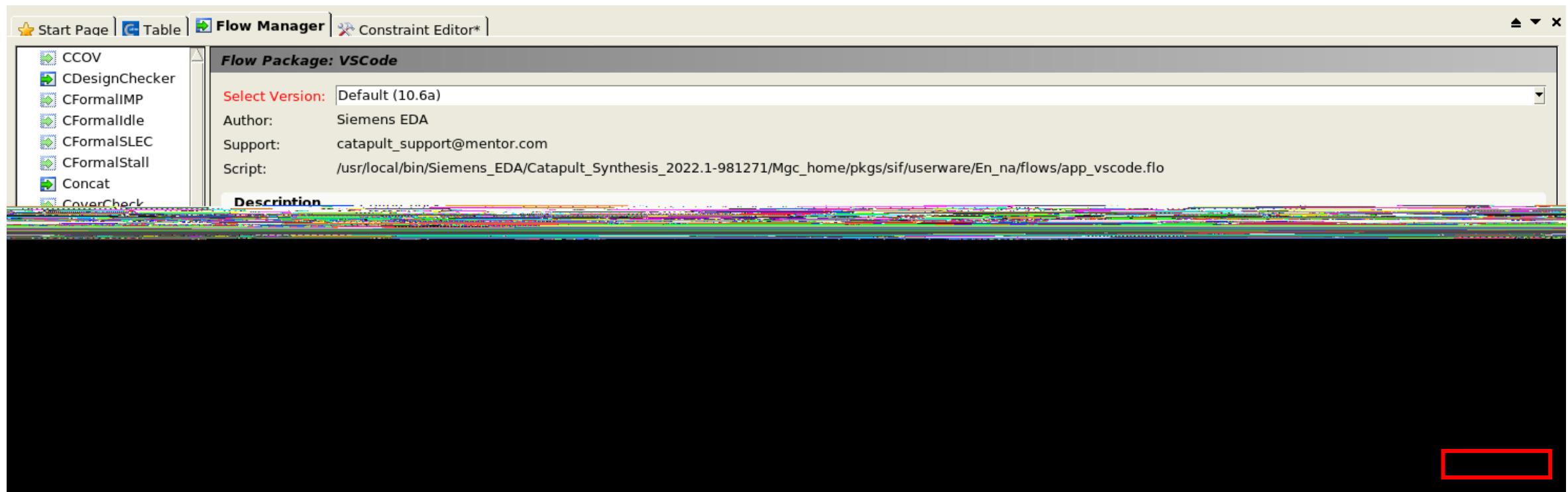
Enable SCVerify



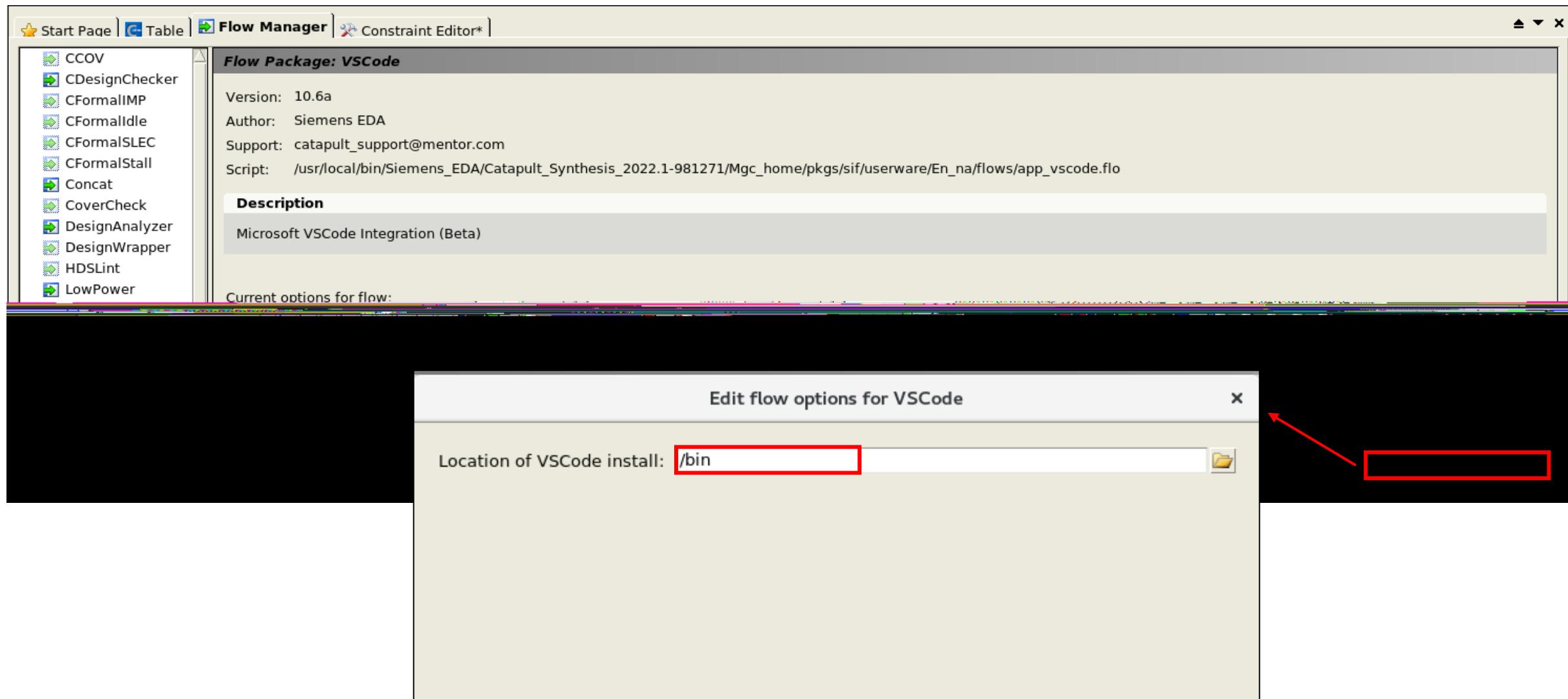
Pre-HLS (C++) Verification by gcc



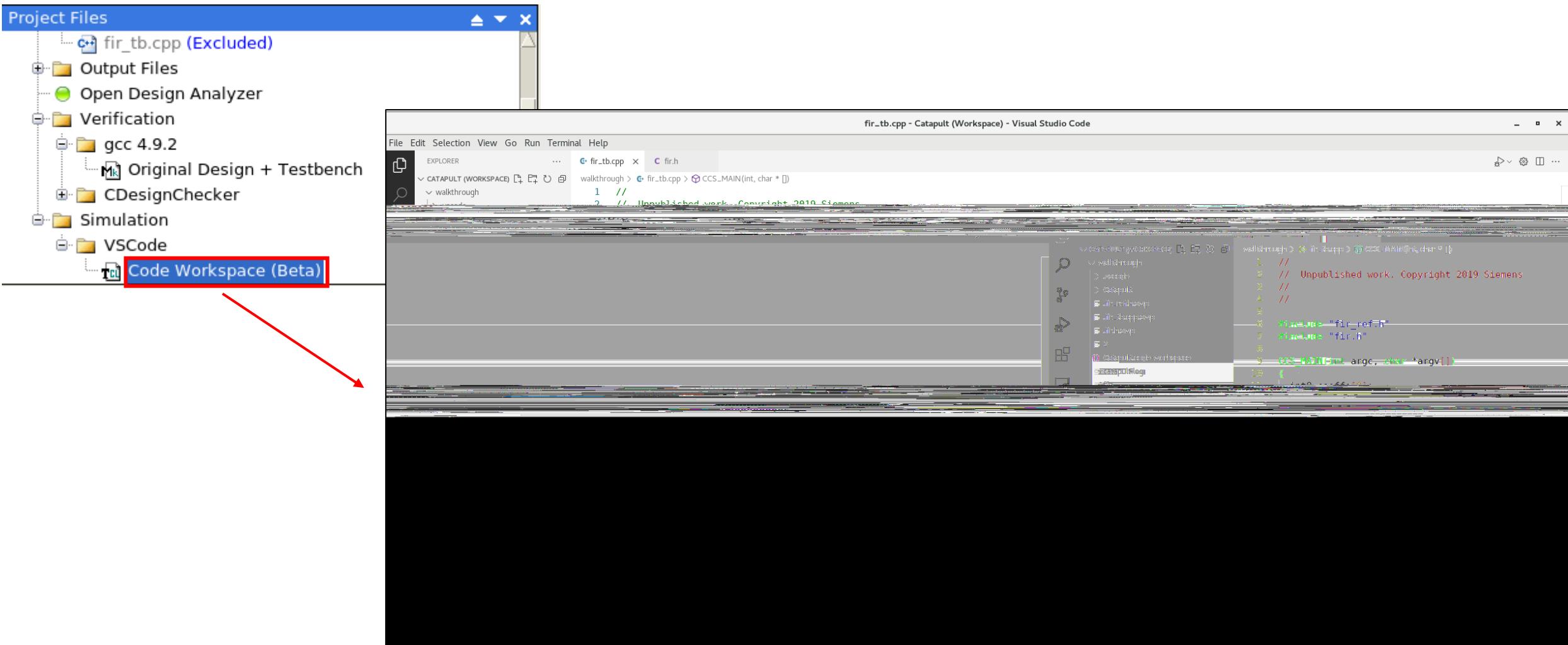
Build VSCode Environment



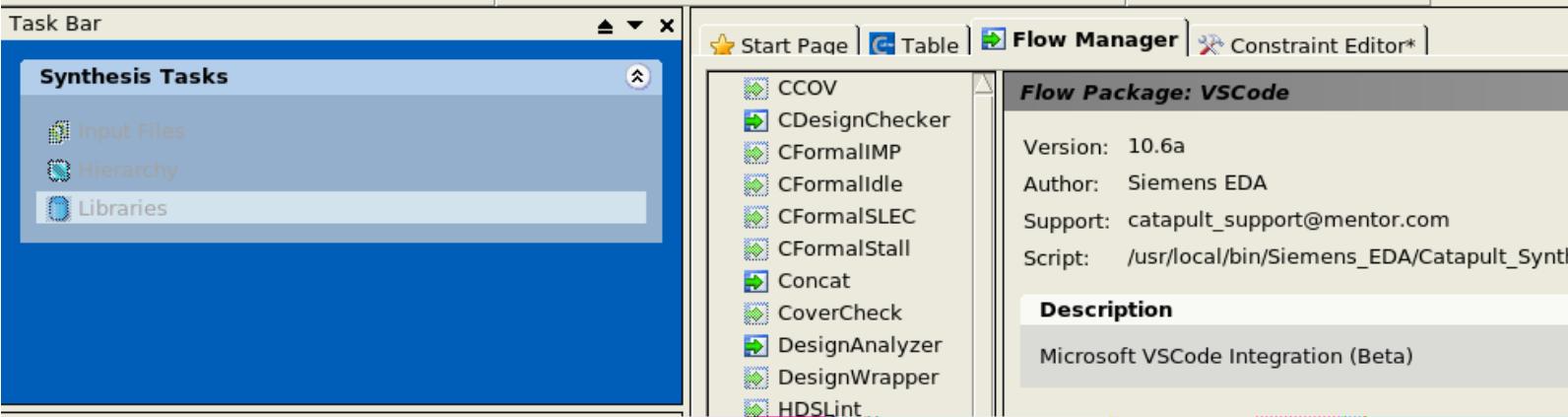
Build VSCode Environment (Cont.)



Build VSCode Environment (Cont.)

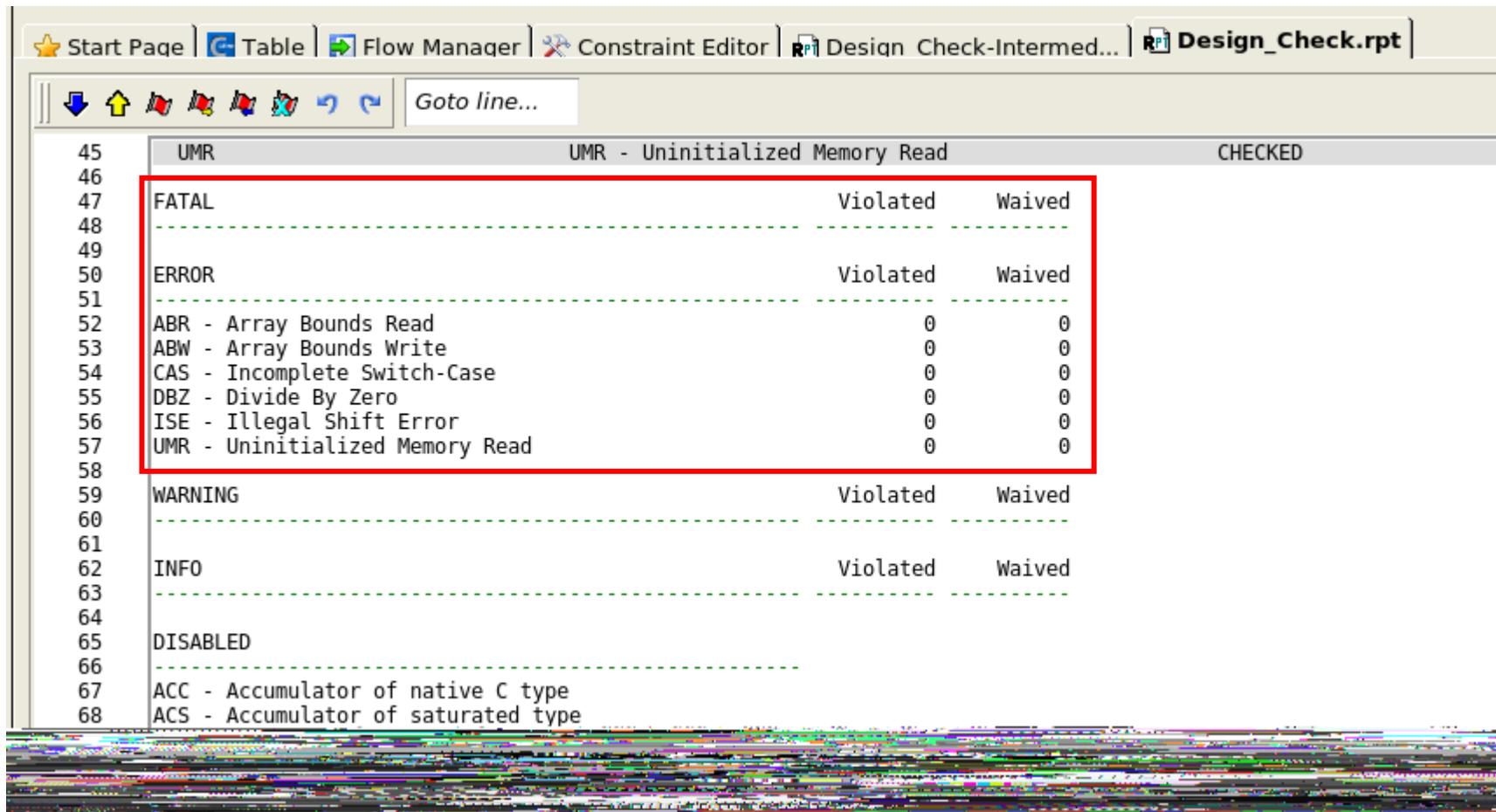


Run CDesignChecker



Select Rules	
<input checked="" type="checkbox"/> ABR - Array Bounds Read	ERROR
<input checked="" type="checkbox"/> ABW - Array Bounds Write	ERROR
<input checked="" type="checkbox"/> ACC - Accumulator of native C type	WARNING
<input checked="" type="checkbox"/> ACS - Accumulator of saturated type	WARNING
<input checked="" type="checkbox"/> AIC - Assignment used Instead of Comparison	WARNING
<input checked="" type="checkbox"/> ALS - Ac_int Left Shift check	WARNING
<input checked="" type="checkbox"/> AOB - Arithmetic Operator with Boolean	ERROR
<input checked="" type="checkbox"/> APT - Array Dimension Power of Two	INFO
<input checked="" type="checkbox"/> AWE - Assignments Without Effect	WARNING
<input checked="" type="checkbox"/> CAS - Incomplete Switch-Case	ERROR
<input checked="" type="checkbox"/> CBU - Conditional break in Unrolled Loop	WARNING
<input checked="" type="checkbox"/> CCC - Static constant comparison	WARNING
<input checked="" type="checkbox"/> CGR - Conditional Guard in Rolled Loop	WARNING
<input checked="" type="checkbox"/> CAS - Incomplete switch Case	ERROR
<input checked="" type="checkbox"/> CBU - Conditional break in Unrolled Loop	WARNING
<input checked="" type="checkbox"/> CCC - Static constant comparison	WARNING
<input checked="" type="checkbox"/> CGR - Conditional Guard in Rolled Loop	WARNING
<input checked="" type="checkbox"/> CIA - Comparison Instead of Assignment	WARNING
<input checked="" type="checkbox"/> CMC - C style Memory Check	INFO
<input checked="" type="checkbox"/> CNS - Constant condition of if/switch	WARNING
<input checked="" type="checkbox"/> CWB - Case Without Break	WARNING
<input checked="" type="checkbox"/> DBZ - Divide By Zero	ERROR
<input checked="" type="checkbox"/> DIU - Dynamic Index in Unrolled Loop	WARNING
<input checked="" type="checkbox"/> FVI - For Loop with Variable Iterations	WARNING
<input checked="" type="checkbox"/> FXD - Mixed fixed and non-fixed datatypes	WARNING

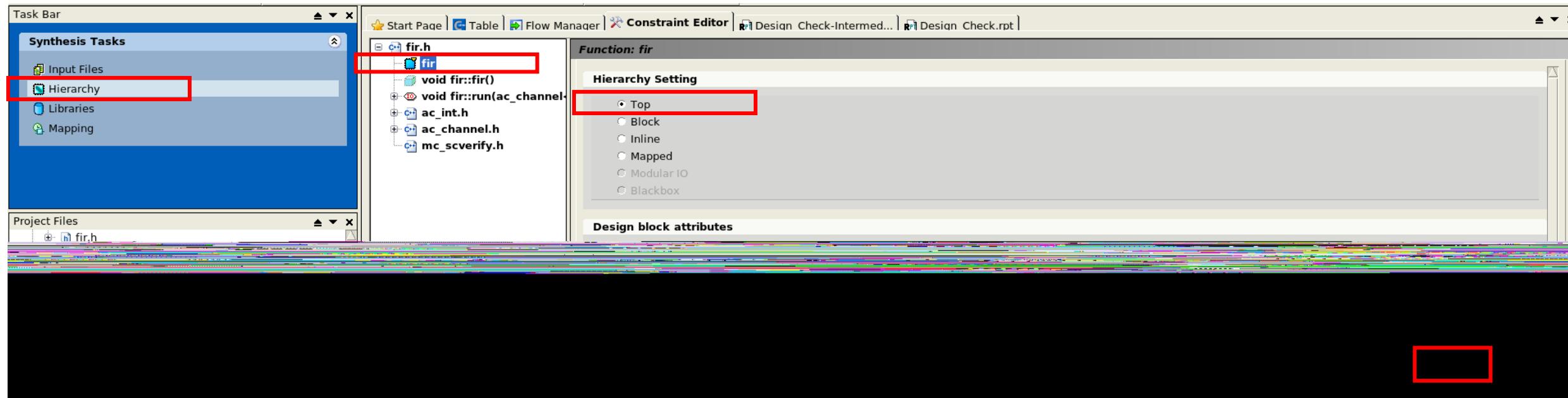
Run CDesignChecker (Cont.)



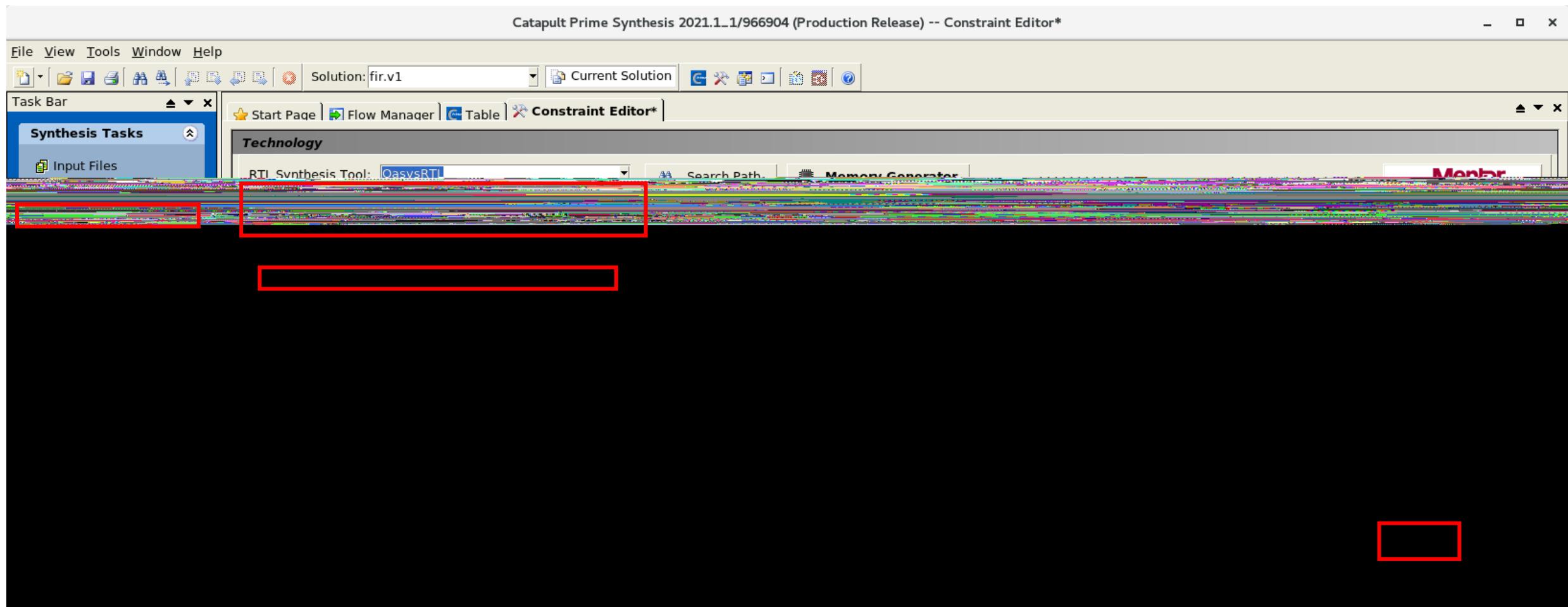
The screenshot shows the SIMATIC Manager interface with the 'Design_Check.rpt' report open. The report lists various error types and their counts:

Category	Violated	Waived
FATAL		
ERROR		
ABR - Array Bounds Read	0	0
ABW - Array Bounds Write	0	0
CAS - Incomplete Switch-Case	0	0
DBZ - Divide By Zero	0	0
ISE - Illegal Shift Error	0	0
UMR - Uninitialized Memory Read	0	0
WARNING		
INFO		
DISABLED		
ACC - Accumulator of native C type		
ACS - Accumulator of saturated type		

HLS Synthesis – Set Top Design

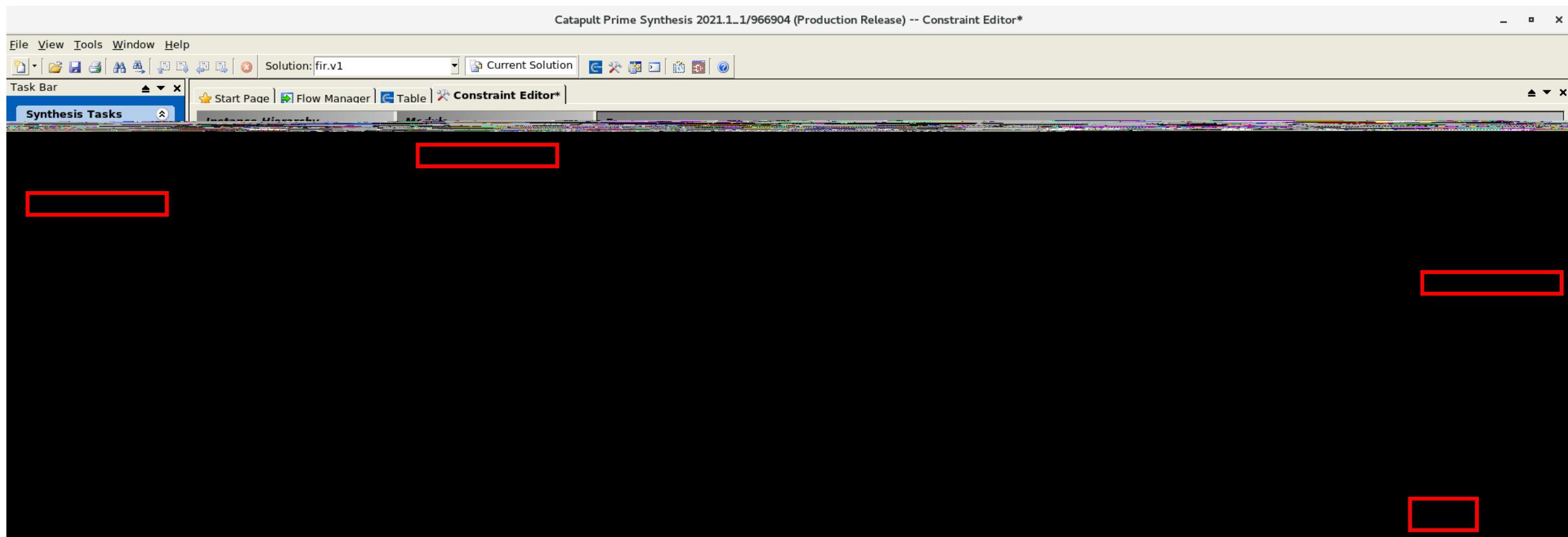


HLS Synthesis – Library Setup



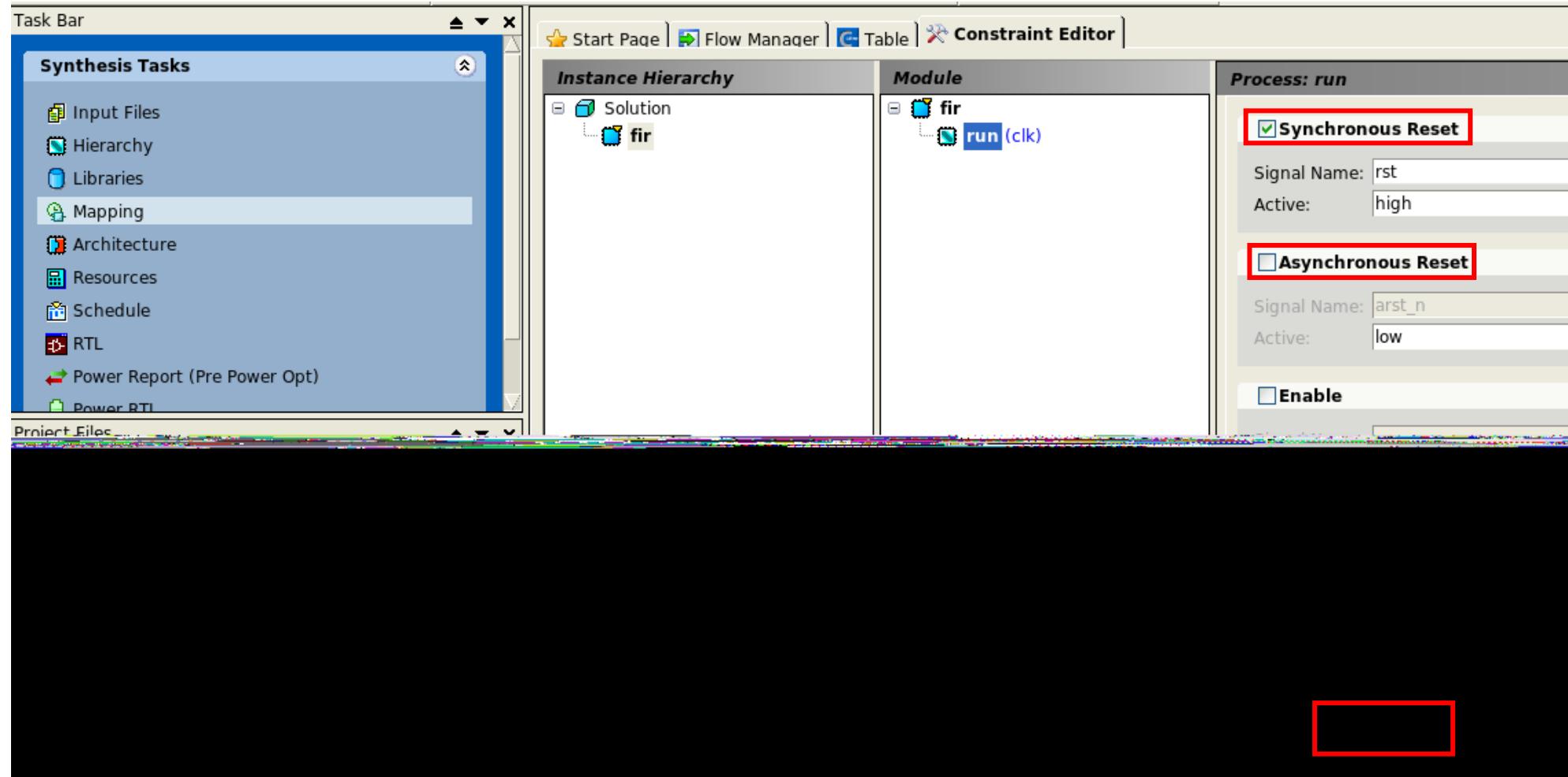
HLS Synthesis – Mapping

Frequency

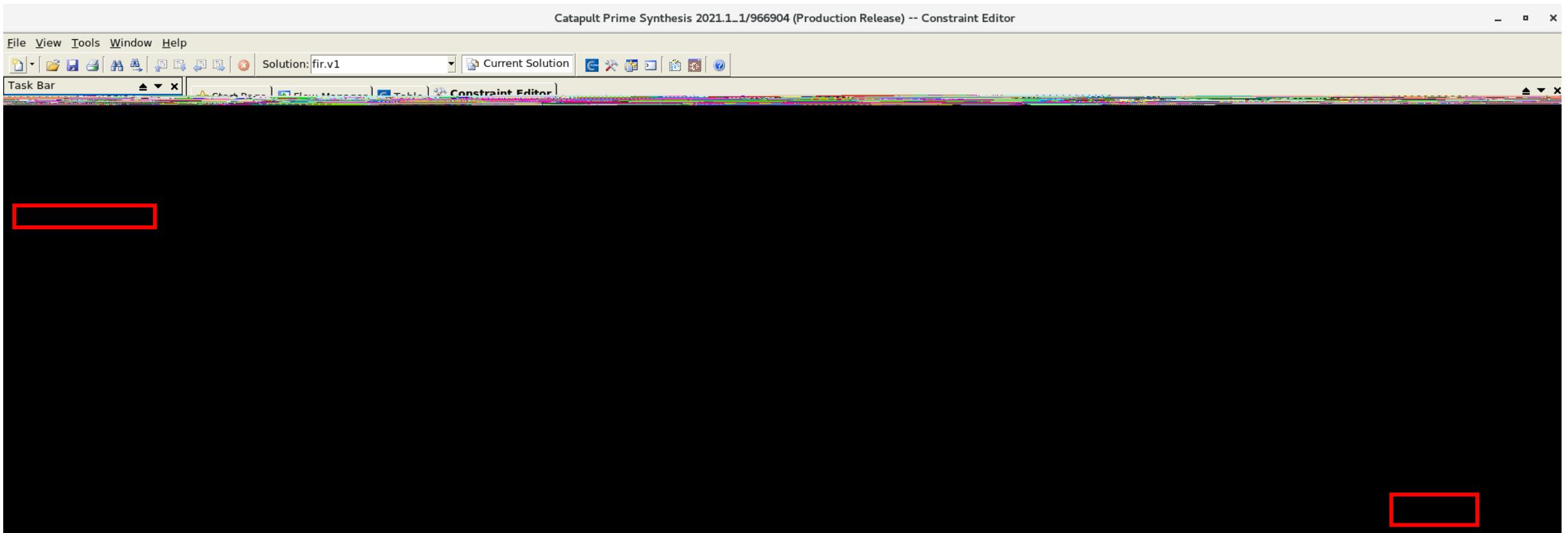


HLS Synthesis – Mapping

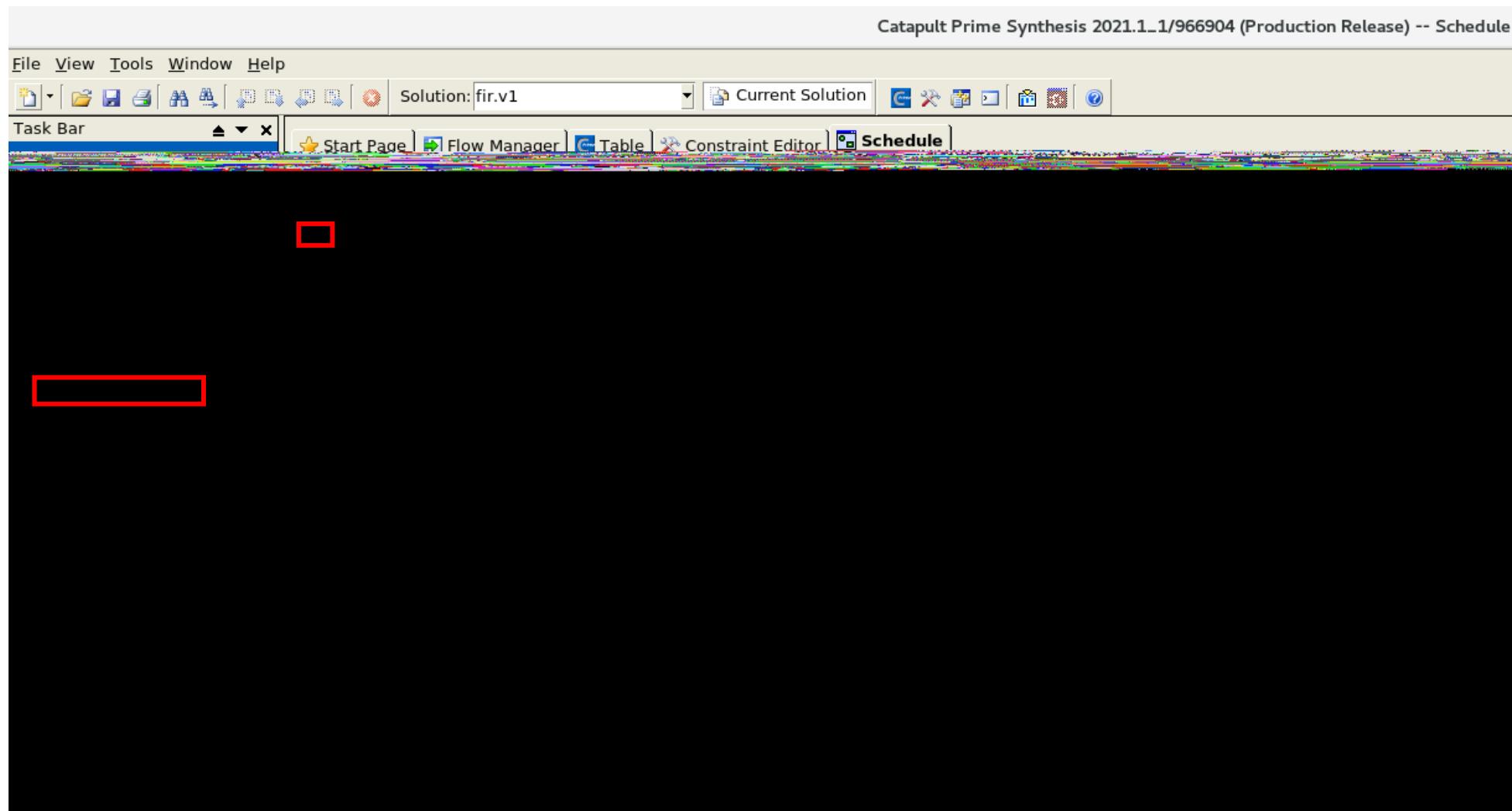
Sync / async reset



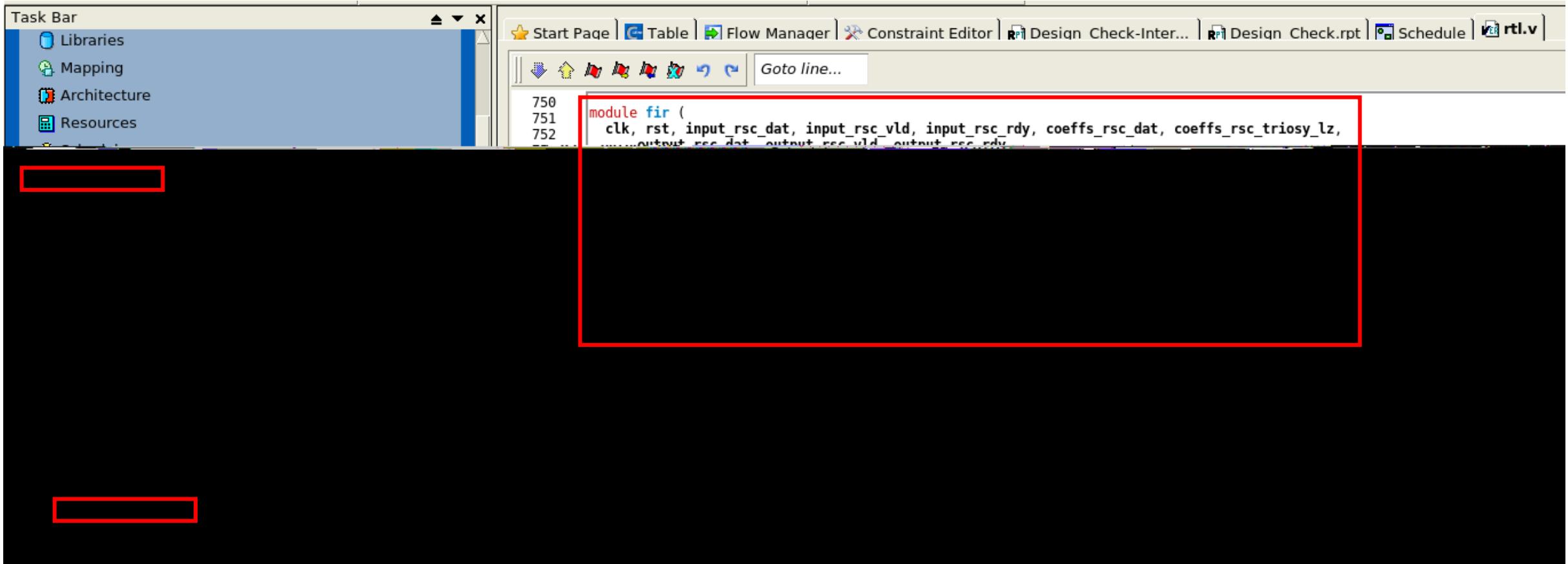
HLS Synthesis – Architecture



HLS Synthesis – Scheduling



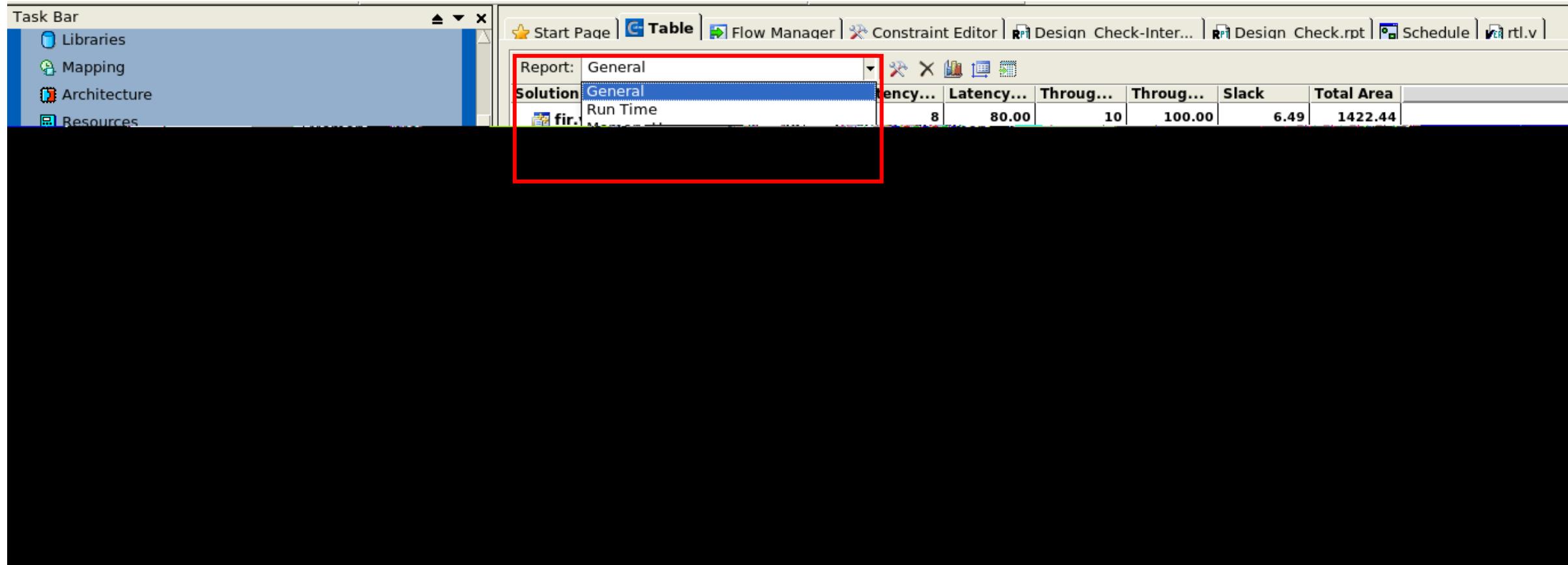
HLS Synthesis – RTL



The screenshot shows the Siemens EDA HLS Synthesis software interface. The top menu bar includes options like Start Page, Table, Flow Manager, Constraint Editor, Design Check-Inter..., Design Check.rpt, Schedule, and rtl.v. The left Task Bar lists Libraries, Mapping, Architecture, and Resources. The main area is a code editor with a red box highlighting the module definition:

```
750
751 module fir (
752   clk, rst, input_rsc_dat, input_rsc_vld, input_rsc_rdy, coeffs_rsc_dat, coeffs_rsc_triosy_lz,
753   .....output_rsc_dat, output_rsc_vld, output_rsc_rdy,
```

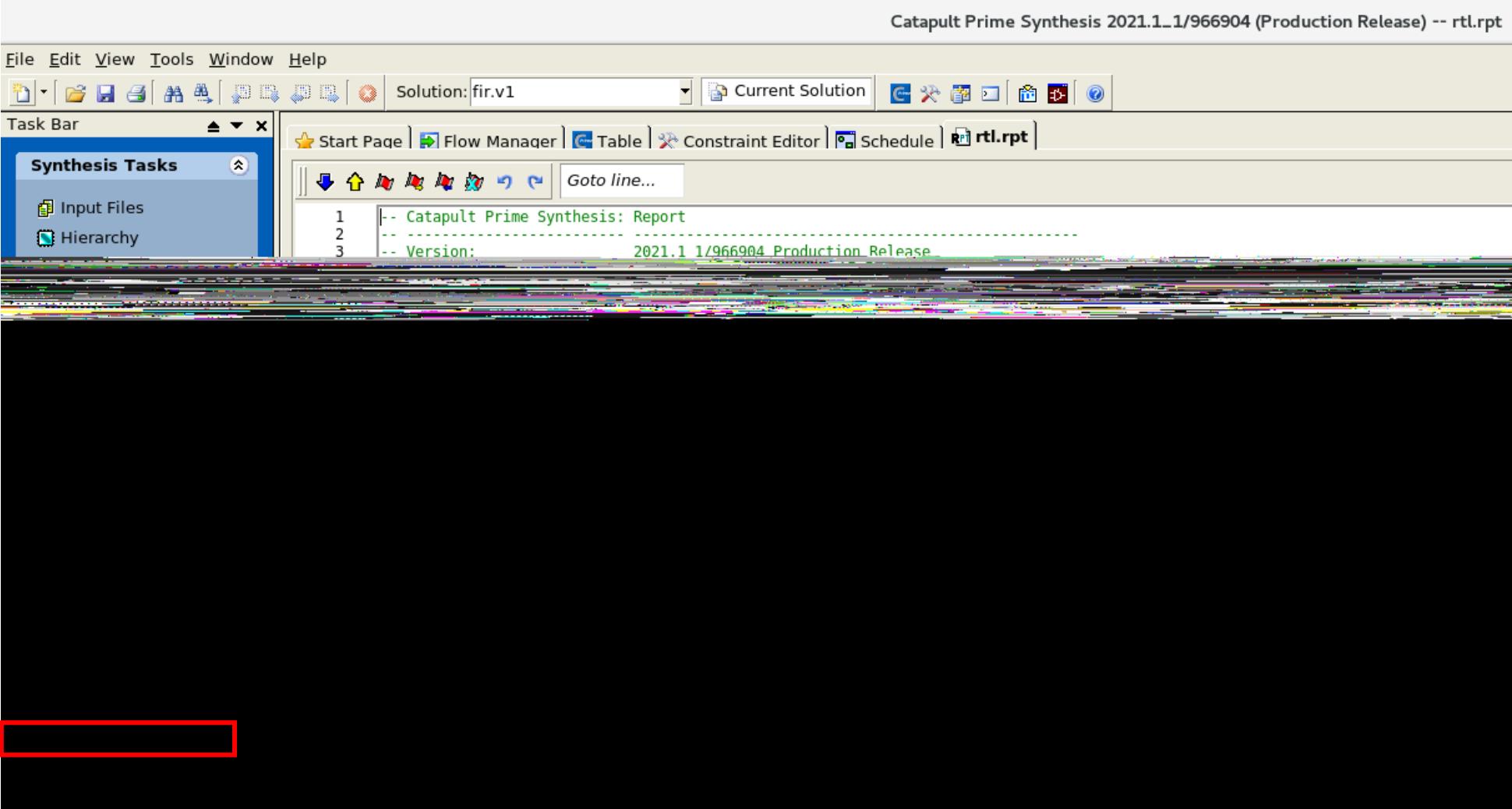
HLS Synthesis – Report



The screenshot shows the HLS Synthesis software interface. The Task Bar on the left includes options for Libraries, Mapping, Architecture, and Resources. The main window features a toolbar with icons for Start Page, Table, Flow Manager, Constraint Editor, Design Check-Inter..., Design Check.rpt, Schedule, and rtl.v. A red box highlights the 'Table' tab, which is currently selected. Below the toolbar is a report table with the following data:

Report:	General	Memory...	Latency...	Through...	Through...	Slack	Total Area
Solution:	General	8	80.00	10	100.00	6.49	1422.44
	Run Time						
	fir.						

HLS Synthesis – Report (Cont.)



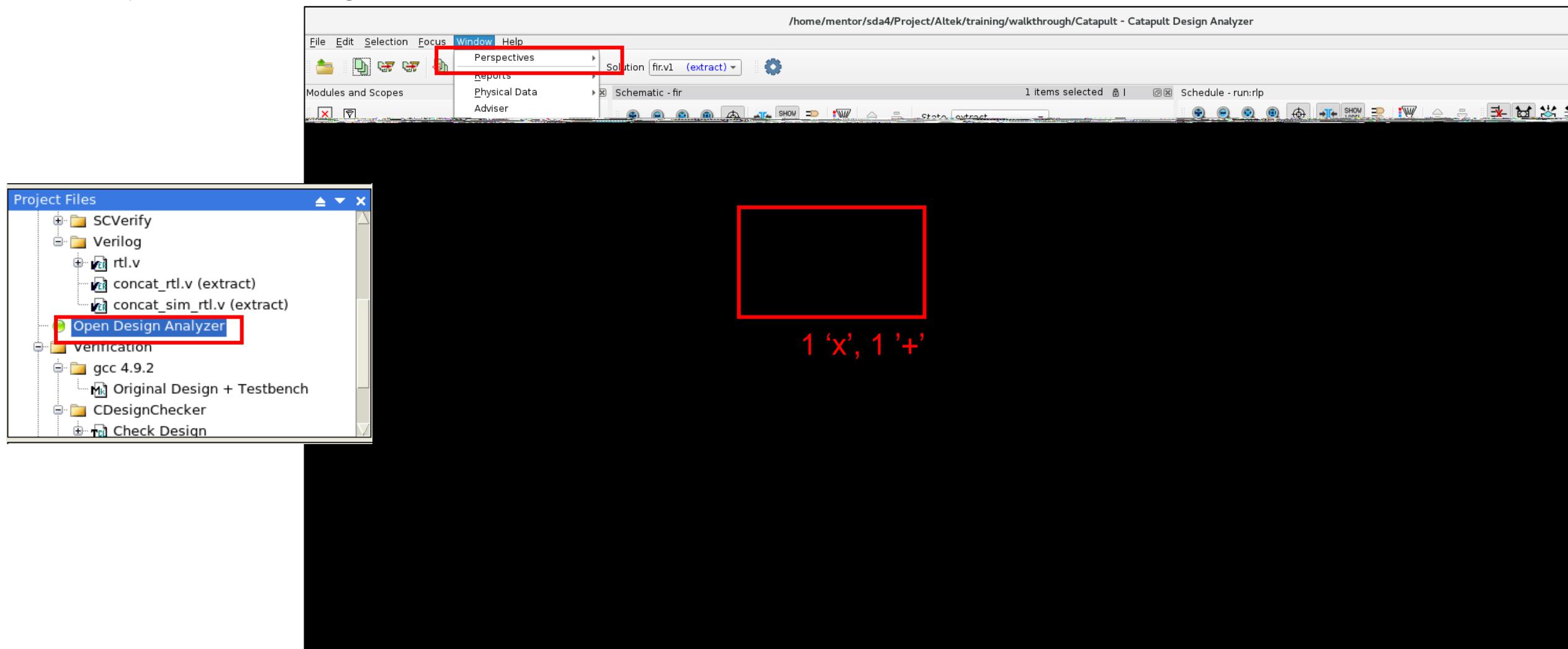
The screenshot shows the Catapult Prime Synthesis 2021.1_1/966904 (Production Release) interface. The window title is "Catapult Prime Synthesis 2021.1_1/966904 (Production Release) -- rtl.rpt". The menu bar includes File, Edit, View, Tools, Window, and Help. The toolbar contains various icons for file operations. The solution name is "fir.v1". The task bar has tabs for Start Page, Flow Manager, Table, Constraint Editor, Schedule, and "rtl.rpt" (which is currently selected). The main area displays a synthesis report with the following content:

```
1 -- Catapult Prime Synthesis: Report
2 --
3 -- Version: 2021.1 1/966904 Production Release
```

A red rectangular box highlights a portion of the blacked-out content at the bottom of the report area.

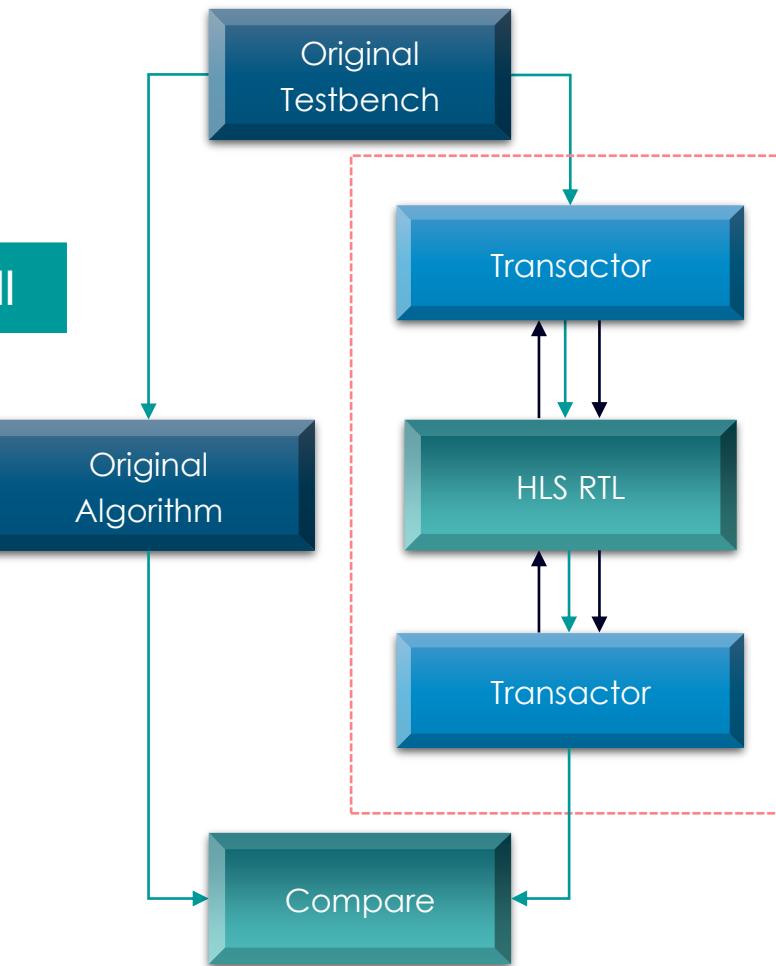
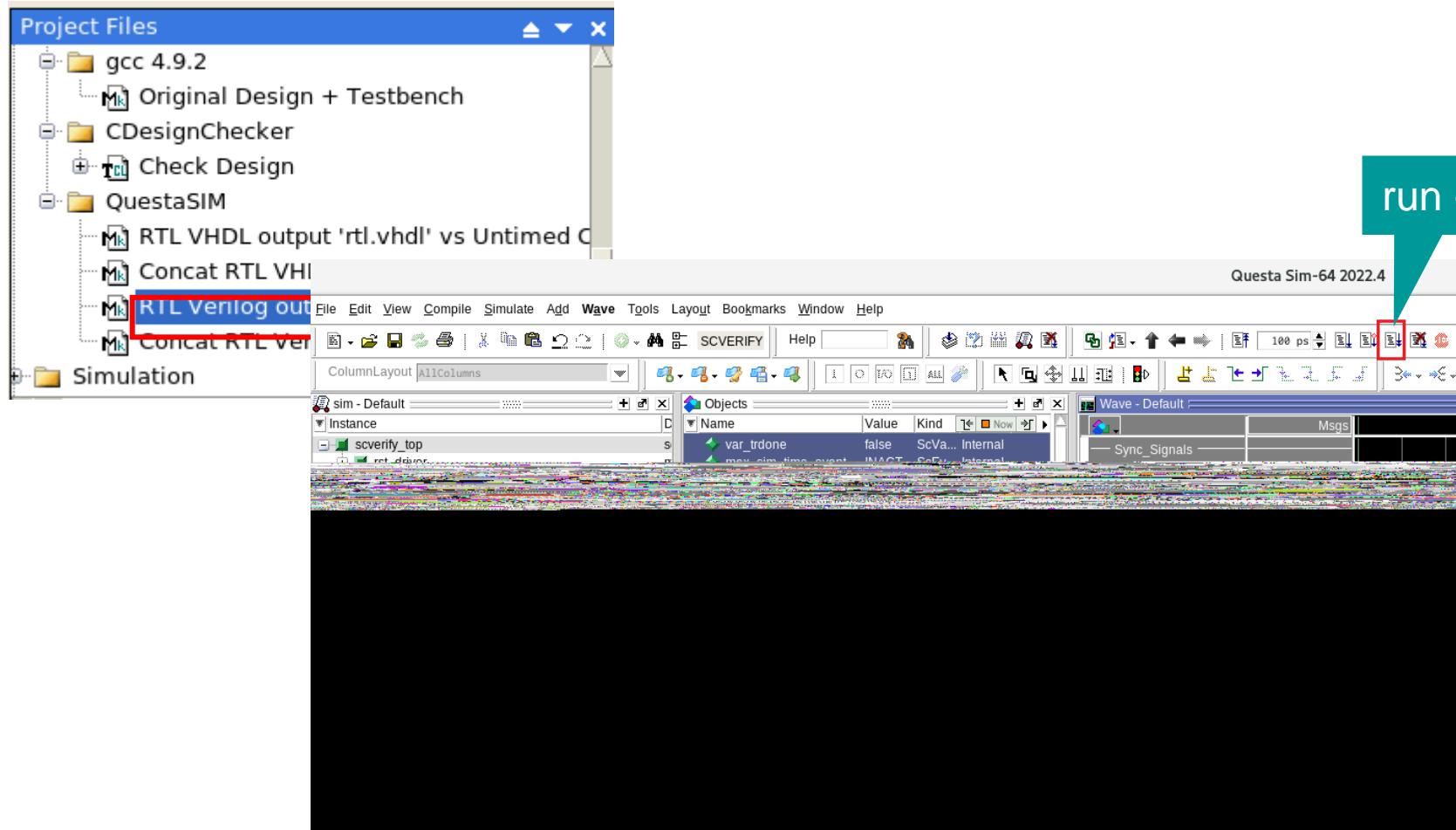
Design Analyzer

Analyze the design in various perspectives



RTL Verification

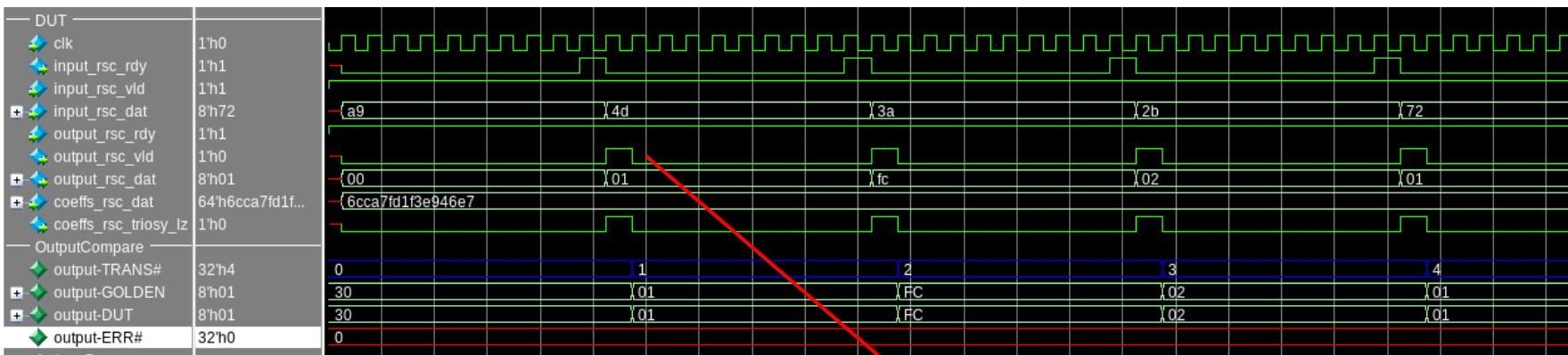
Automatically compare the outputs of HLC C and RTL



RTL Verification

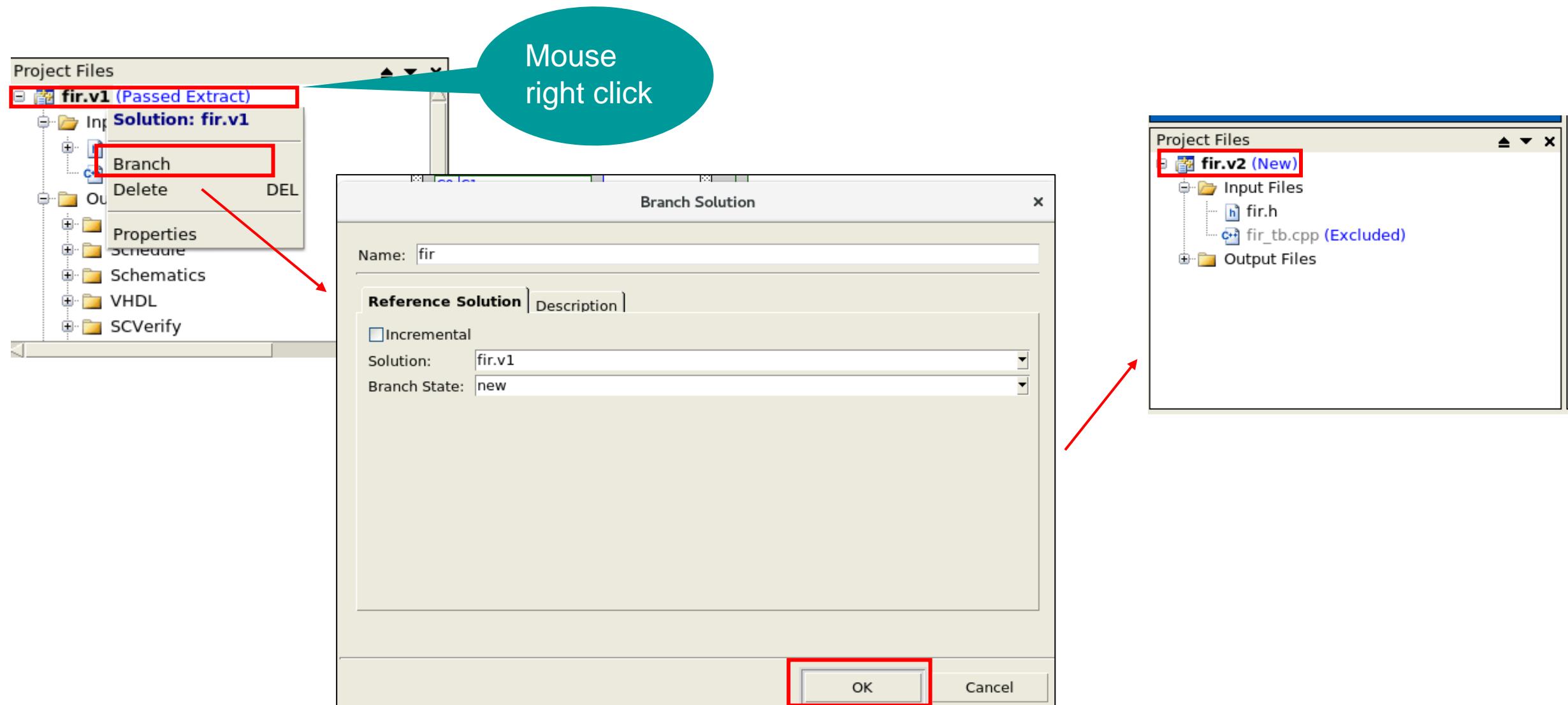
Automatically compare the outputs of HLC C and RTL

```
Transcript
#   captured 100 values of output
# Info: scverify_top/user_tb: Simulation completed
#
# Checking results
# 'output'
#   capture count      = 100
#   comparison count   = 100
#   ignore count       = 0
#   error count        = 0
#   stuck in dut fifo = 0
#   stuck in golden fifo = 0
#
# Info: scverify_top/user_tb: Simulation PASSED @ 10016 ns
# ** Note: (vsim-6574) SystemC simulation stopped by user.
# 1
#
VSIM 3>
```

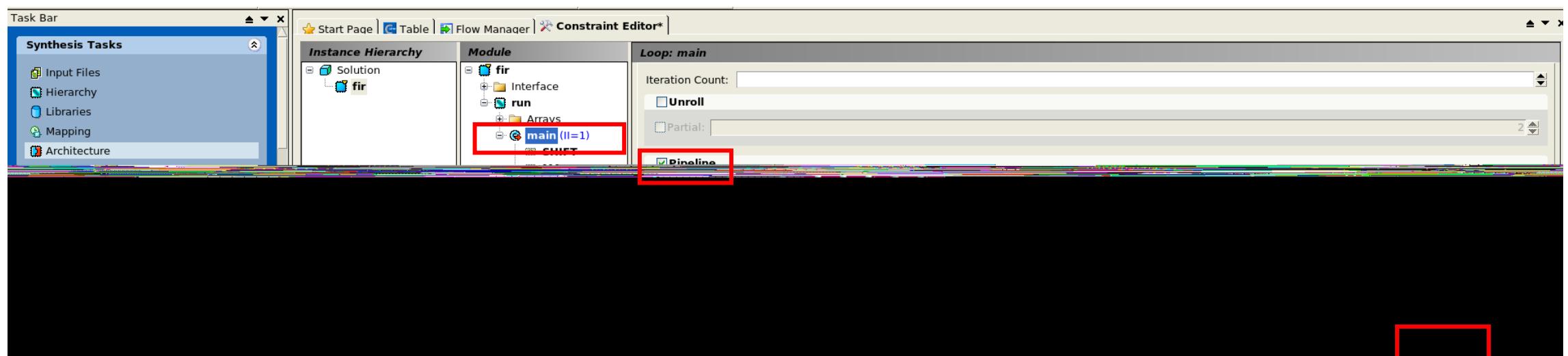
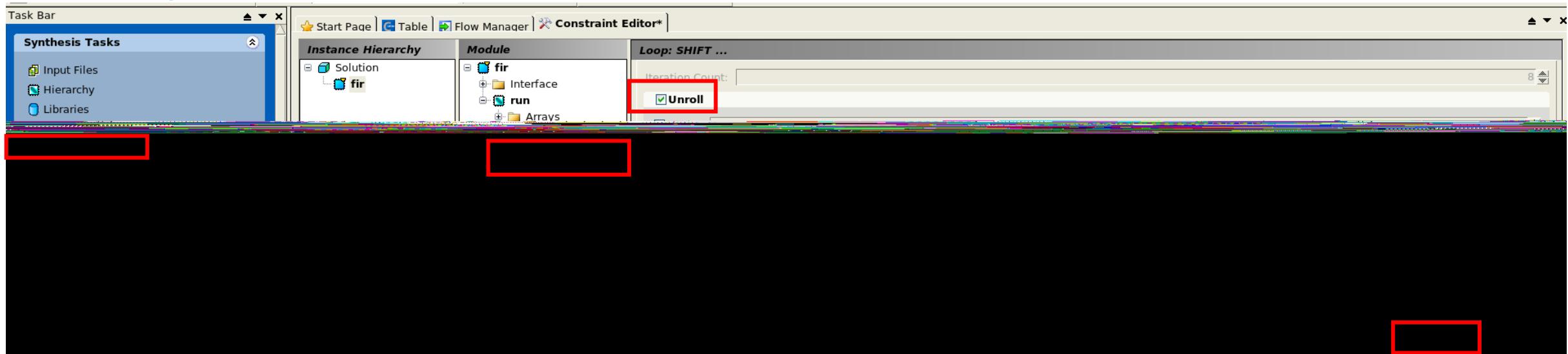


| Loop Unrolling and Pipelining

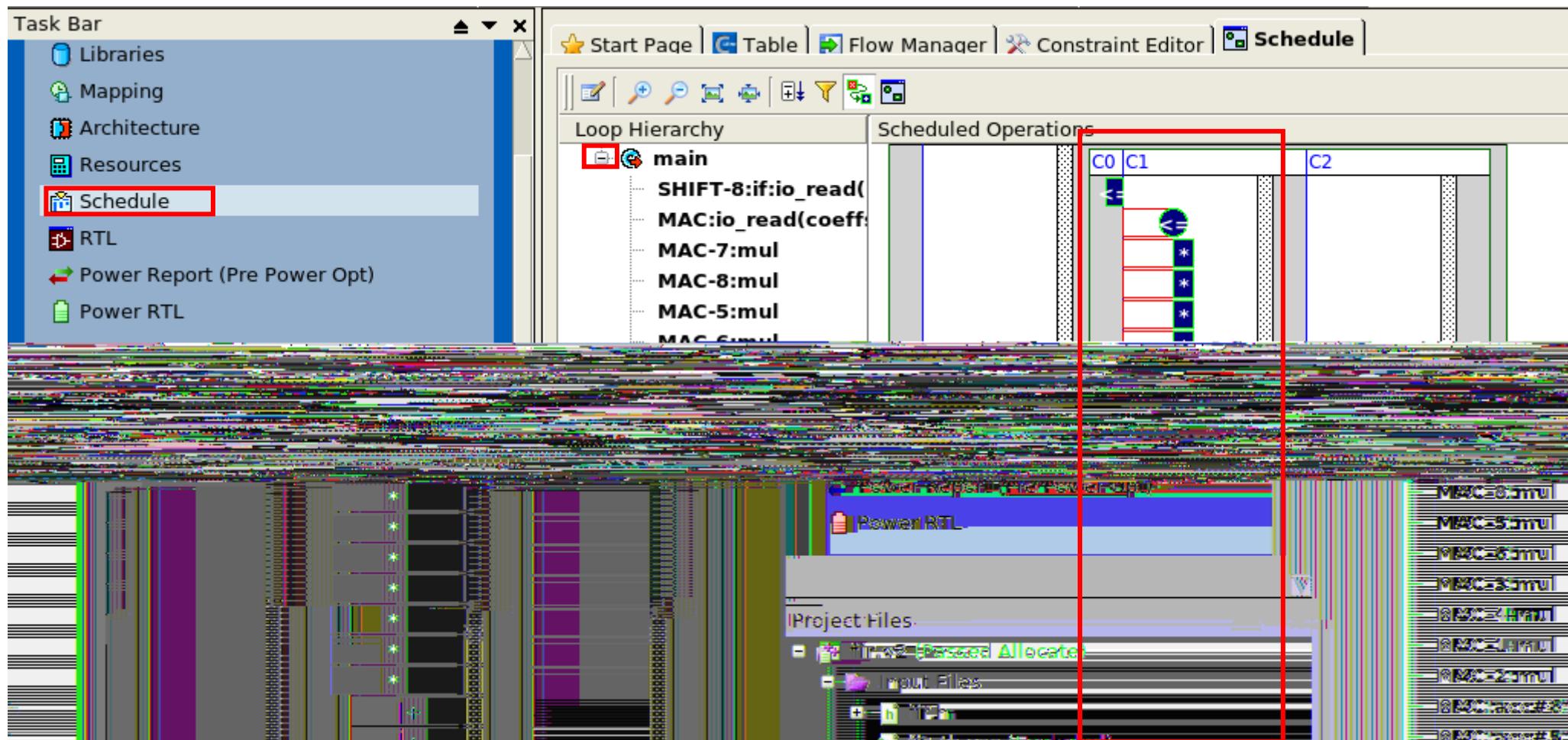
Branch a New Solution



HLS Synthesis – Architecture



HLS Synthesis – Scheduling



HLS Synthesis – RTL

The screenshot shows the HLS Synthesis software interface. The left sidebar, titled "Task Bar", contains icons for Libraries, Mapping, Architecture, Resources, Schedule, and RTL, with "RTL" being the active tab and highlighted by a red box. Below the Task Bar is a large black workspace area. The top menu bar includes "Start Page", "Table", "Flow Manager", "Constraint Editor", "Schedule", and "rtl.v". The main code editor window displays the following Verilog-like code:

```
// Design Unit: fir
// -----
module fir (
    clk, rst, input_rsc_dat, input_rsc_vld, input_rsc_rdy, coeffs_rsc_dat, coeffs_rsc_triosy_lz,
    output_rsc_dat, output_rsc_vld, output_rsc_rdy
);
```

A red rectangular box highlights the entire module definition. A smaller red box is also present at the bottom left of the black workspace.

HLS Synthesis – Report

The screenshot shows the HLS Synthesis software interface. On the left is the Task Bar with the following items:

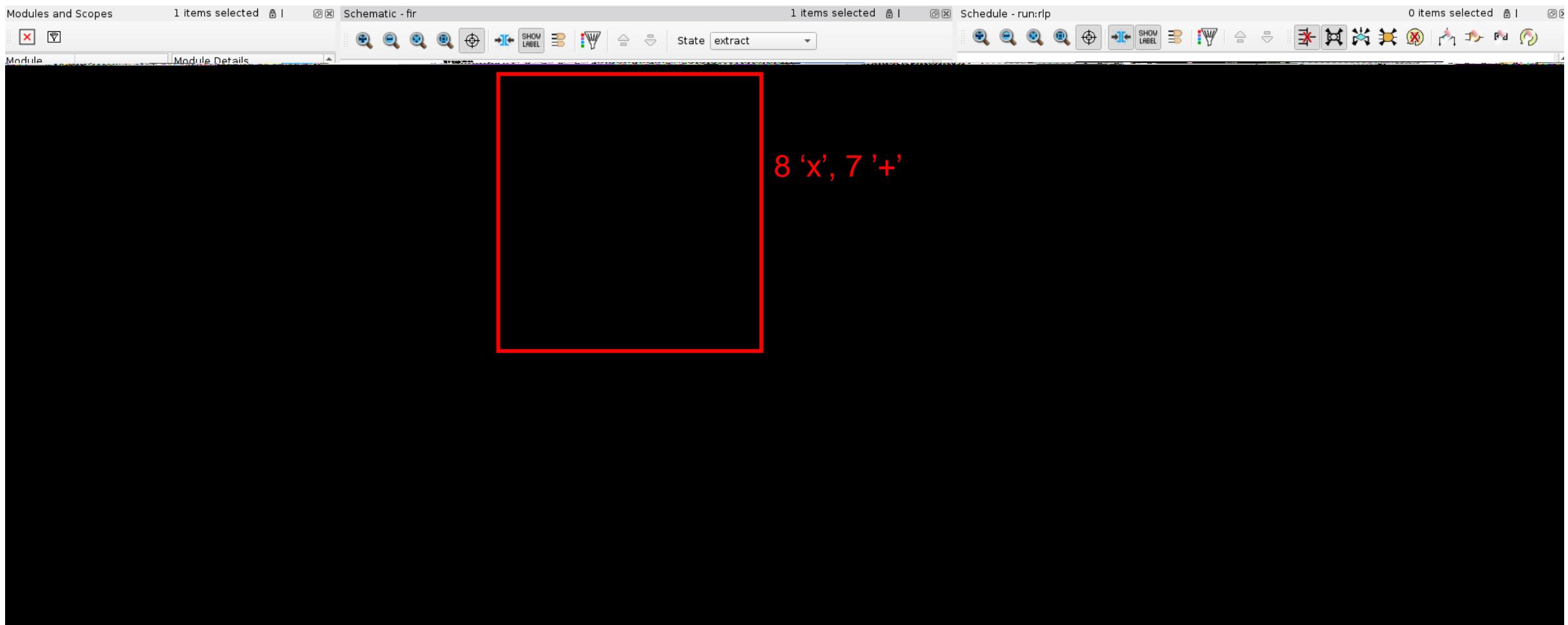
- Libraries
- Mapping
- Architecture
- Resources
- Schedule
- RTL** (highlighted)
- Power Report (Pre Power Opt)
- Power RTL

The main window displays a Table view titled "Table". The table has a red border around its header and data rows. The columns are labeled: Solution, Latency..., Latency..., Through..., Through..., Slack, and Total Area.

Solution	Latency...	Latency...	Through...	Through...	Slack	Total Area
fir.v1 (extract)	8	80.00	10	100.00	6.49	1422.44
fir.v2 (extract)	1	10.00	1	10.00	4.82	3355.97

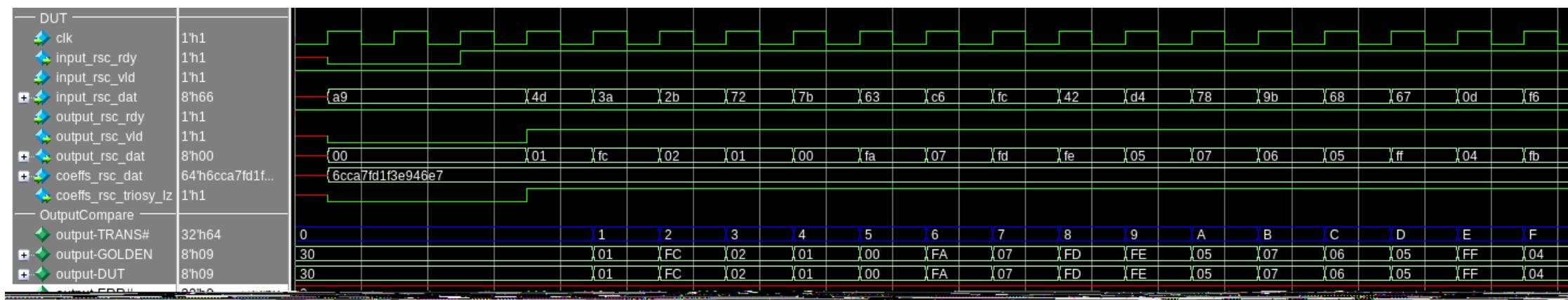
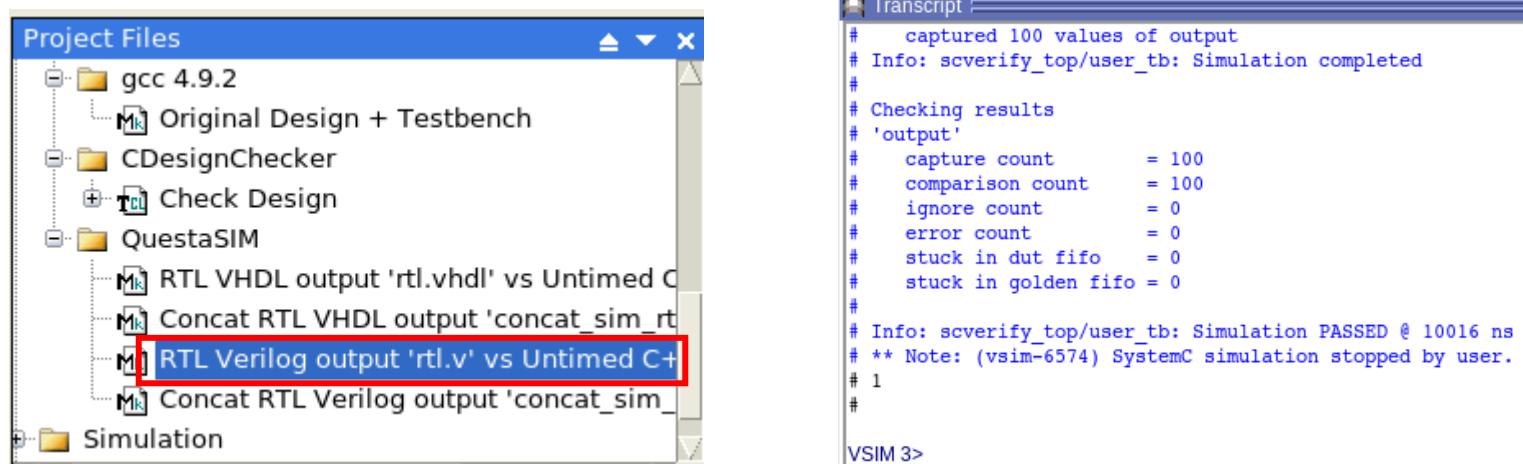
Design Analyzer

Analyze the design in various perspectives



RTL Verification

Automatically compare the outputs of HLC C and RTL



Reproduce the Solution

All command has been recorded in a tcl script

```
[mentor@RHEL74 walkthrough]$ ll ./Catapult/fir.v2/directives.tcl
-rw-rw-r-- 1 mentor mentor 3531 Aug 22 11:45 ./Catapult/fir.v2/directives.tcl

go new
go compile
solution library add nangate-45nm_beh -- -rtlsvyntool DesignCompiler -vendor Nangate -technology 045nm
go libraries
directive set -CLOCKS {clk {-CLOCK_PERIOD 10.0 -CLOCK_EDGE rising -CLOCK_UNCERTAINTY 0.0 -CLOCK_HIGH_TIME 5.0}}
go assembly
directive set /fir/run/main -PIPELINE_INIT_INTERVAL 1
directive set /fir/run/SHIFT -UNROLL yes
```

The same result can be produced with this .tcl

- catapult –f ./Catapult/fir.v2/directives.tcl &

Memory Interface

Source Code (fir.h)

```
void CCS_BLOCK(run)(ac_channel<ac_int<8>> &input,
                     ac_int<8> coeffs[32][8],
                     ac_channel<ac_int<5, false>> &coeff_addr,
                     ac_channel<ac_int<8>> &output) {
    ac_int<19> temp = 0;
    ac_int<5, false> addr = coeff_addr.read();
```

```
SHIFT: for (int i = 7; i >= 0; i--) {
    if (i == 0) {
        regs[i] = input.read();
    } else {
        regs[i] = regs[i - 1];
    }
}
```

```
MAC: for (int i = 7; i >= 0; i--) {
    temp += coeffs[addr][i] * regs[i];
}
output.write(temp >> 11);
}
```

TestBench (fir_tb.cpp)

```
#include "fir.h"

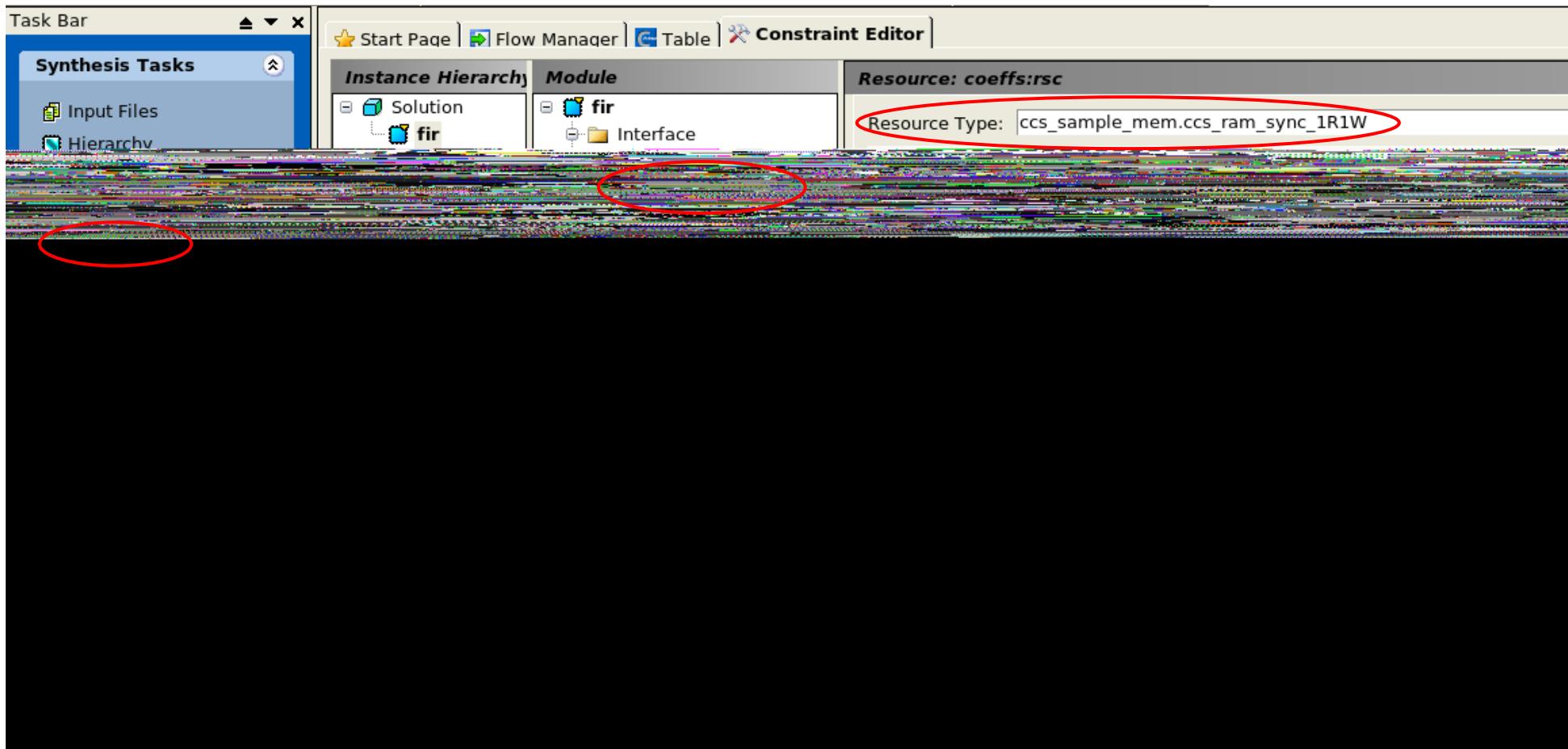
CCS_MAIN(int argc, char *argv[])
{
    ac_int<8> coeffs[32][8];
    ac_channel<ac_int<8>> input;
    ac_channel<ac_int<5, false>> coeff_addr;
    ac_channel<ac_int<8>> output;
    ac_int<8> data;
    ac_int<5, false> addr;
    fir_inst;
    ...
}
```

```
...
// Test Impulse
for (int j=0; j < 32; j++)
    for (int i=0; i < 8; i++)
        coeffs[j][i] = rand();
for (int i = 0; i < 10; i++) {
    data = rand();
    input.write(data);
    addr = rand();
    coeff_addr.write(addr);
    inst.run(input, coeffs, coeff_addr, output);
}
while (output.available(1))
    printf("Output = %3d\n", output.read().to_int());
CCS_RETURN(0);
}
```

Start with running directives.tcl

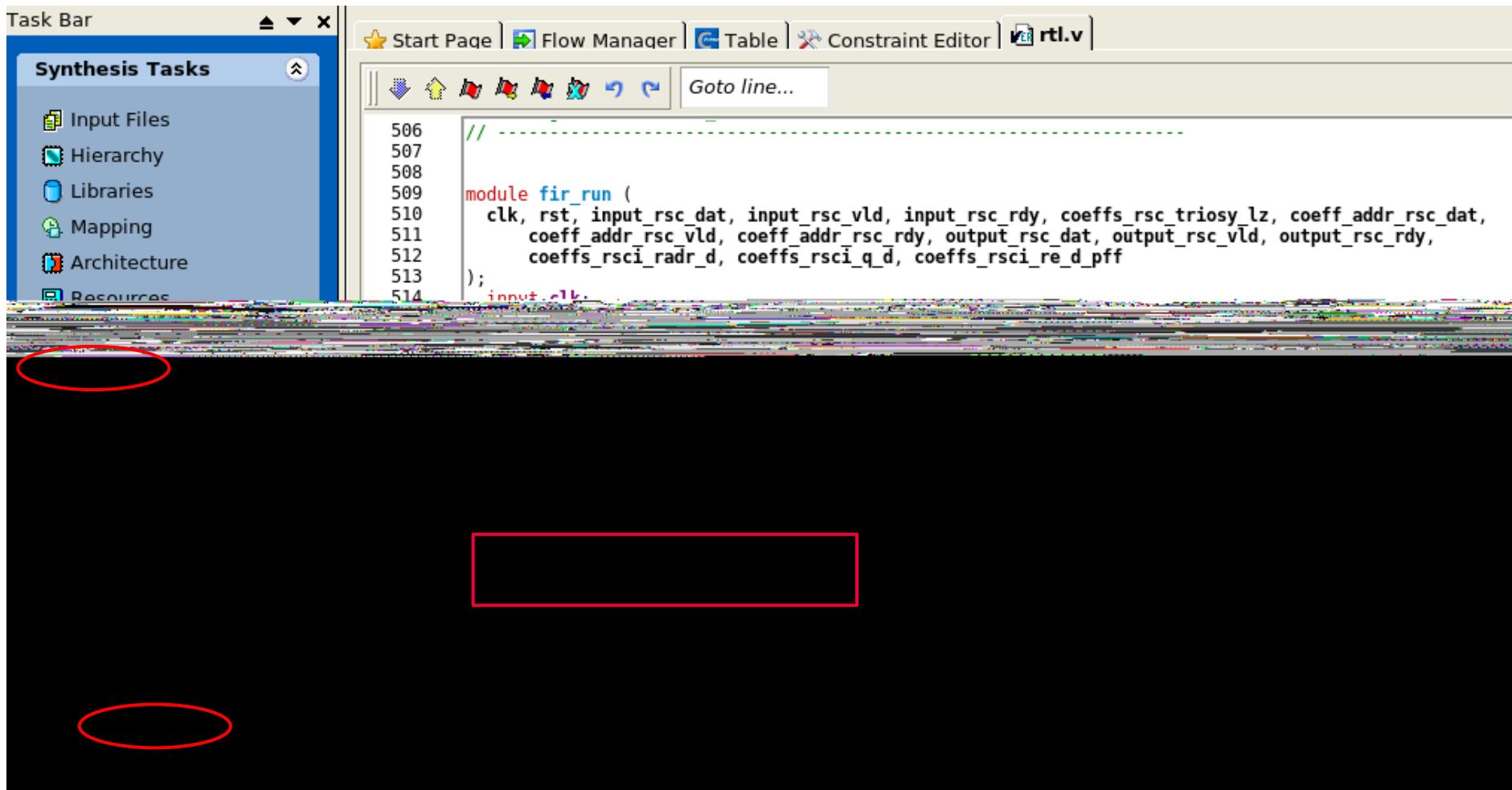
catapult –f directives.tcl

Check coeffs SRAM: (1) size: 256x8b (2) resource type: 1R1W RAM



Check SRAM Interface in RTL

re: ram read enable; *_radr_*: ram read address; *_d*: ram read data

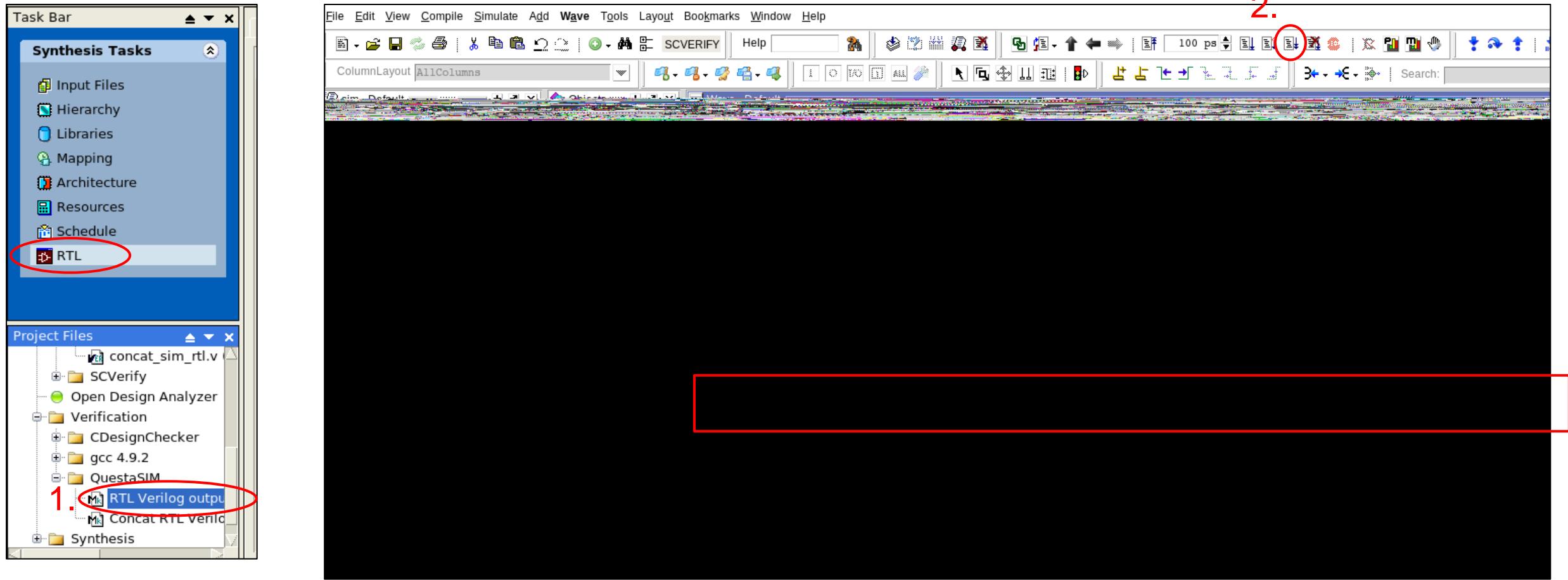


The screenshot shows a software interface for digital design. On the left, a 'Task Bar' window titled 'Synthesis Tasks' lists several options: Input Files, Hierarchy, Libraries, Mapping, Architecture, and Resources. A red oval highlights the 'Input Files' item. The main area is a code editor window titled 'rtl.v'. The code is a Verilog module definition:

```
// -----
module fir_run (
    clk, rst, input_rsc_dat, input_rsc_vld, input_rsc_rdy, coeffs_rsc_triosy_lz, coeff_addr_rsc_dat,
    coeff_addr_rsc_vld, coeff_addr_rsc_rdy, output_rsc_dat, output_rsc_vld, output_rsc_rdy,
    coeffs_rsci_radr_d, coeffs_rsci_q_d, coeffs_rsci_re_d_pff
);
    input clk;
```

A red rectangle highlights the entire code block, and another red oval highlights the 'clk' input port.

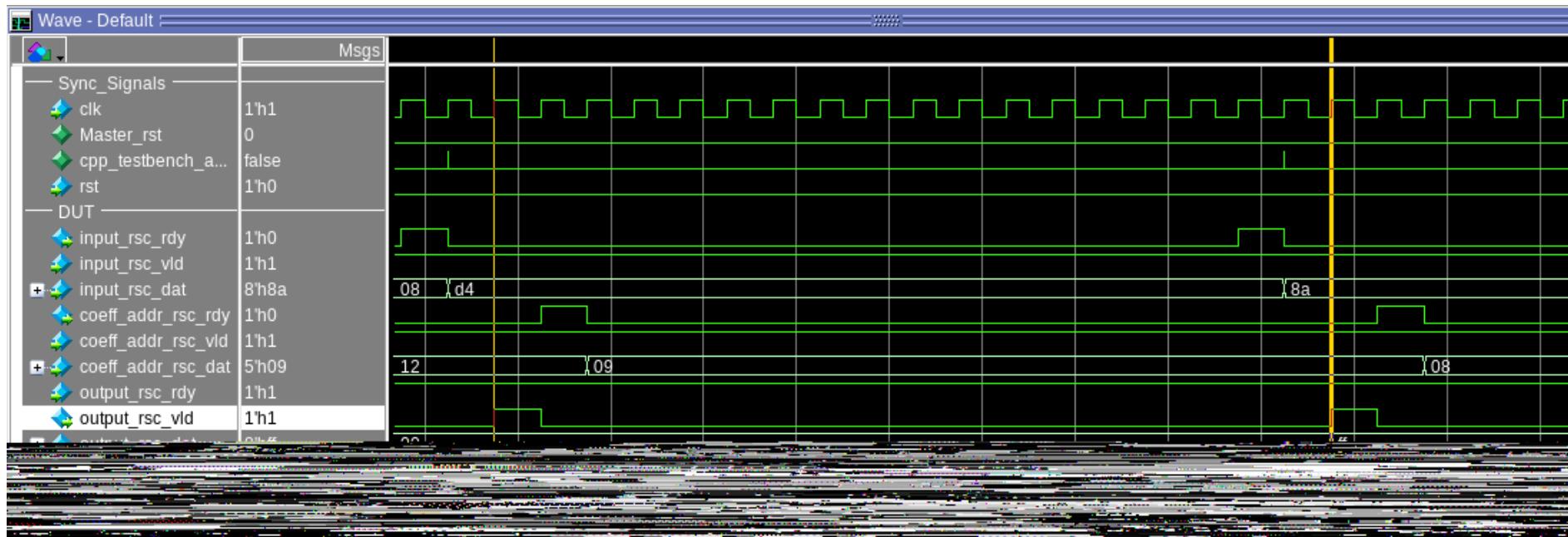
Check SRAM Interface in Waveform



How do we improve the throughput

Current throughput is 18

Solution	Latency...	Latency...	Throughput Cycles	Thro...	Slack	Total Area
solution.v1 (new)						
fir.v1 (extract)	17	170.00	18	180.00	7.68	1461.69



Make convolution in parallel

```
void CCS_BLOCK(run) (ac_channel <ac_int<8>> &input,
                      ac_int<8> coeffs[32][8],
                      ac_channel <ac_int<5, false>> &coeff_addr,
                      ac_channel <ac_int<8>> &output) {
    ac_int<19> temp = 0;
    ac_int<5, false> addr = coeff_addr.read();

    SHIFT: for (int i = 7; i >= 0; i--) {
        if (i == 0) {
            regs[i] = input.read();
        } else {
            regs[i] = regs[i - 1];
        }
    }

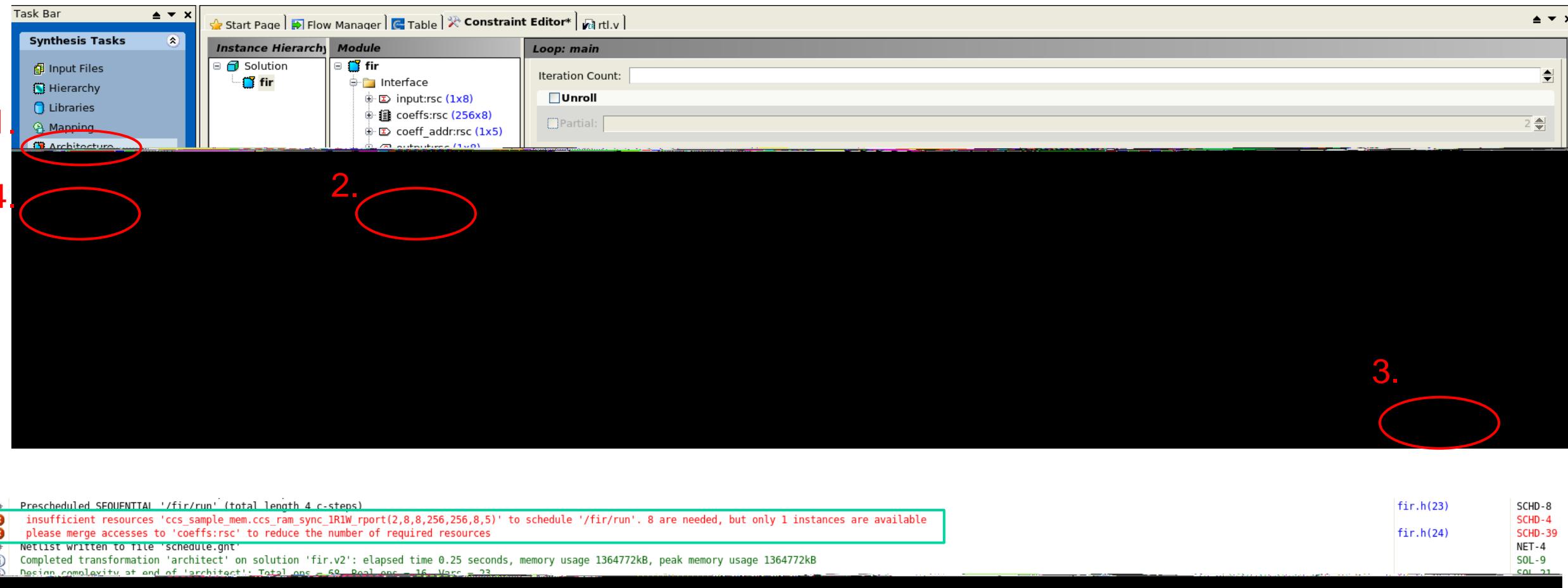
    MAC: for (int i = 7; i >= 0; i--) {
        temp += coeffs[addr][i] * regs[i];
    }

    output.write(temp >> 11);
}
```

Make convolution parallel

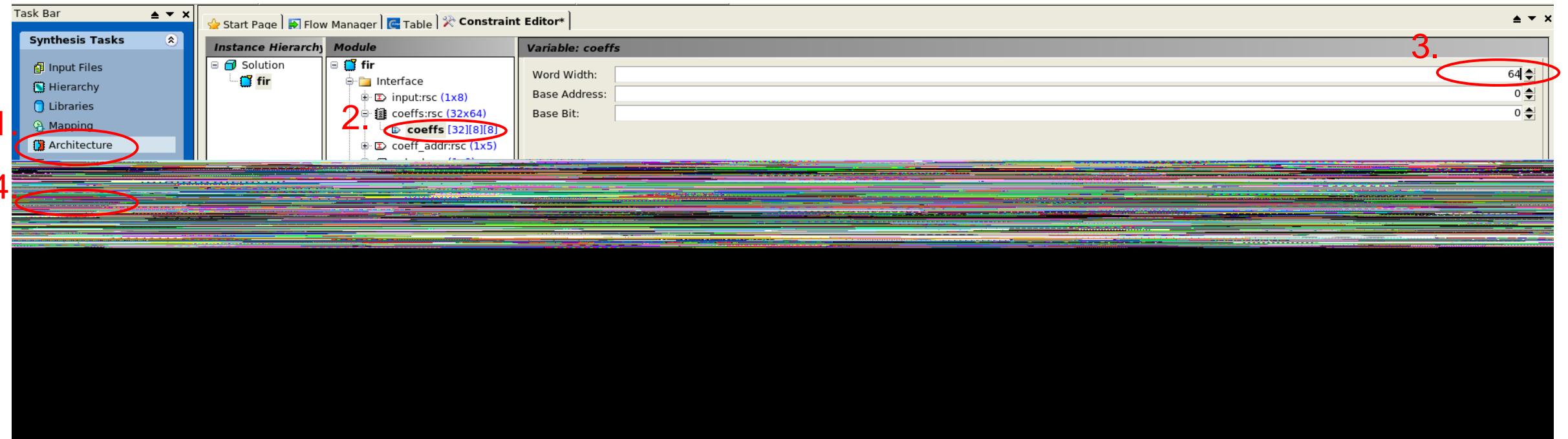
Change the Architecture

Unroll 'SHIFT' and 'MAC', and Pipeline 'main'



Increase Memory Data Width

Word Width = 64 (why ?)

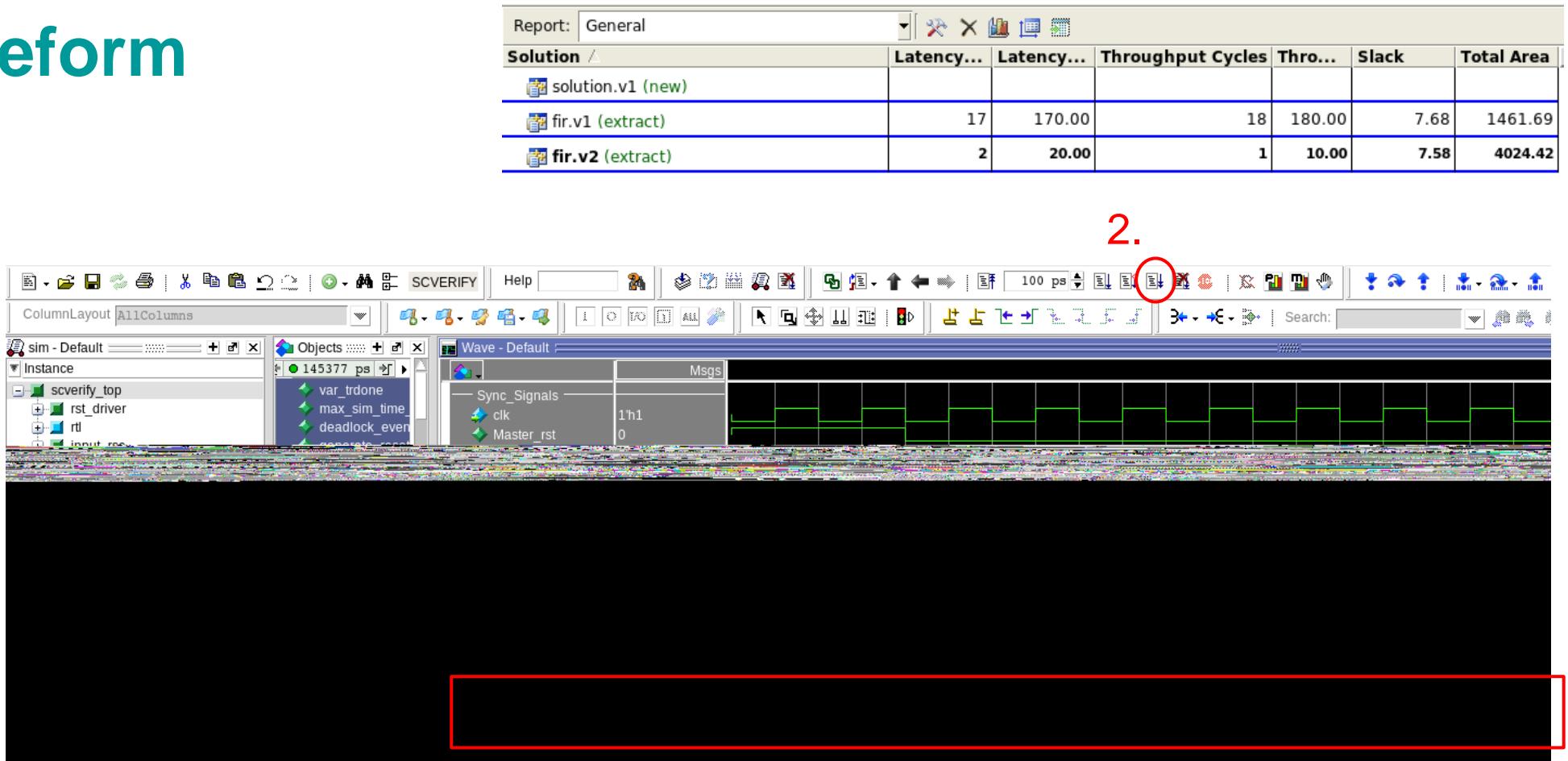
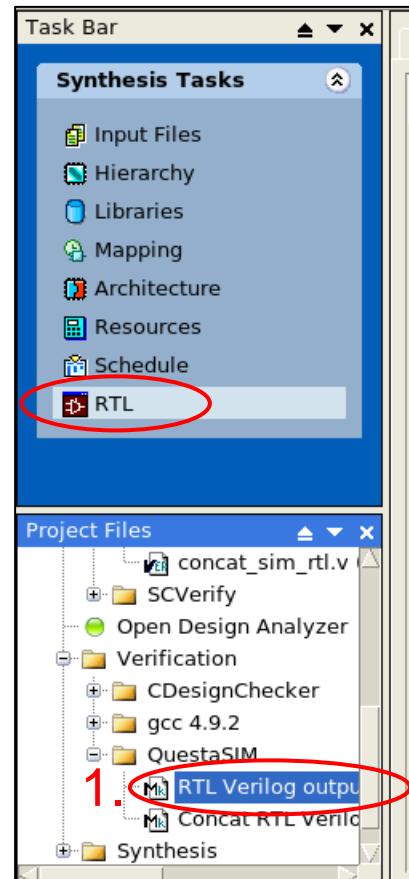


```
# Generating JTAGVerify testbench files
# Makefile for RTL Verilog output 'rtl.v' vs Untimed C++ written to file './scverify/Verify_rtl_v_msim.mk'
# Makefile for Concat RTL Verilog output 'concat_sim_rtl.v' vs Untimed C++ written to file './scverify/Verify_concat_sim_rtl_v_msim.mk'
(i) Completed transformation 'extract' on solution 'fir.v3': elapsed time 3.66 seconds, memory usage 1364772kB, peak memory usage 1364816kB
(i) Design complexity at end of 'extract': Total ops = 269, Real ops = 84, Vars = 199
```

SOL-9
SOL-21

Check Waveform

Throughput = 1



Multiple Blocks

A Design with Two 8-Tap FIRs and One Decimator

top.h

```
#include "fir.h"
#include "decimator.h"
class top {
    ac_channel<ac_int<8>> connect0;
    ac_channel<ac_int<8>> connect1;
    fir_block0;
    fir_block1;
    decimator_block2;
public:
    top() {}
#pragma HLS_design_interface top
void CCS_BLOCK(run) (ac_channel<ac_int<8>> &din,
                     ac_int<8> coeffs[8],
                     ac_channel<ac_int<8>> &dout)
{
    block0.run(din, coeffs, connect0);
    block1.run(connect0, coeffs, connect1);
    block2.run(connect1, dout);
    ...
}
```

decimator.h

```
class decimator {
    int count;
public:
    decimator () {
        count = 0;
    }
    #pragma hls_design_interface
    void CCS_BLOCK(run) (ac_channel<ac_int<8>> &din,
                         ac_channel<ac_int<8>> &dout) {
        if (count==4)
            count = 0;
        ac_int<8> temp = din.read();
        if (count==0)
            dout.write(temp);
        count++;
    ...
}
```

fir.h

```
class fir {
    ac_int<8> regs[8];
public:
    fir() {
        for (int i = 7; i >= 0; i--)
            {regs[i] = 0; }
    }
    ...
}
```

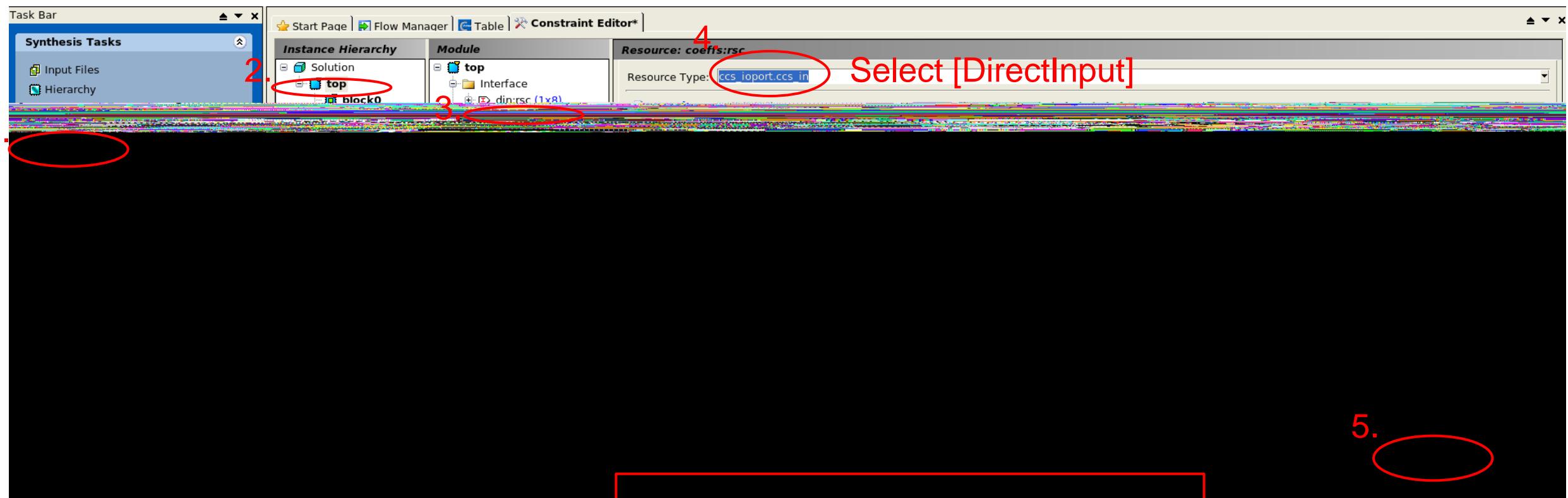
```
#pragma hls_design_interface
void CCS_BLOCK(run)(ac_channel< ac_int<8> > &input,
                     ac_int<8> coeffs[8],
                     ac_channel< ac_int<8> > &output) {
    ac_int<19> temp = 0;
    SHIFT: for (int i = 7; i >= 0; i--) {
        if (i == 0) {
            regs[i] = input.read();
        } else {
            regs[i] = regs[i - 1];
        }
    }
    MAC: for (int i = 7; i >= 0; i--) {
        temp += coeffs[i]*regs[i];
    }
    output.write(temp>>11);
    ...
}
```

Start from the ‘Architecture’ Stage

catapult -f directives.tcl &

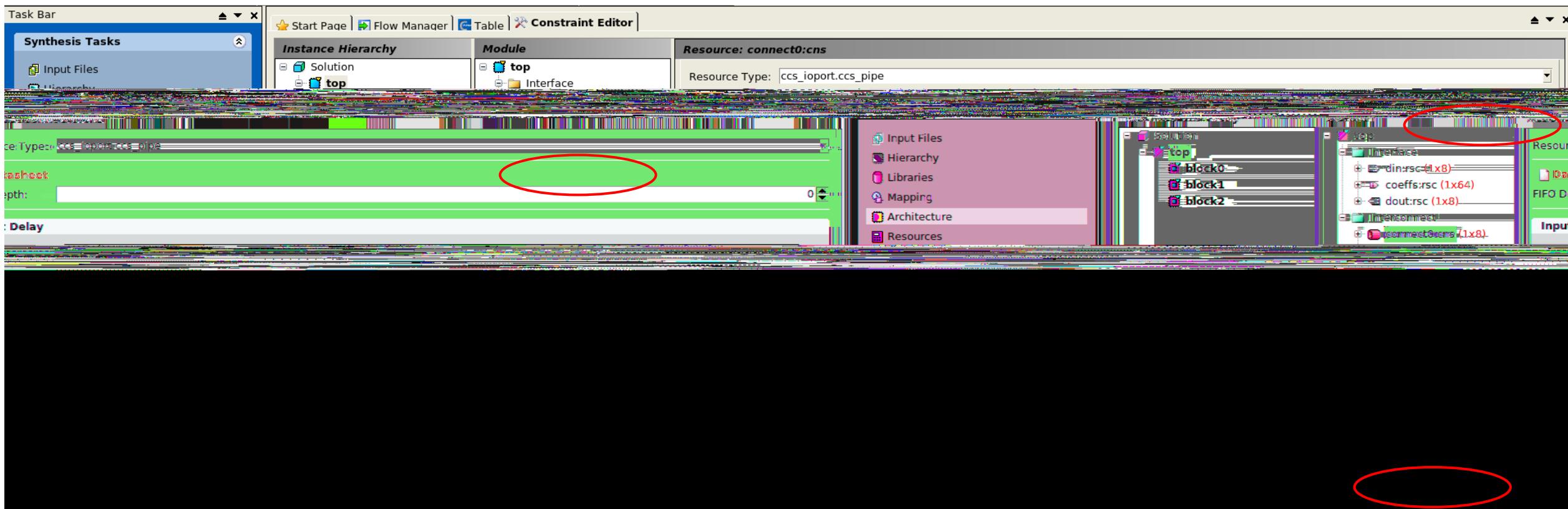
There are three design blocks: block0, block1, block2

Set ‘DirectInput’ for the coeffs



Set FIFO Depth as '0'

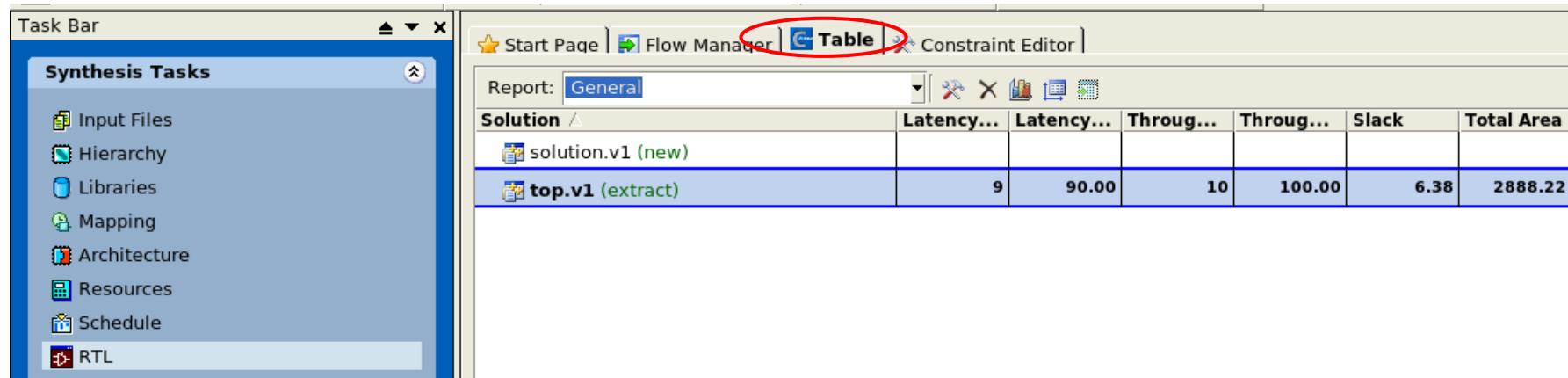
Set FIFO depth as '0' for connect0 and connect1



Look Report

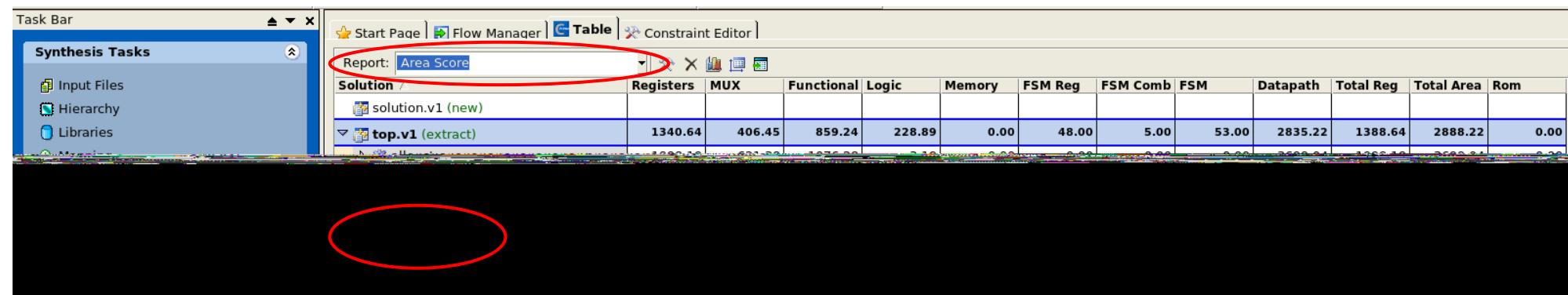
Latency = ?, Throughput = ?

Area: block0=? , block1=? , block2=?



The screenshot shows the Synthesis Tasks interface with the Table tab selected. The report is set to General. The table displays two solutions: 'solution.v1 (new)' and 'top.v1 (extract)'. The columns represent Latency, Throughput, Slack, and Total Area. The 'top.v1 (extract)' row shows values 9, 90.00, 10, 100.00, 6.38, and 2888.22 respectively.

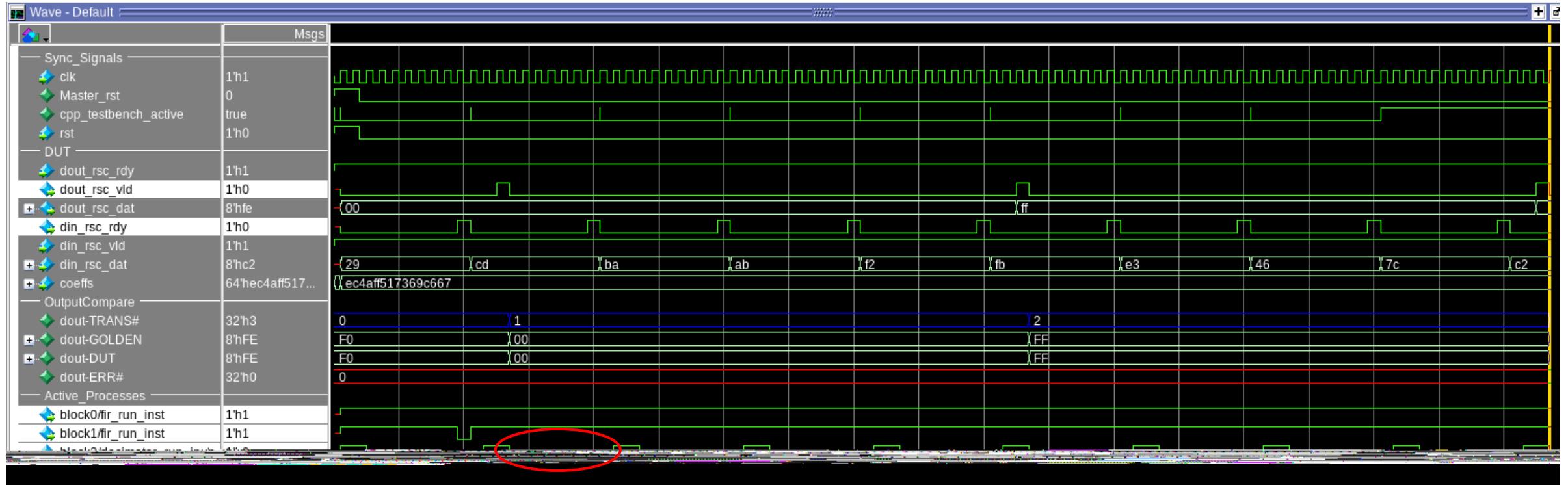
Solution	Latency...	Latency...	Throug...	Throug...	Slack	Total Area
solution.v1 (new)						
top.v1 (extract)	9	90.00	10	100.00	6.38	2888.22



The screenshot shows the Synthesis Tasks interface with the Table tab selected and the report set to Area Score. The table displays two solutions: 'solution.v1 (new)' and 'top.v1 (extract)'. The columns represent Registers, MUX, Functional, Logic, Memory, FSM Reg, FSM Comb, FSM, Datapath, Total Reg, Total Area, and Rom. The 'top.v1 (extract)' row shows values 1340.64, 406.45, 859.24, 228.89, 0.00, 48.00, 5.00, 53.00, 2835.22, 1388.64, 2888.22, and 0.00 respectively. A red oval highlights the bottom of the interface.

Solution	Registers	MUX	Functional	Logic	Memory	FSM Reg	FSM Comb	FSM	Datapath	Total Reg	Total Area	Rom
solution.v1 (new)												
top.v1 (extract)	1340.64	406.45	859.24	228.89	0.00	48.00	5.00	53.00	2835.22	1388.64	2888.22	0.00

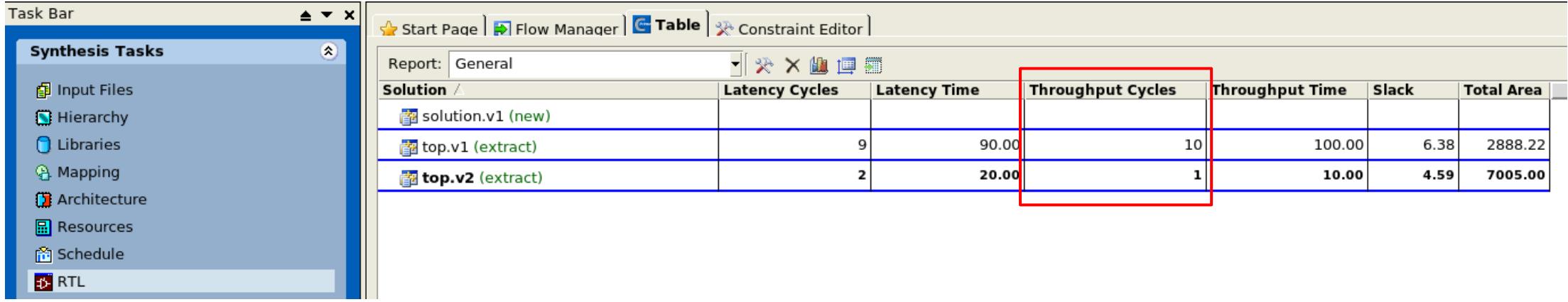
RTL Simulation



why idle ?

Improve Throughput

Improve the throughput from 10 to 1 by yourself



The screenshot shows the Task Bar interface with the 'Synthesis Tasks' panel open. The 'Table' tab is selected in the top navigation bar. The main area displays a report table with the following data:

Solution	Latency Cycles	Latency Time	Throughput Cycles	Throughput Time	Slack	Total Area
solution.v1 (new)						
top.v1 (extract)	9	90.00	10	100.00	6.38	2888.22
top.v2 (extract)	2	20.00	1	10.00	4.59	7005.00

RTL Simulation (Throughput=1)

One dout every four din because of the decimation

