# Step-By-Step Lab1: FIR

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



SIEMENS

# Objectives

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




SIEMENS

# Environment Variables

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

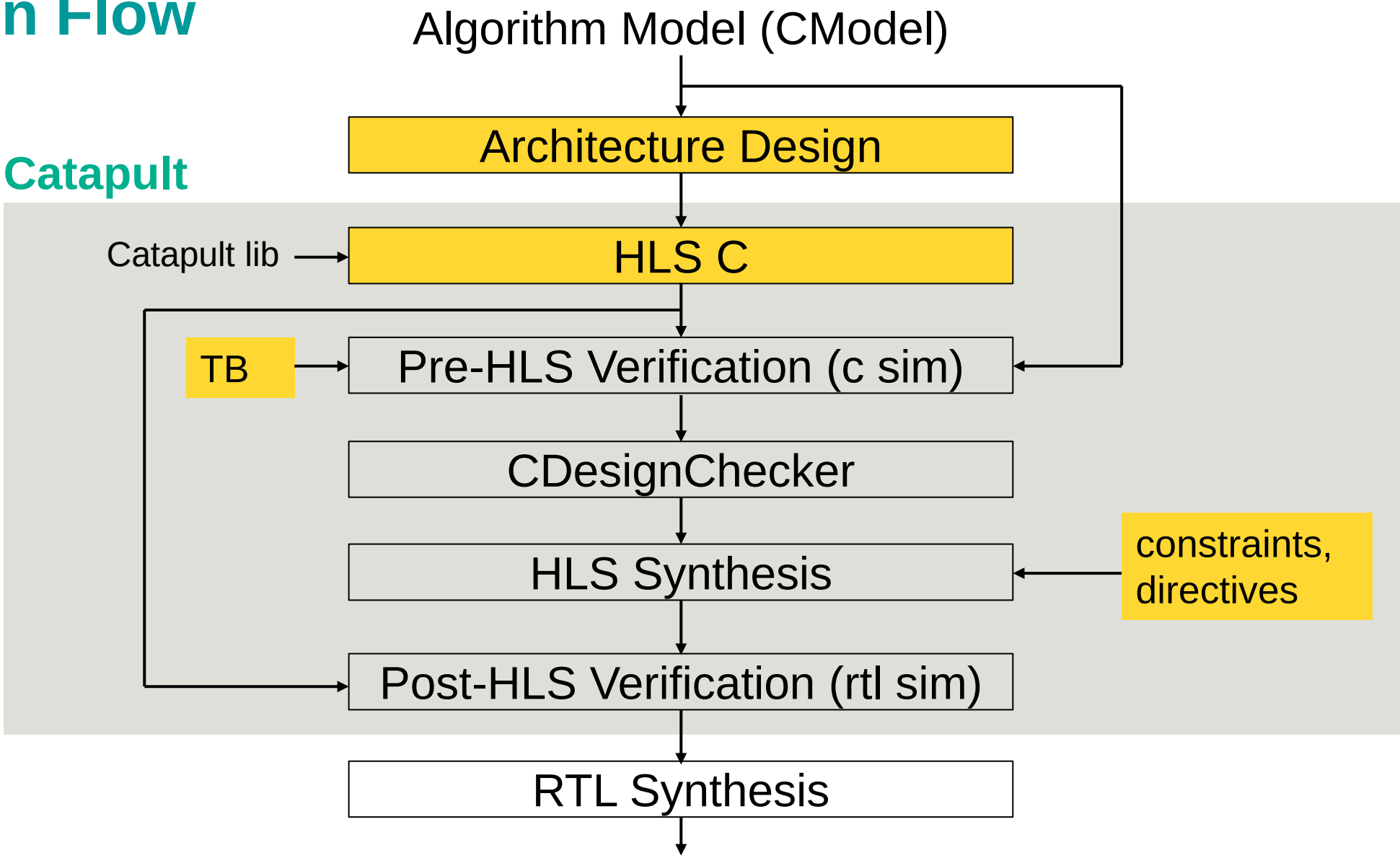
- Define them in a shell script like setup-catapult.csh and source it
- Modify the path depending on the tool install path in your workstation

```
 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"
```

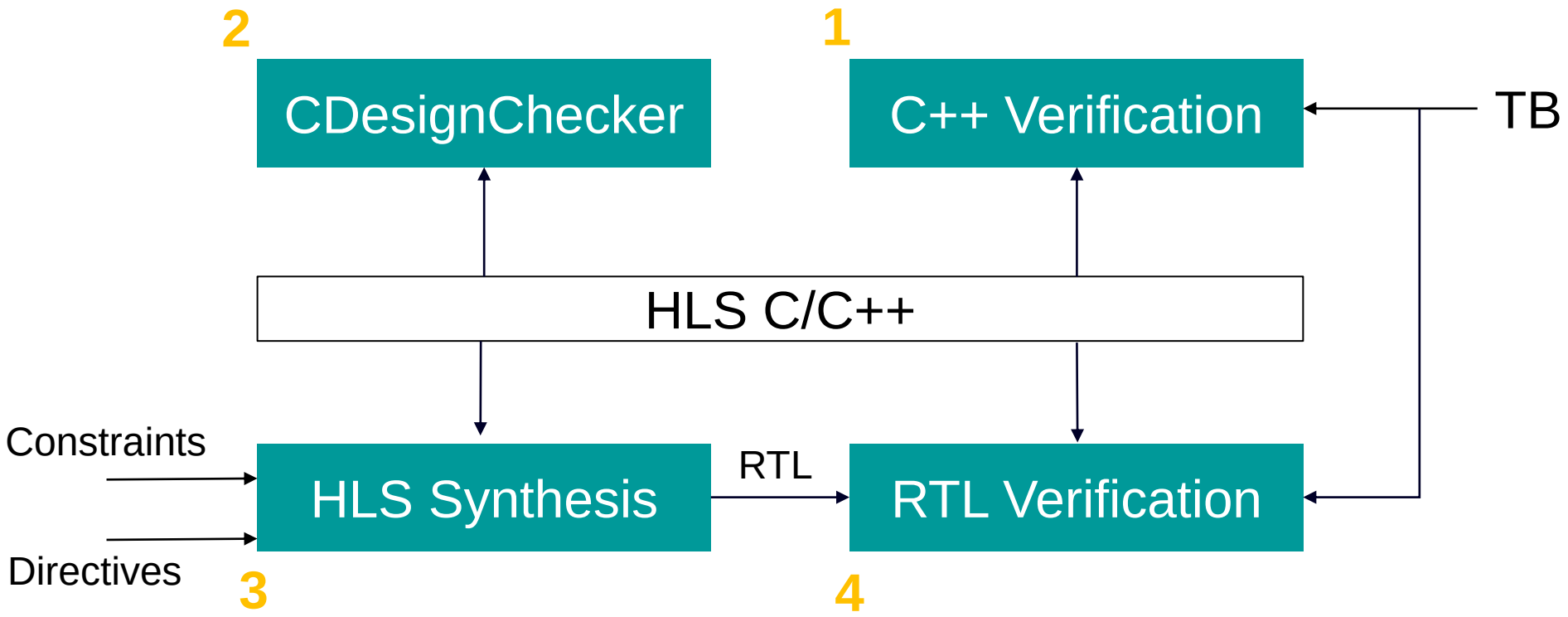



SIEMENS

# Design Flow



# Catapult Flow



# Walkthrough of Catapult C++ Flow

An 8-tap FIR




SIEMENS

# fir_ref.h (Reference model)

$$y = \sum_{i=0}^{7} c_i \cdot r[i]$$
$$= c_0 r[0] + c_1 r[1] + \ldots + c_7 r[7]$$

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

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

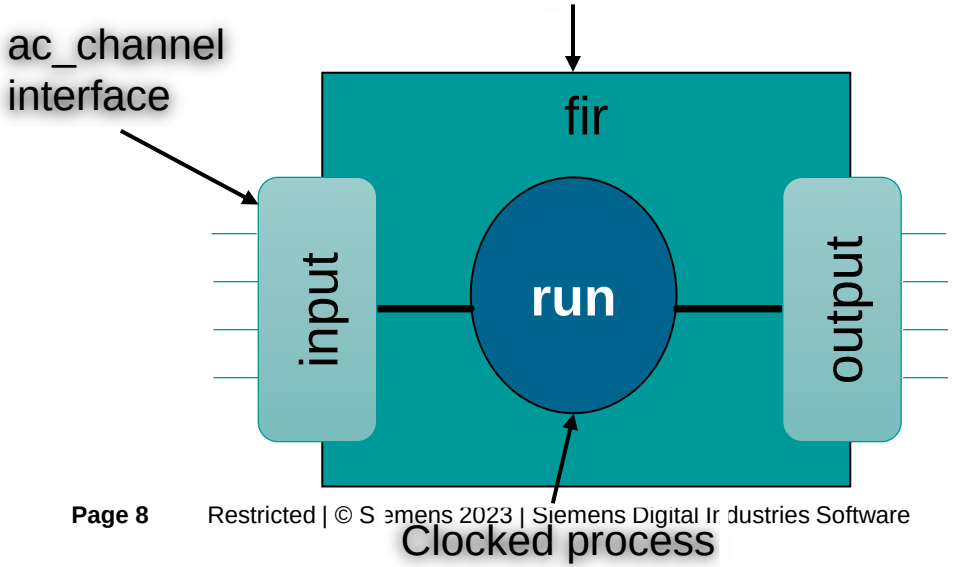
```
void run(int input, int coeffs[8], int &output)
{
    int temp = 0;

    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




label

```
#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




SIEMENS

# 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

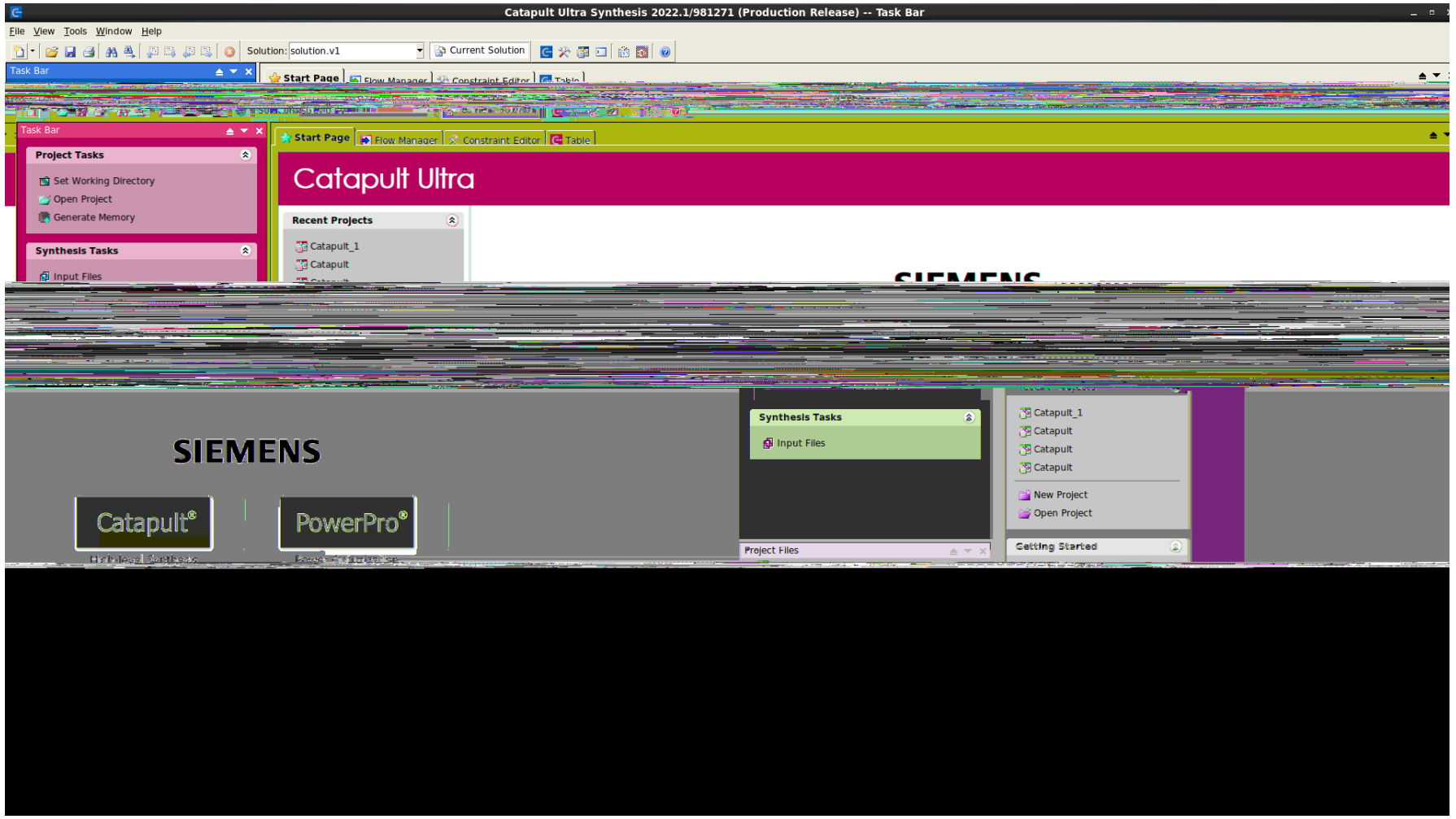
```
pass @  89:    3 ==    3
pass @  90:  -18 ==  -18
pass @  91:   21 ==   21
pass @  92:  -12 ==  -12
pass @  93:    2 ==    2
pass @  94:   -6 ==   -6
pass @  95:    5 ==    5
pass @  96:  -21 ==  -21
pass @  97:   18 ==   18
pass @  98:   -5 ==   -5
pass @  99:    9 ==    9

total pass count 100
total fail count 0
```




SIEMENS

# Launch Catapult

catapult &

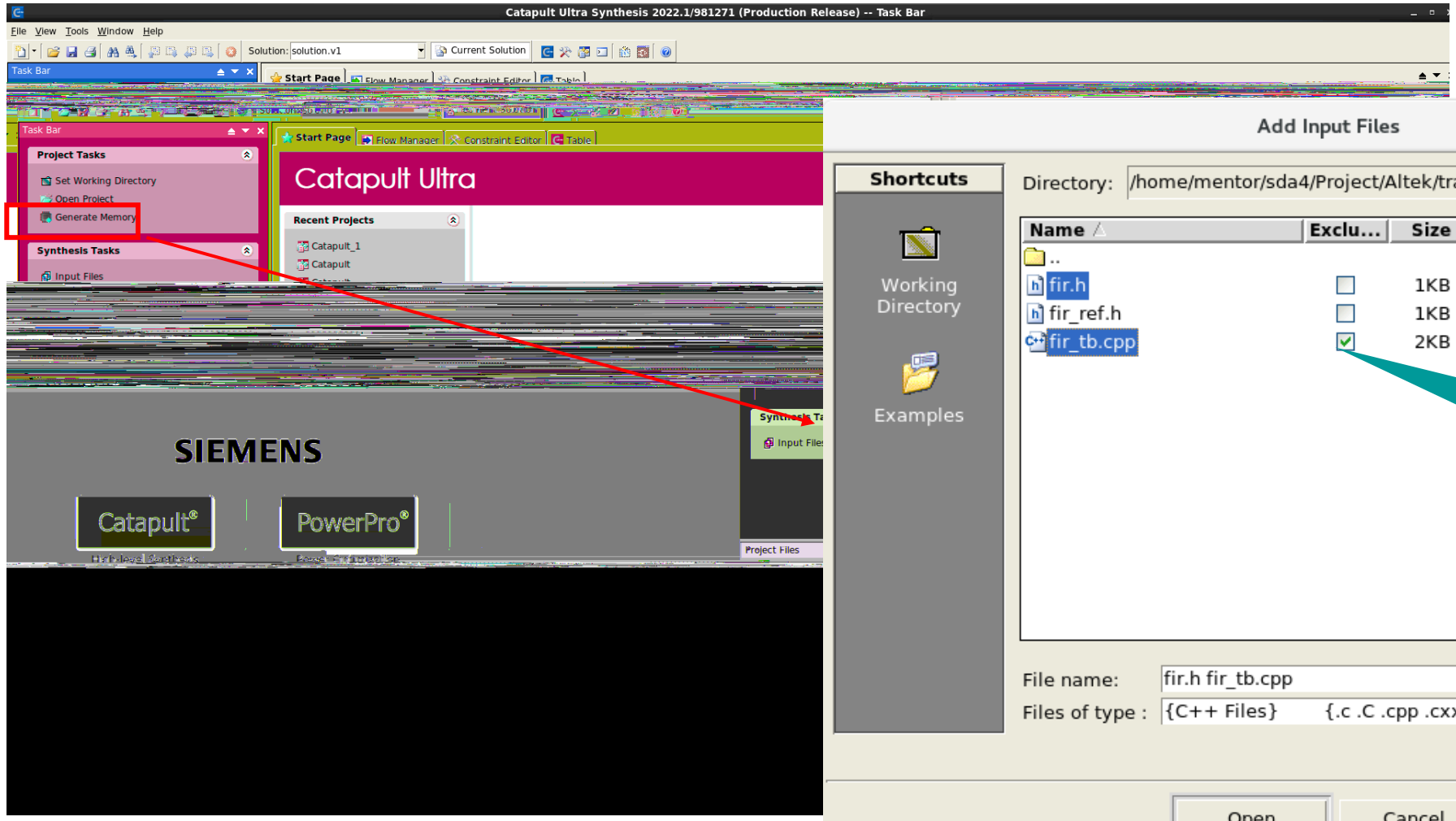






SIEMENS

# Add input file





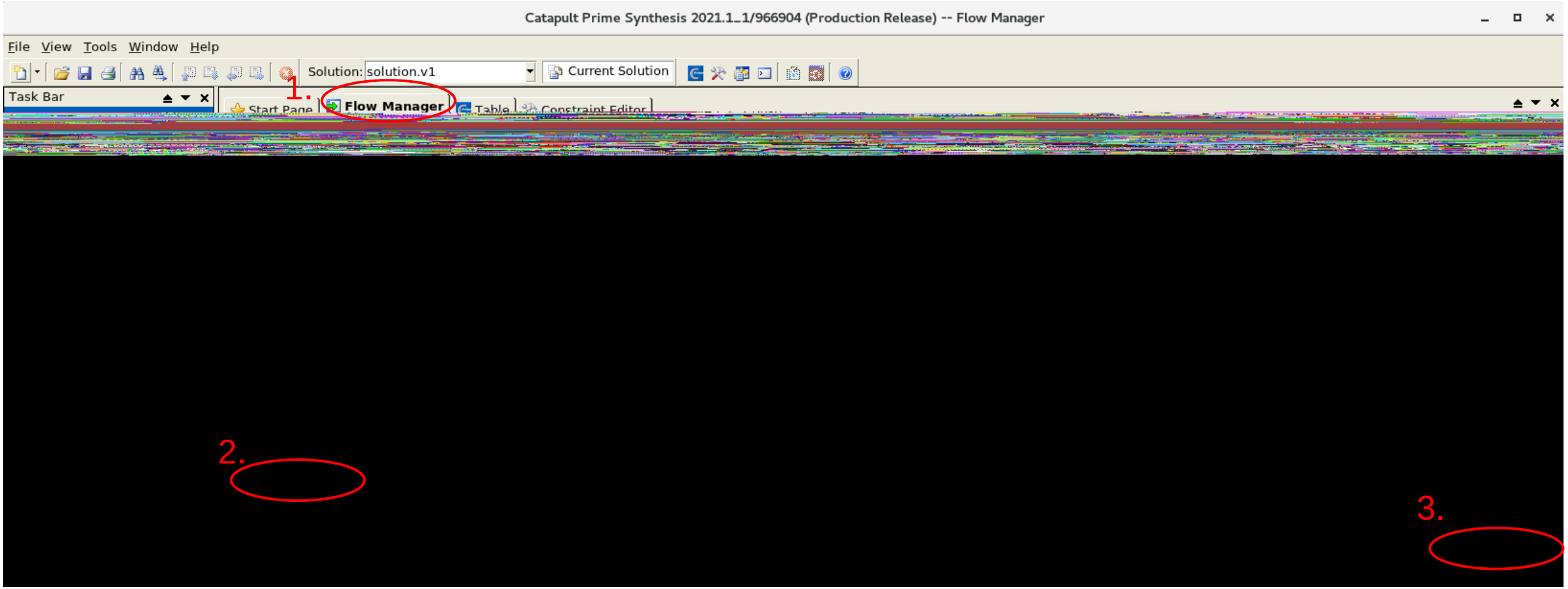


TB is not required for synthesis




SIEMENS

# Enable SCVerify

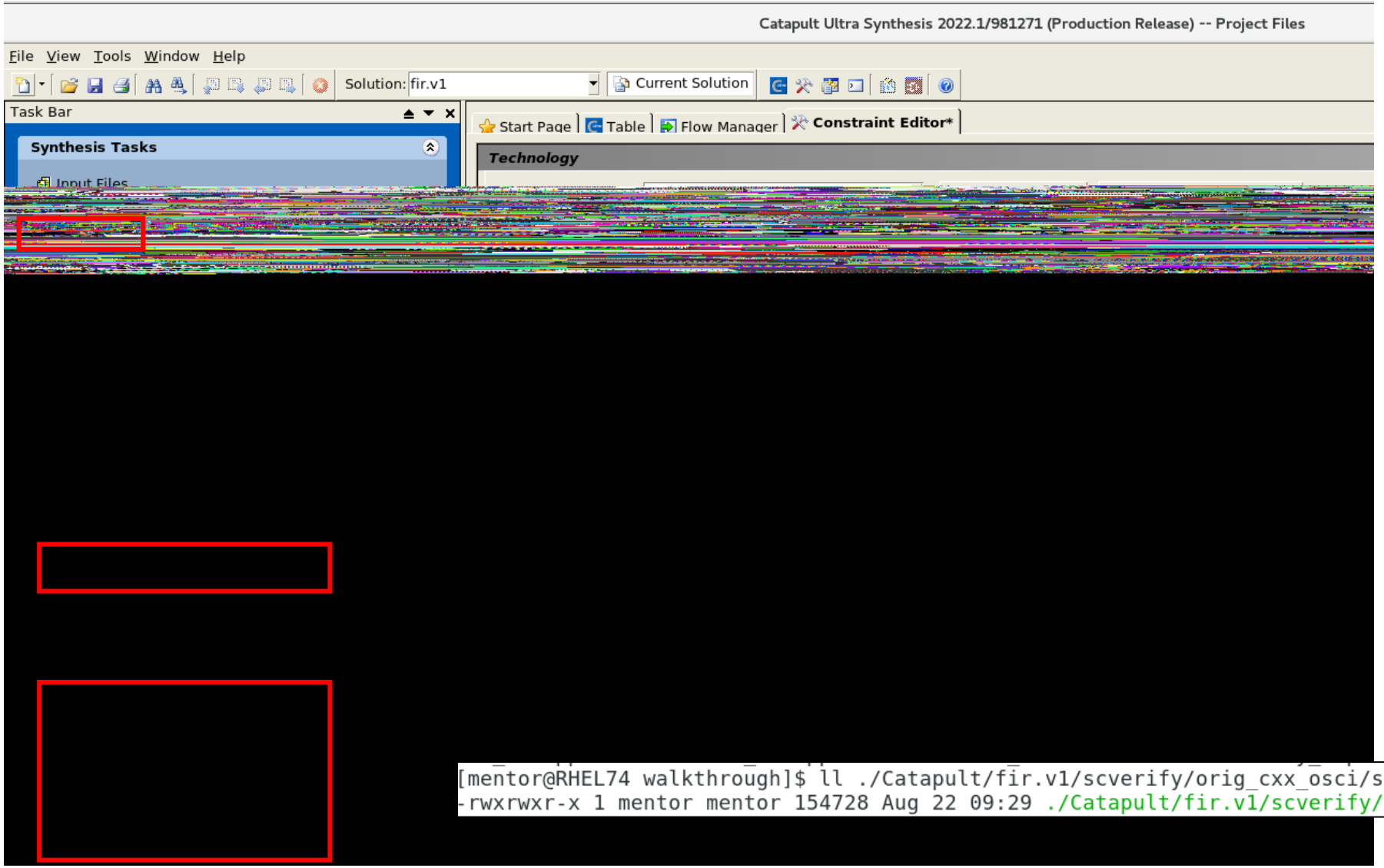






SIEMENS

# Pre-HLS (C++) Verification by gcc

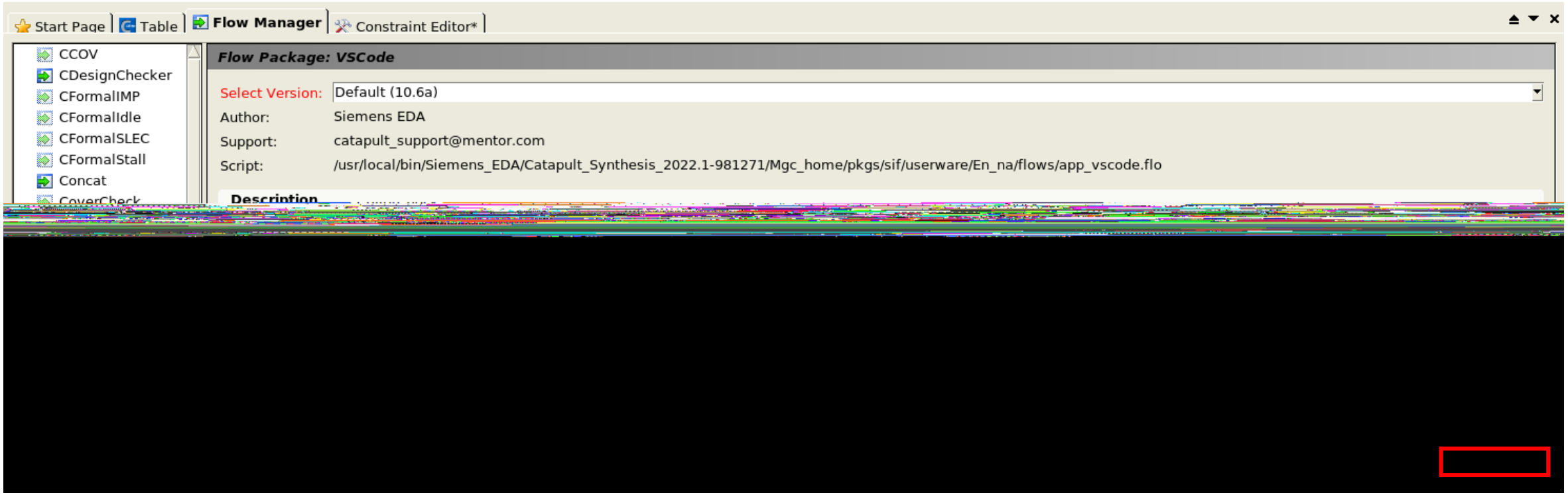






SIEMENS

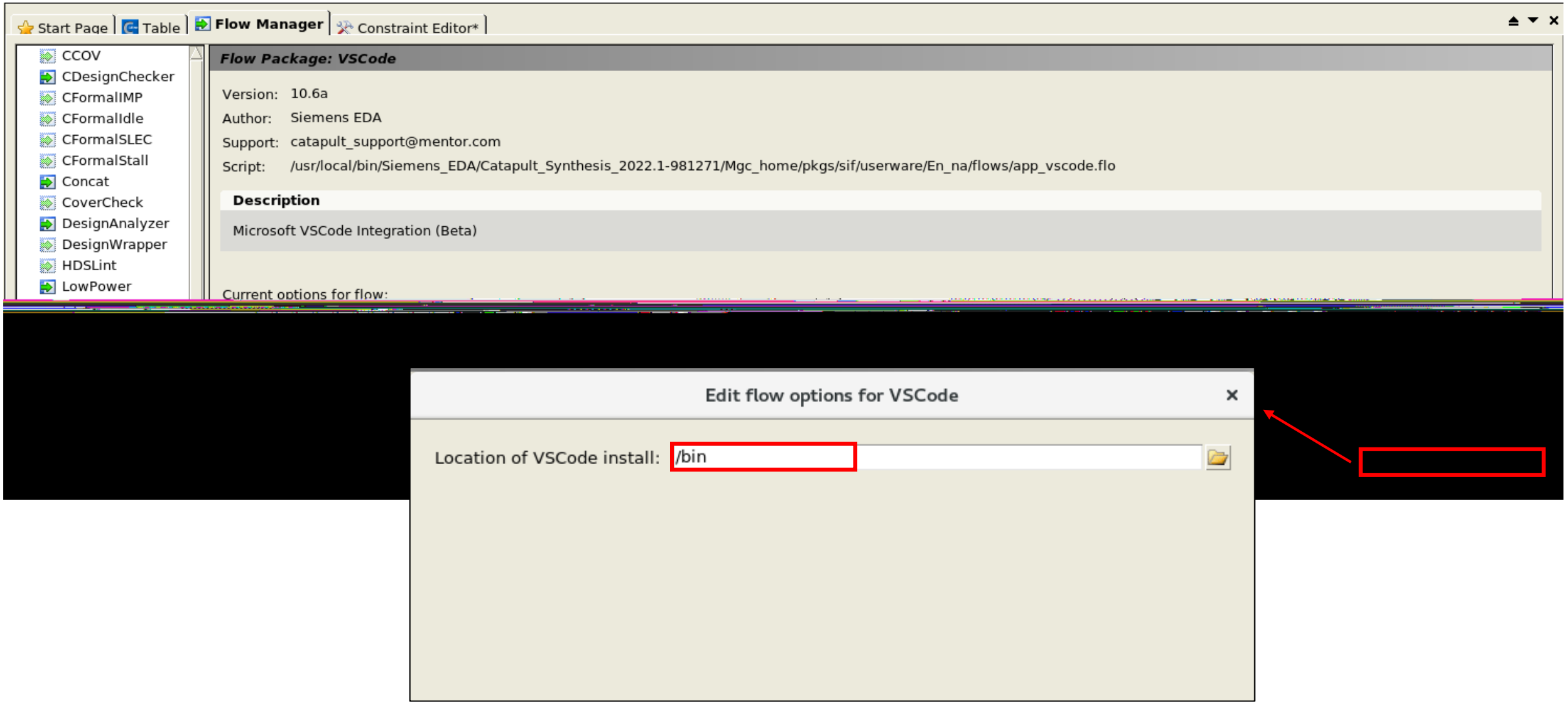
# Build VSCode Environment



SIEMENS

# Build VSCode Environment (Cont.)

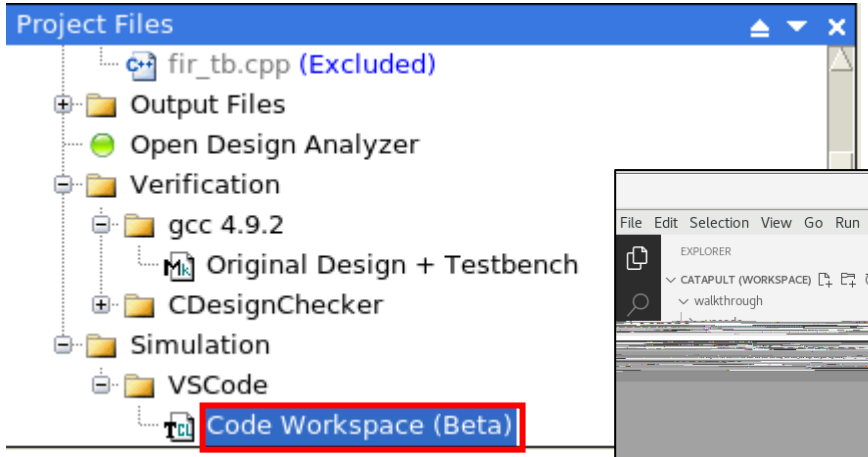


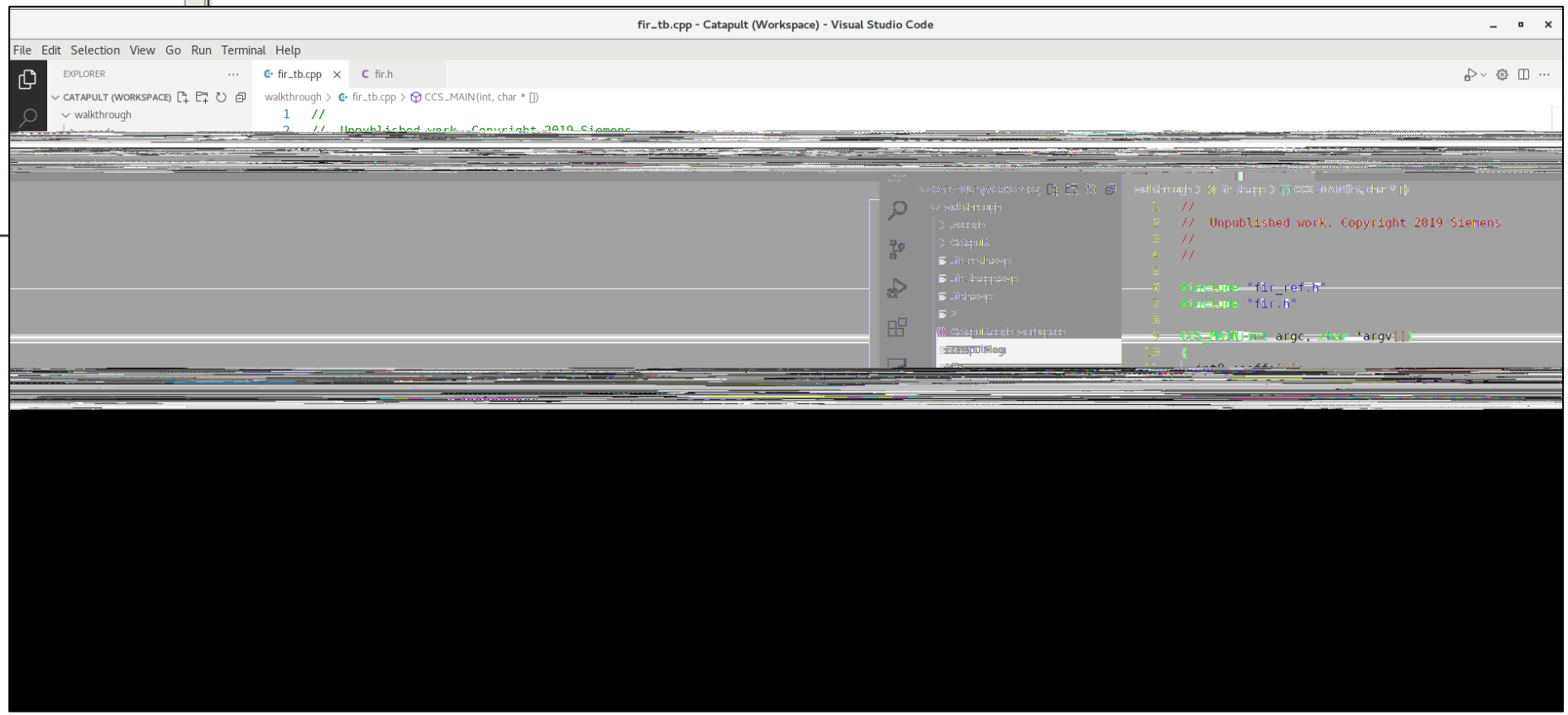




SIEMENS

# Build VSCode Environment (Cont.)

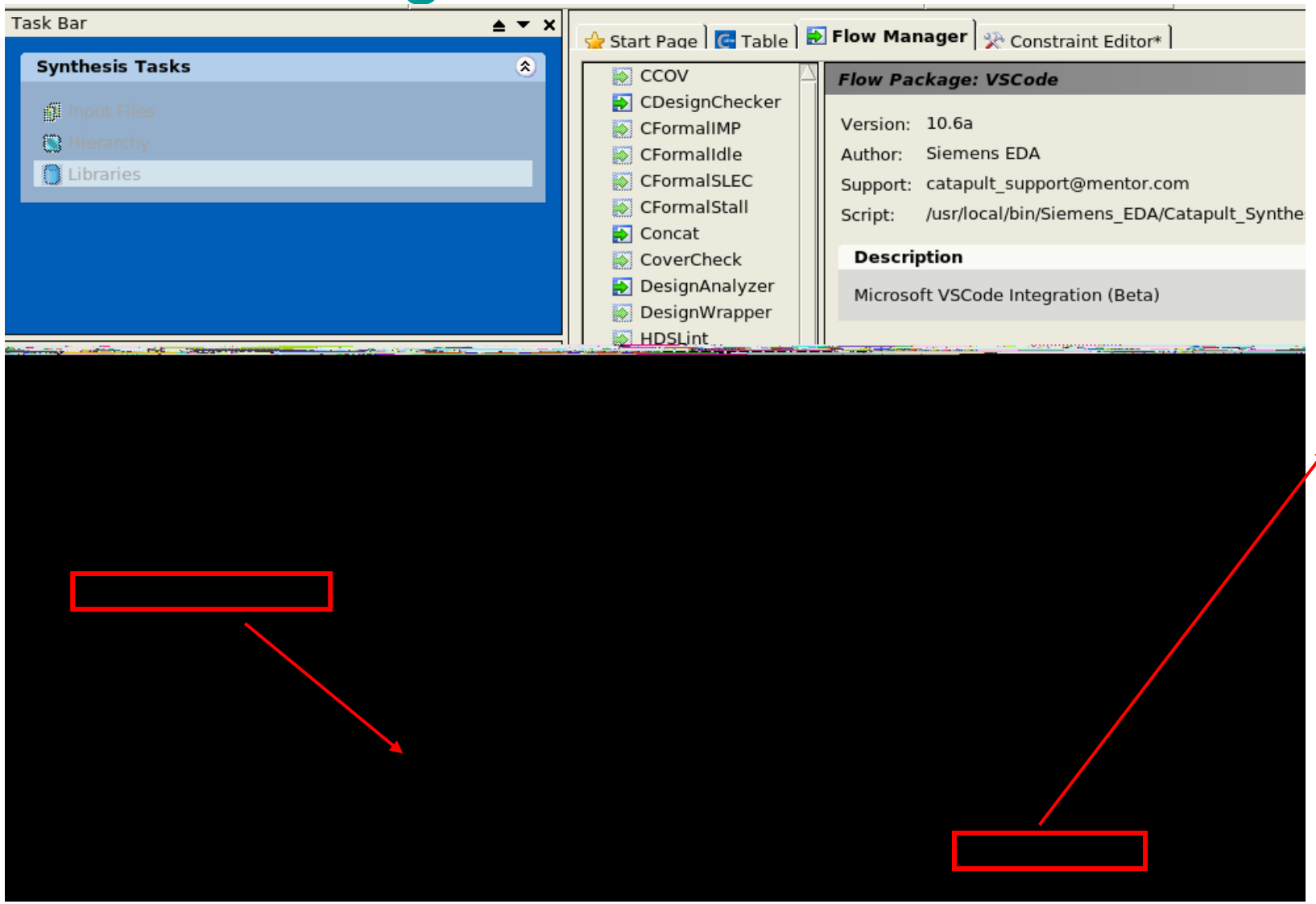


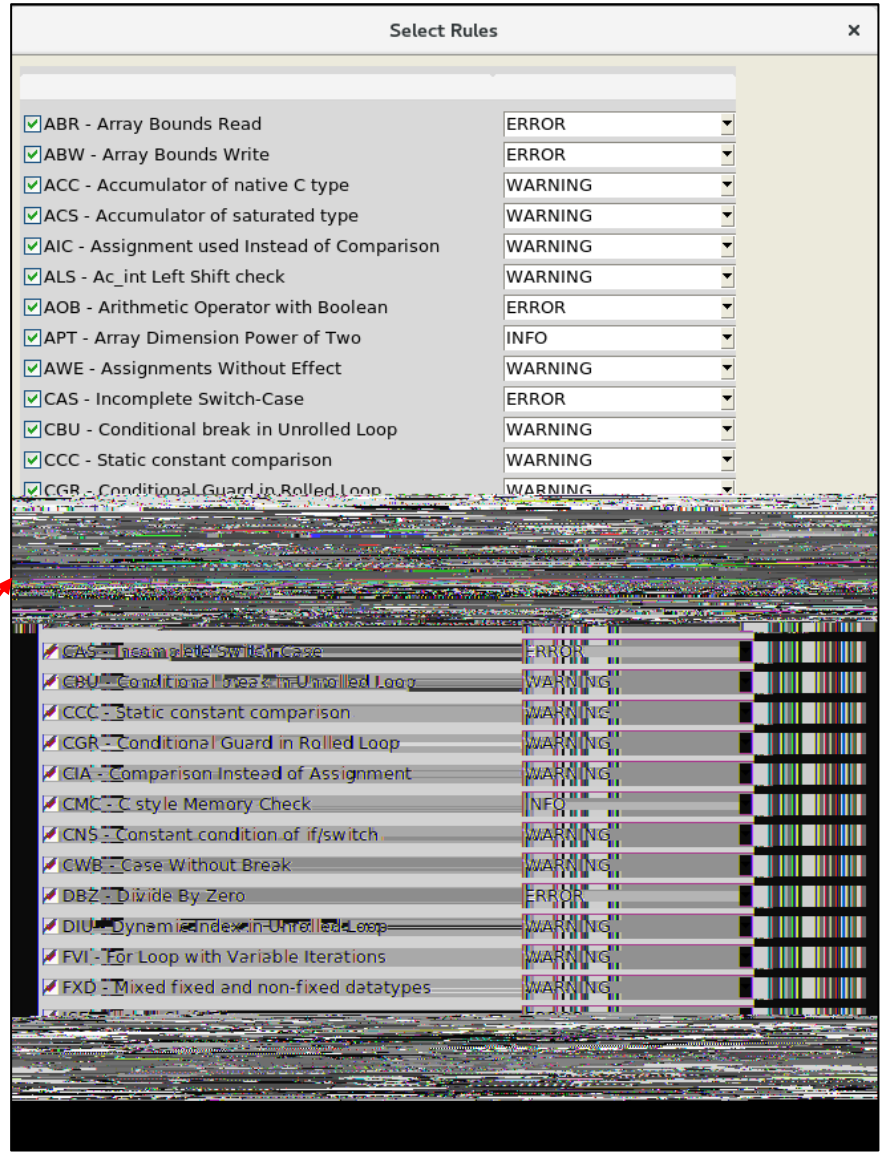



SIEMENS

# Run CDesignChecker

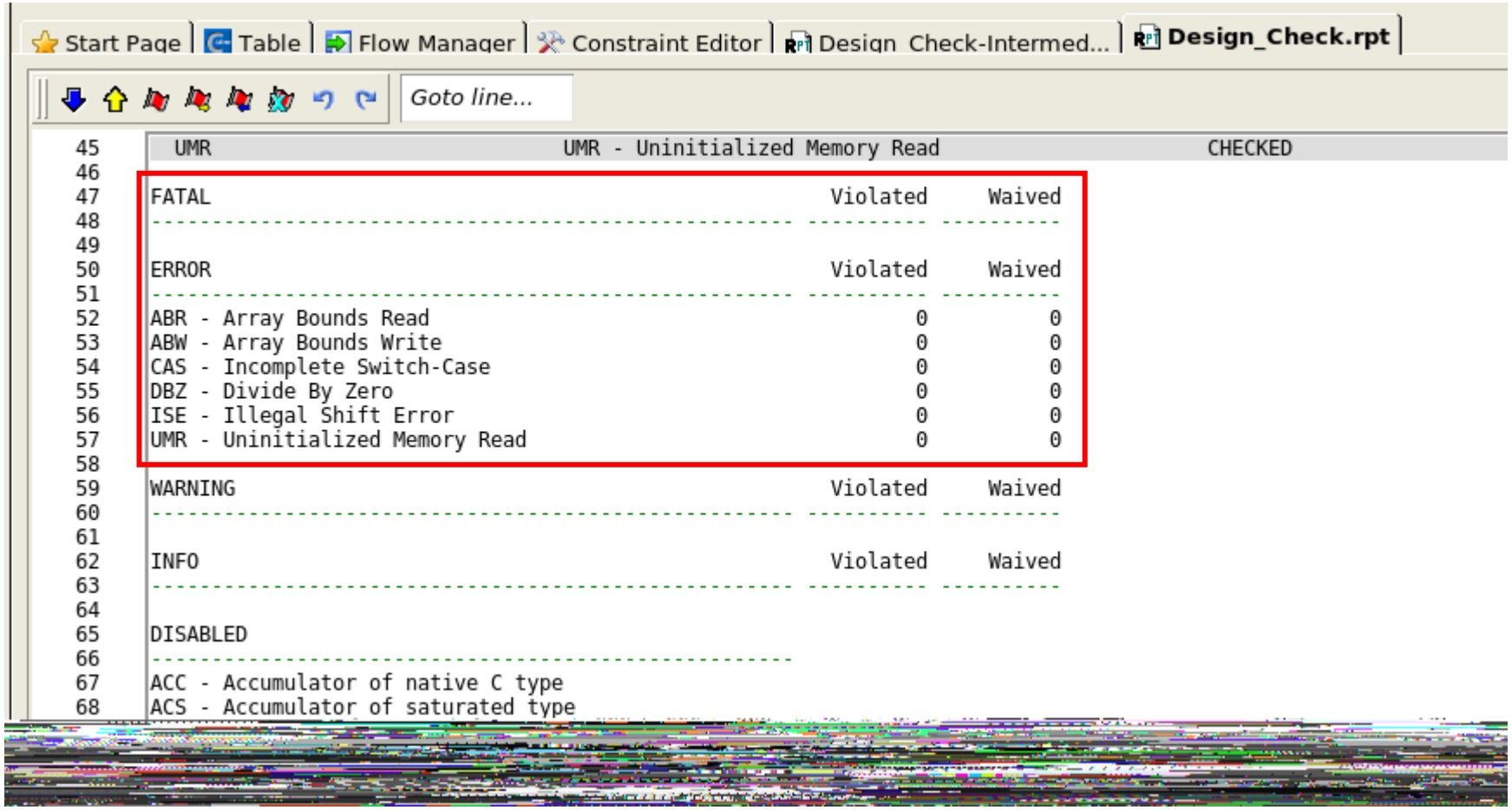


SIEMENS

# Run CDesignChecker (Cont.)

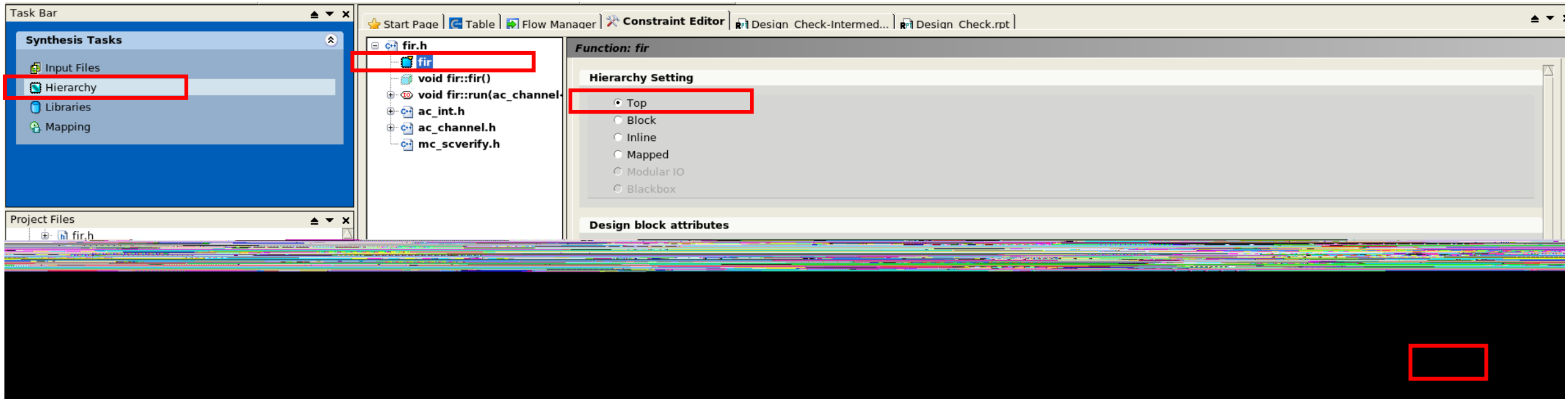






SIEMENS

# HLS Synthesis – Set Top Design

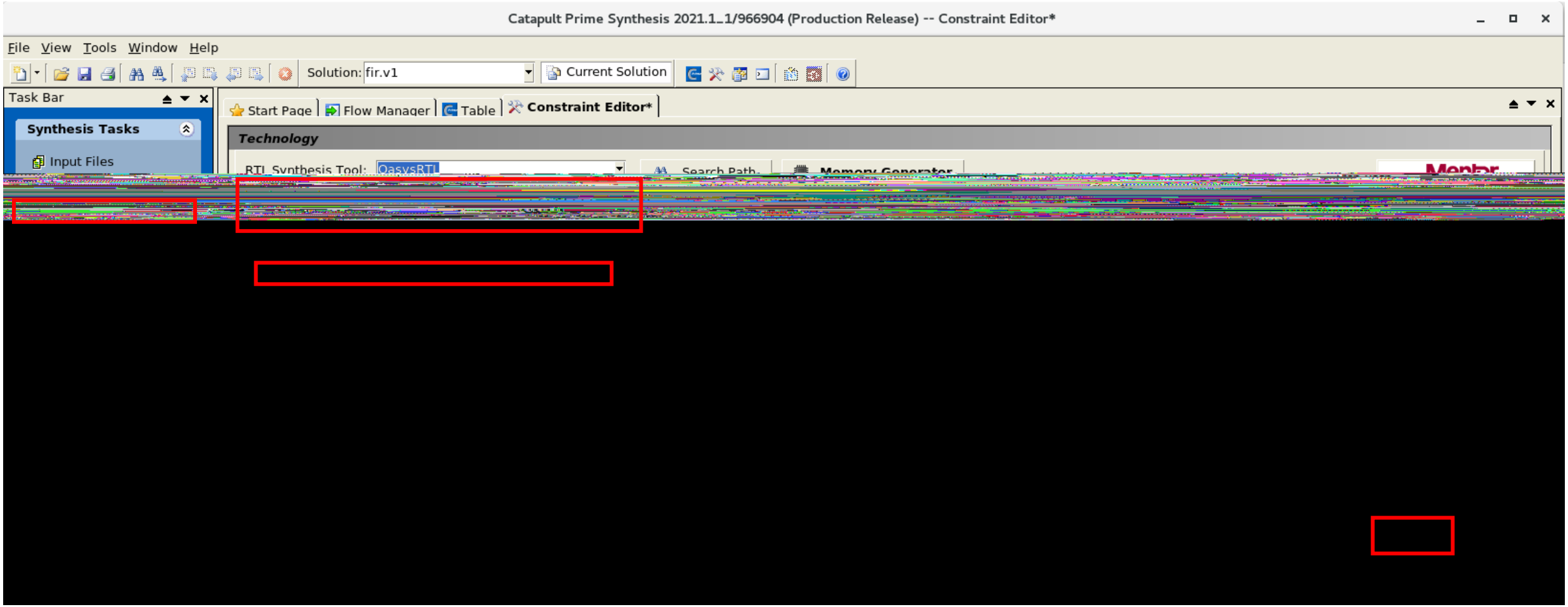






SIEMENS

# HLS Synthesis – Library Setup



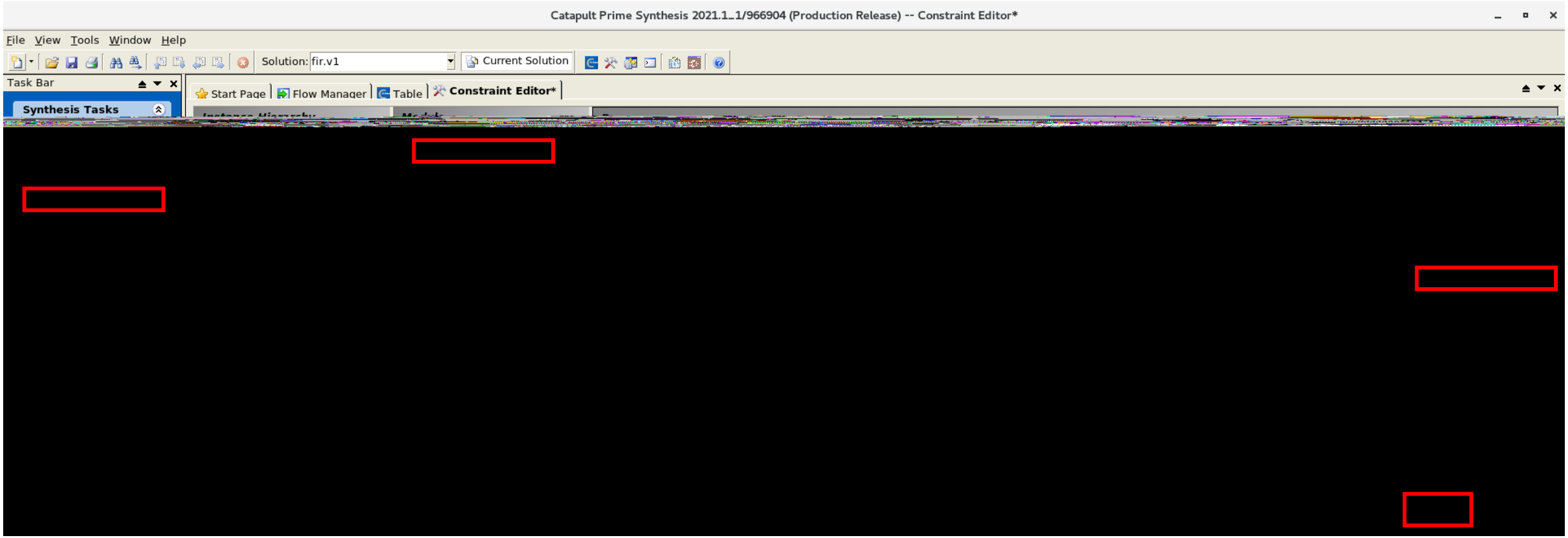




SIEMENS

# HLS Synthesis – Mapping

## Frequency



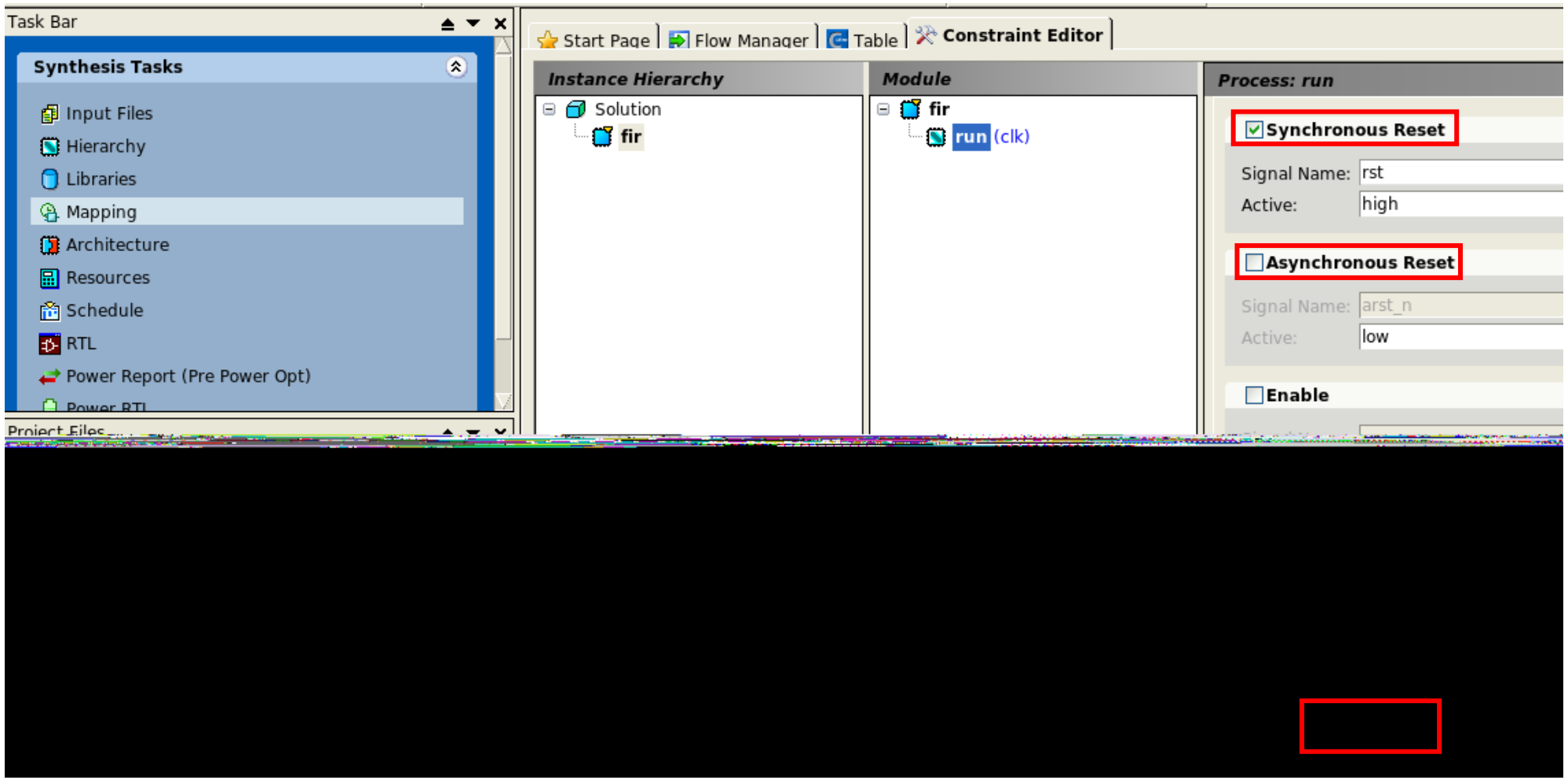




SIEMENS

# HLS Synthesis – Mapping

Sync / async reset

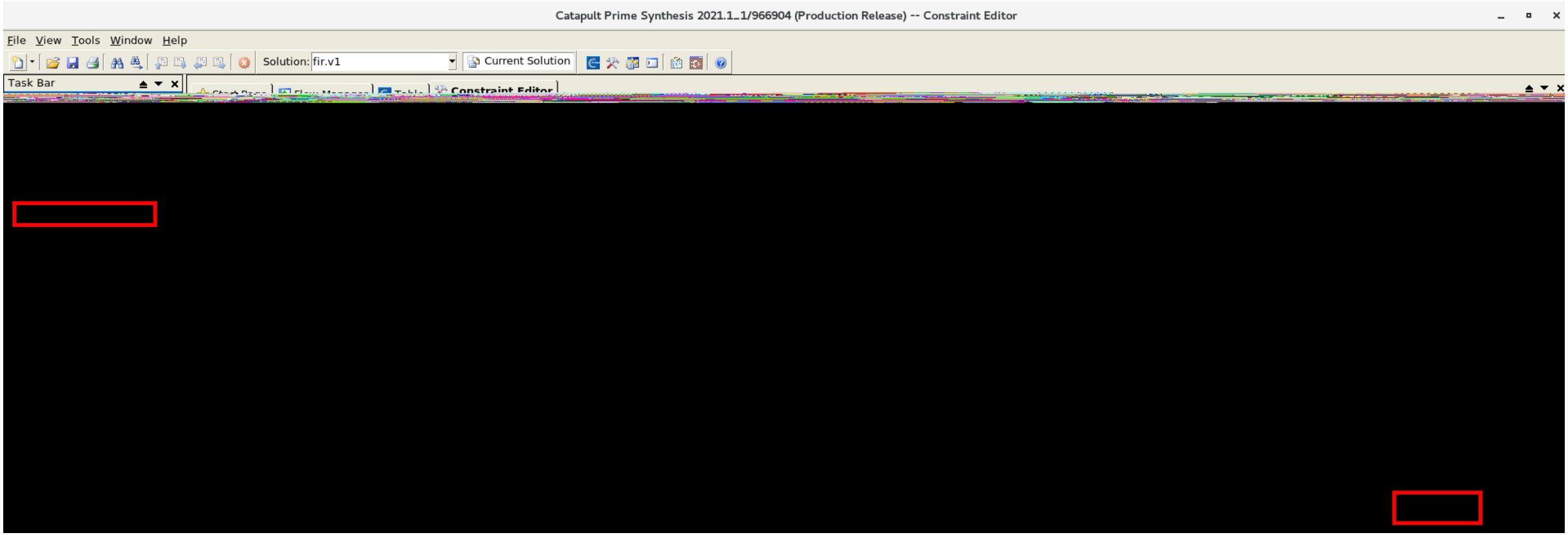




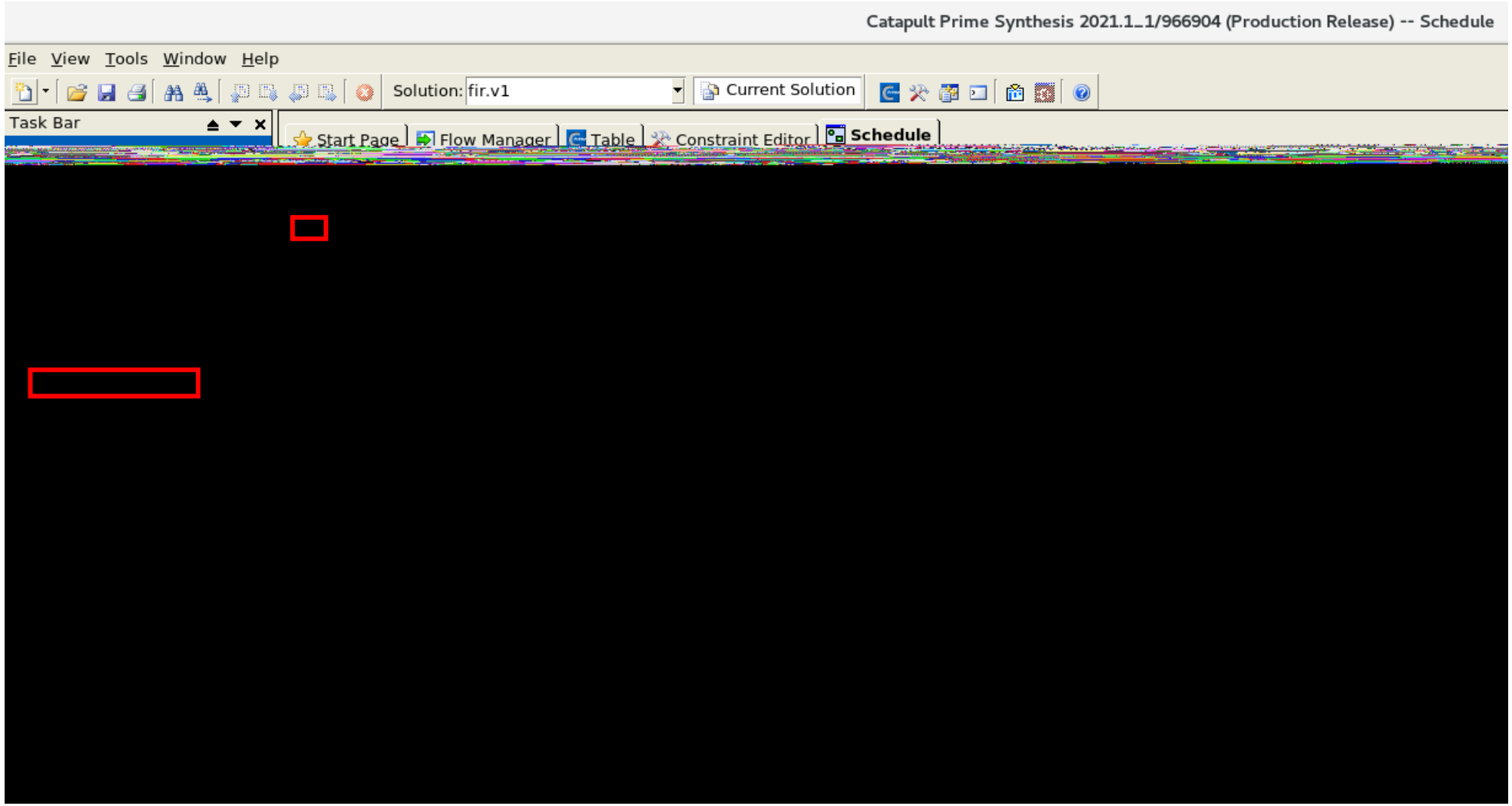


SIEMENS

# HLS Synthesis – Architecture



# HLS Synthesis – Scheduling

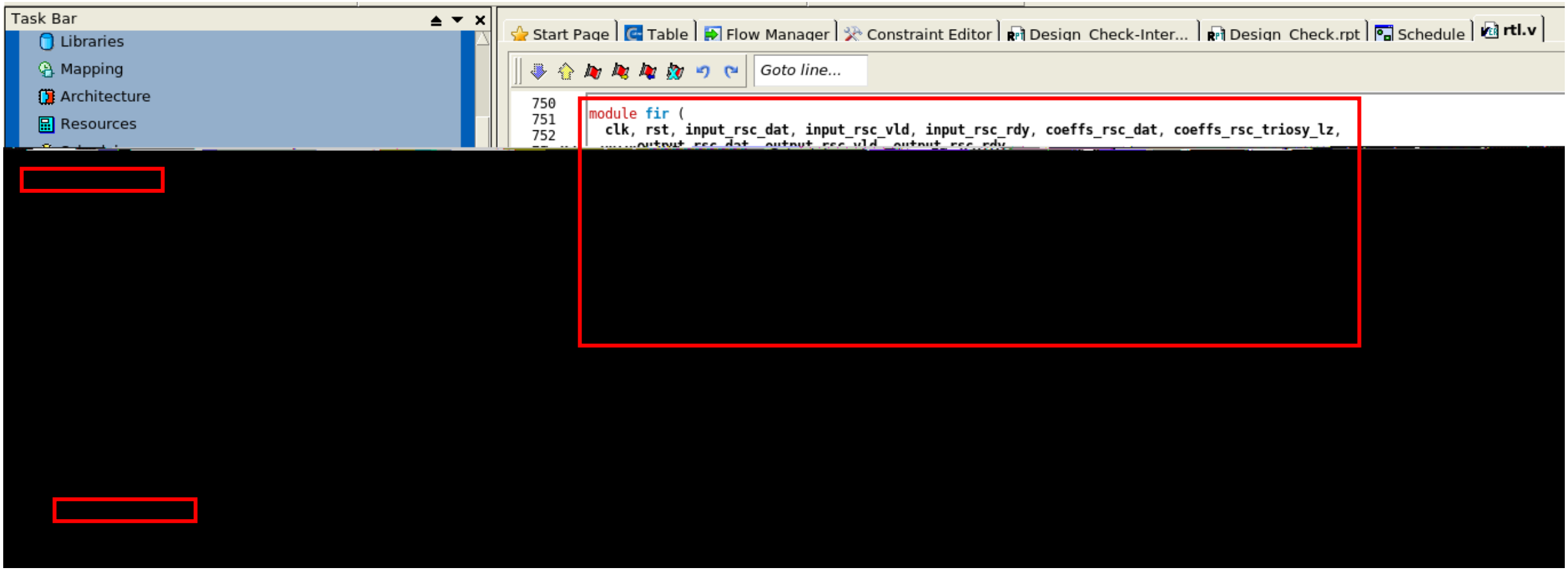






SIEMENS

# HLS Synthesis – RTL

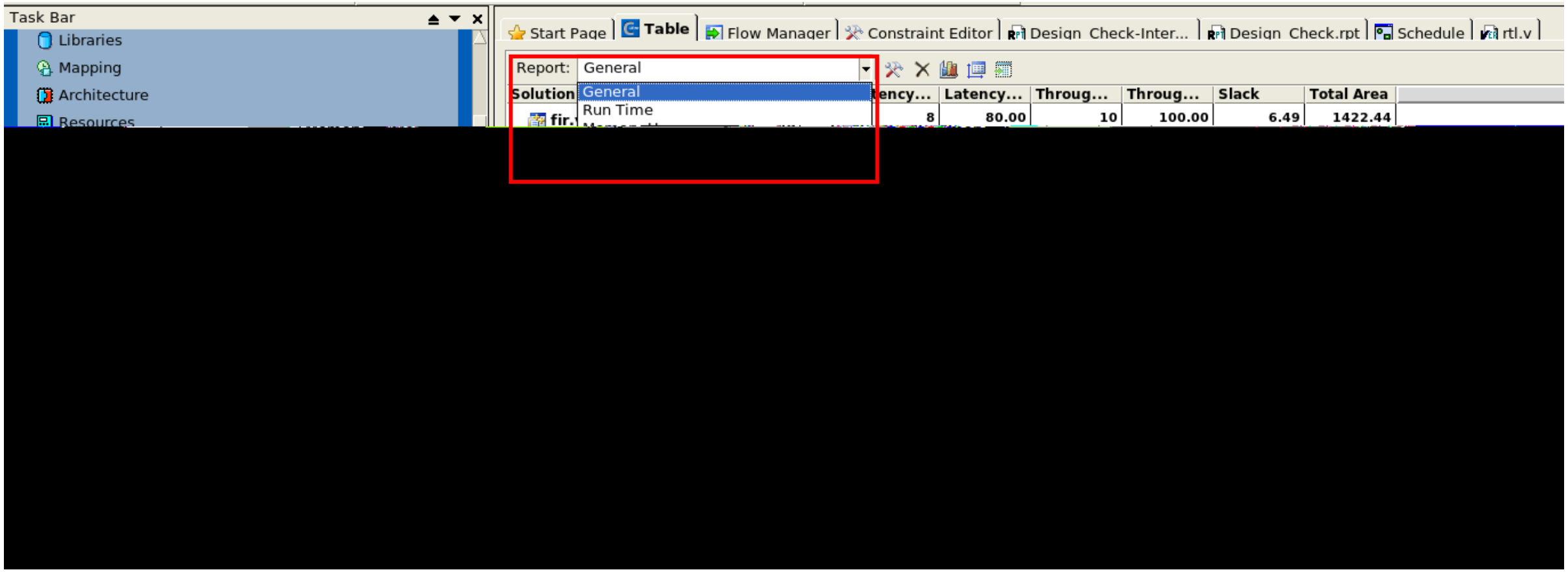






SIEMENS

# HLS Synthesis – Report

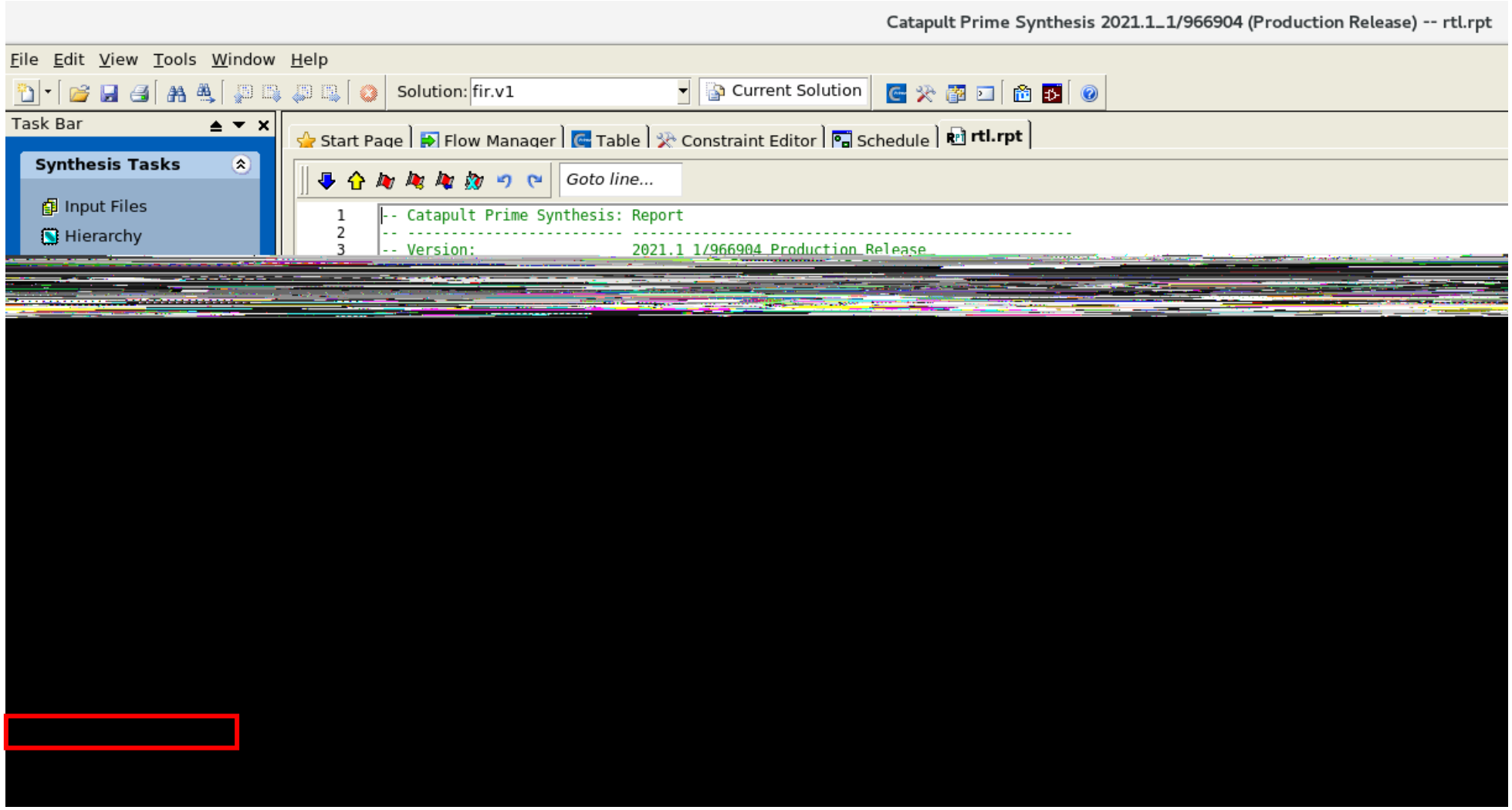






SIEMENS

# HLS Synthesis – Report (Cont.)



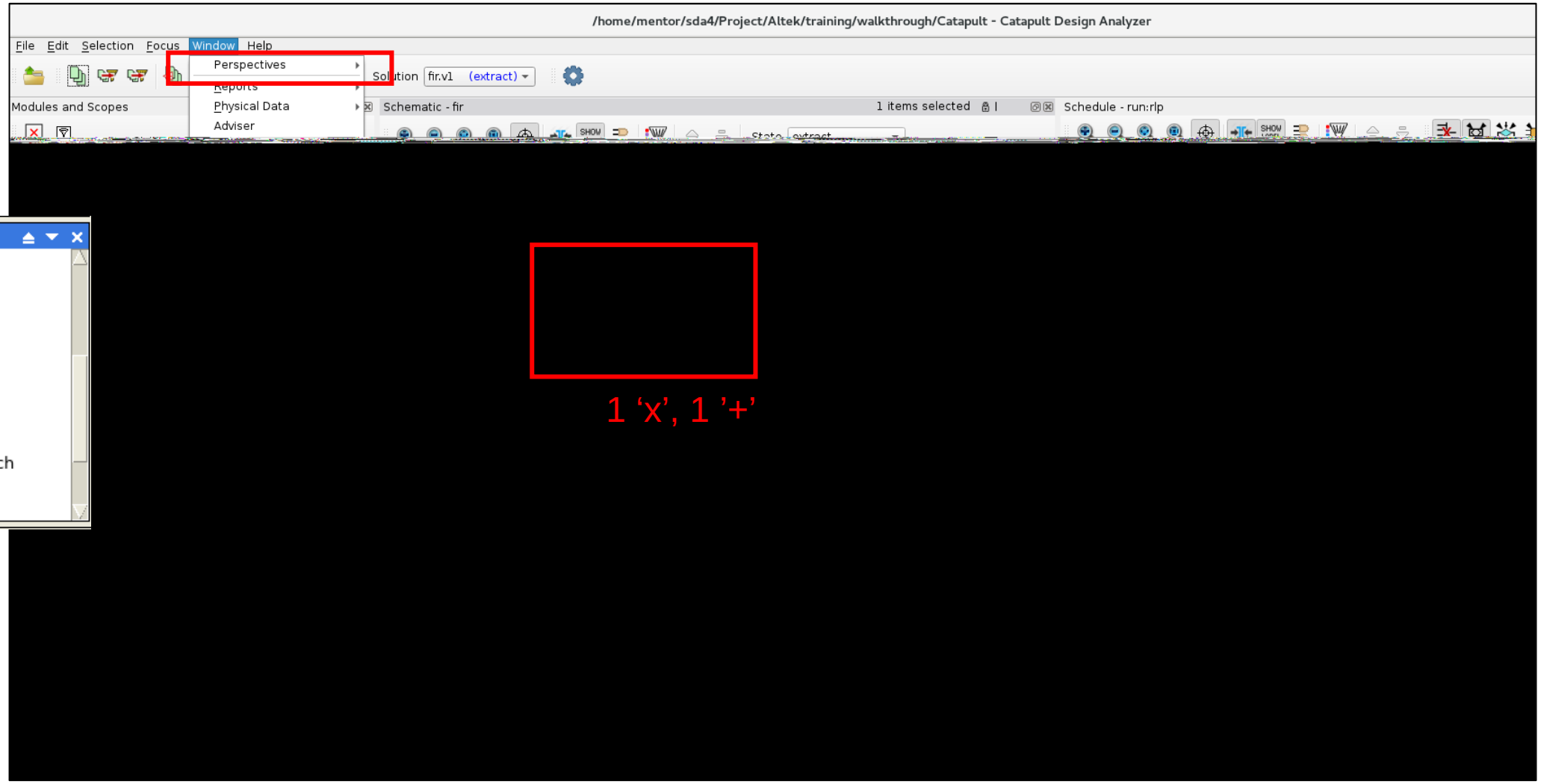


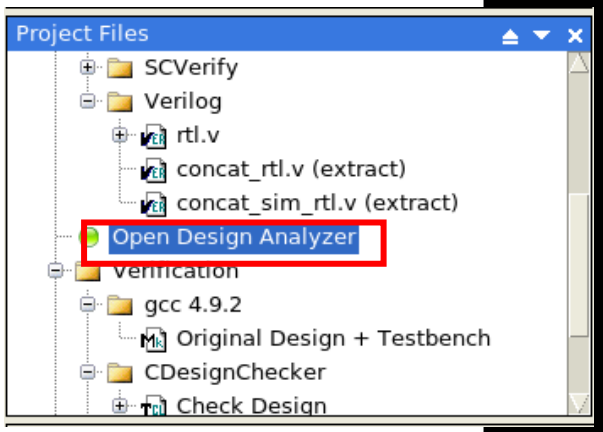


SIEMENS

# Design Analyzer

## Analyze the design in various perspectives



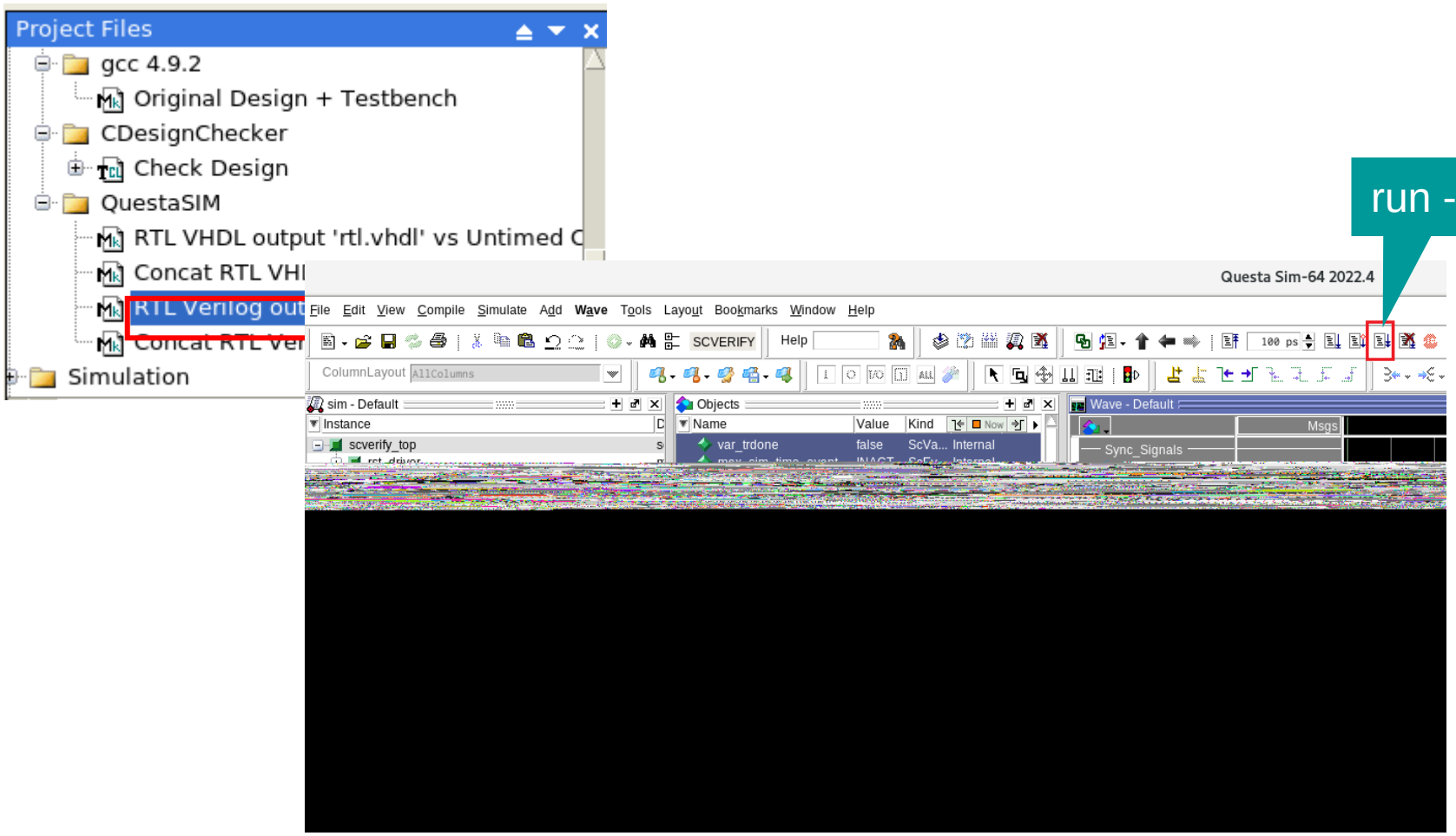


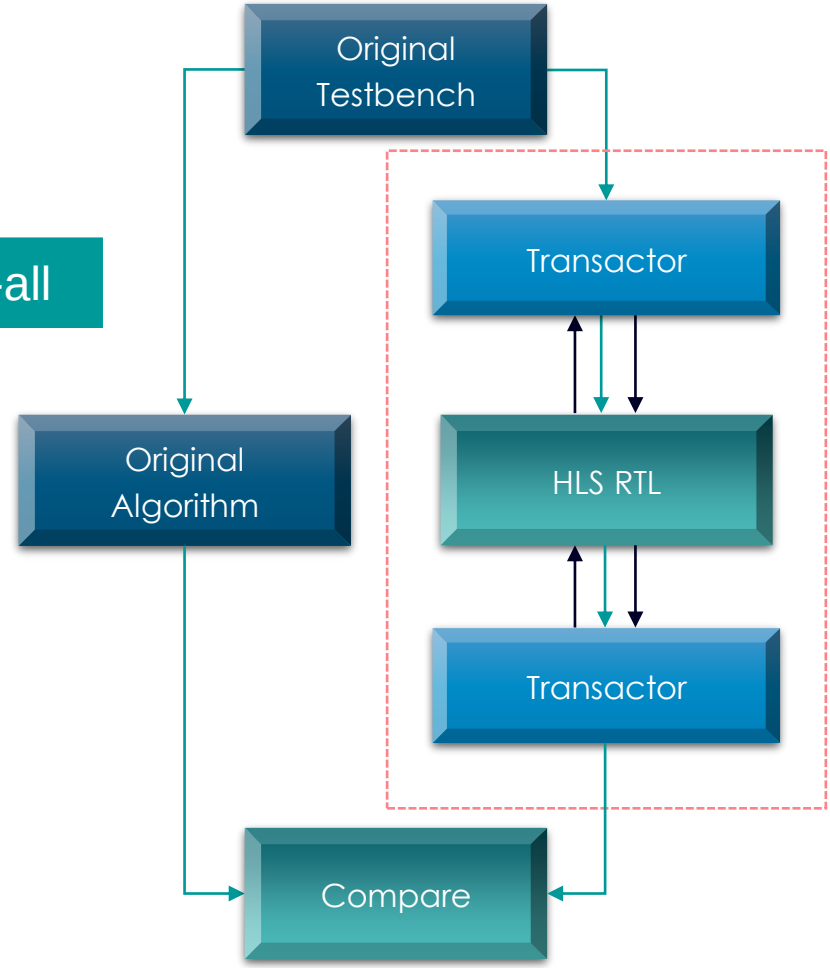


SIEMENS

# RTL Verification

Automatically compare the outputs of HLC C and RTL





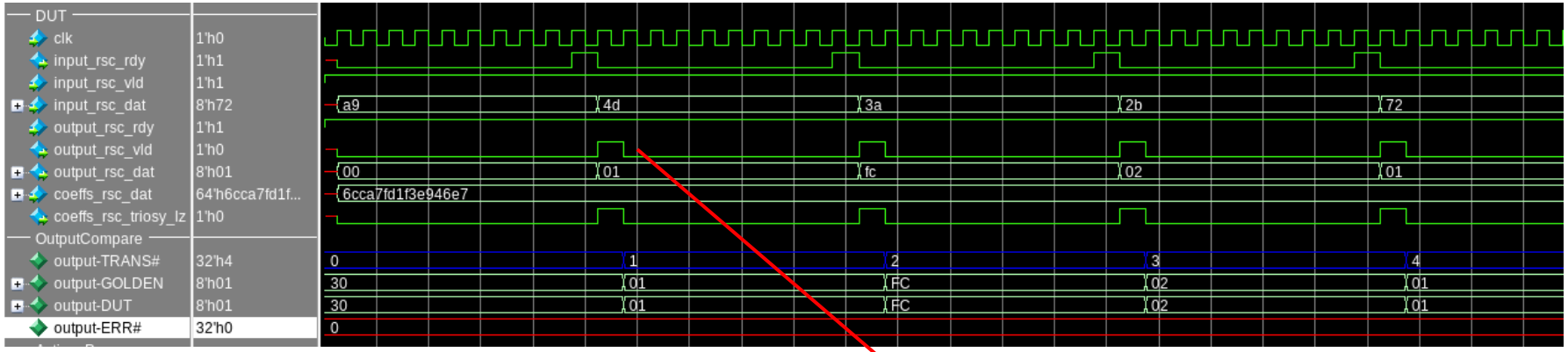


SIEMENS

# 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>
```




Throughput = 10



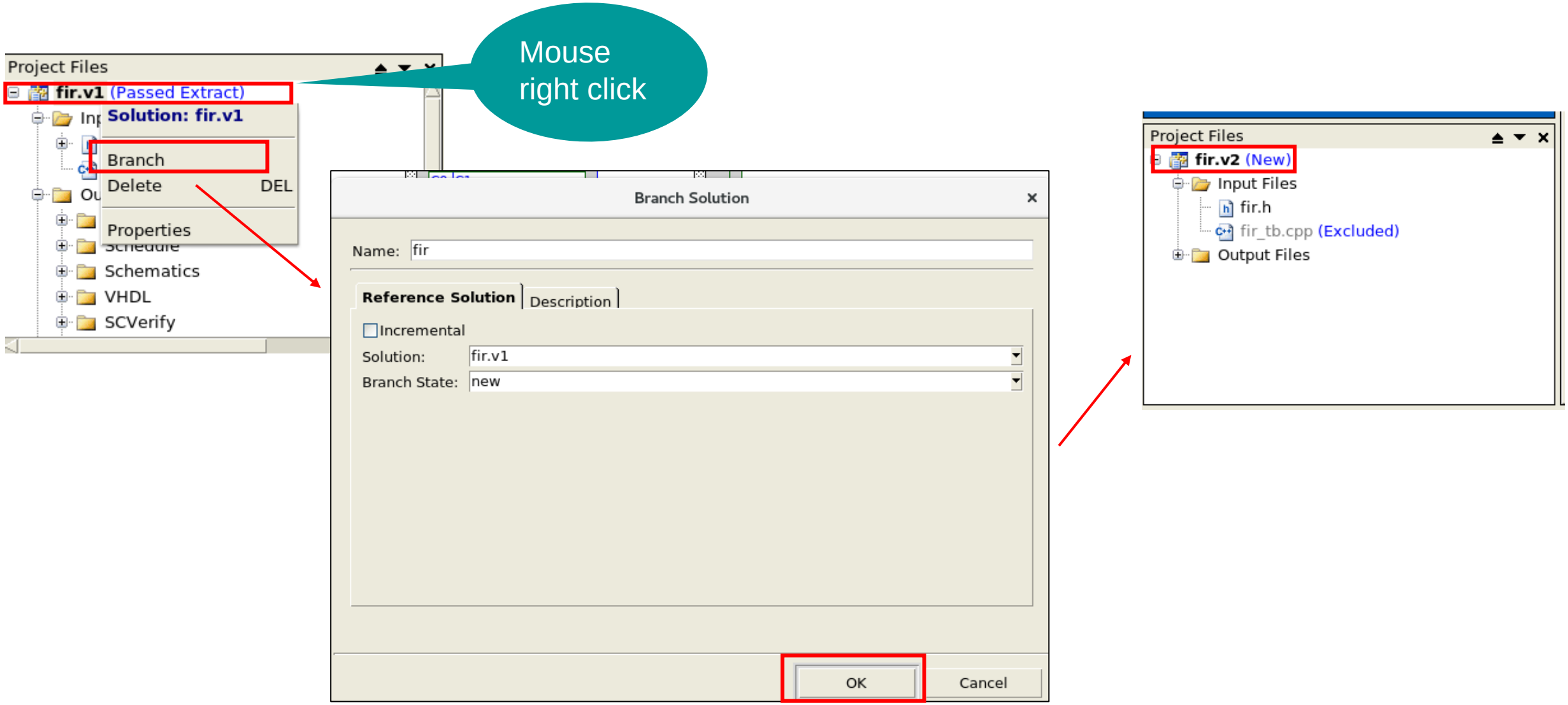

SIEMENS

# Loop Unrolling and Pipelining

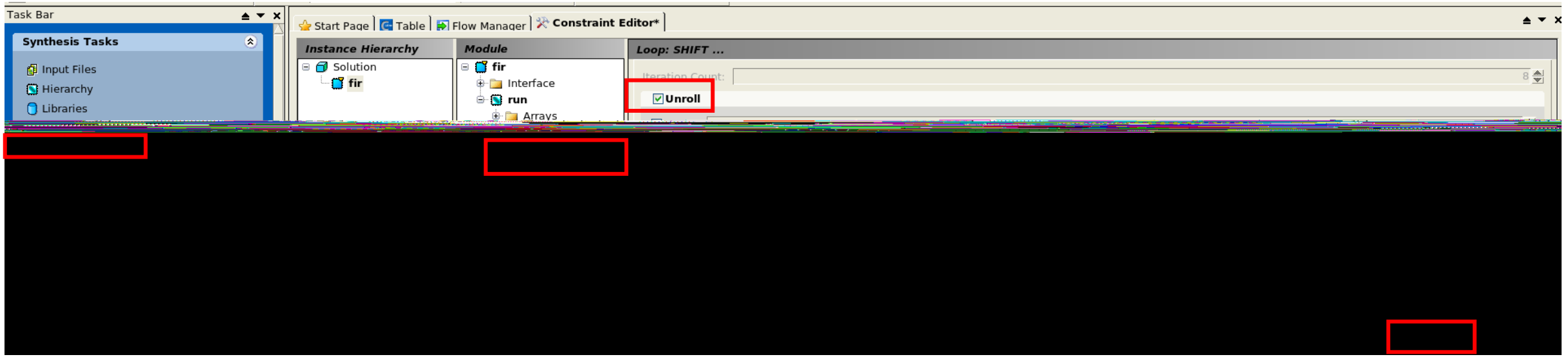


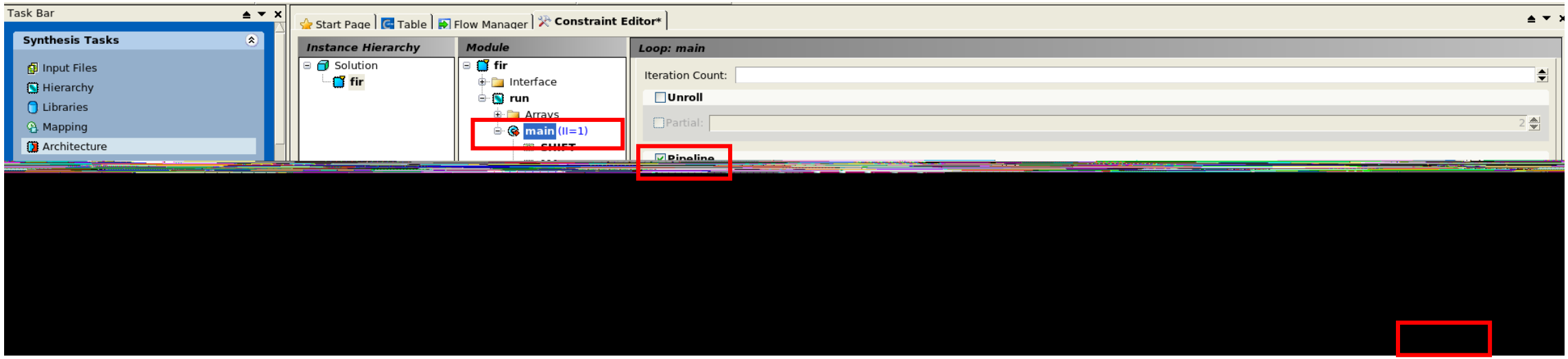

SIEMENS

# Branch a New Solution

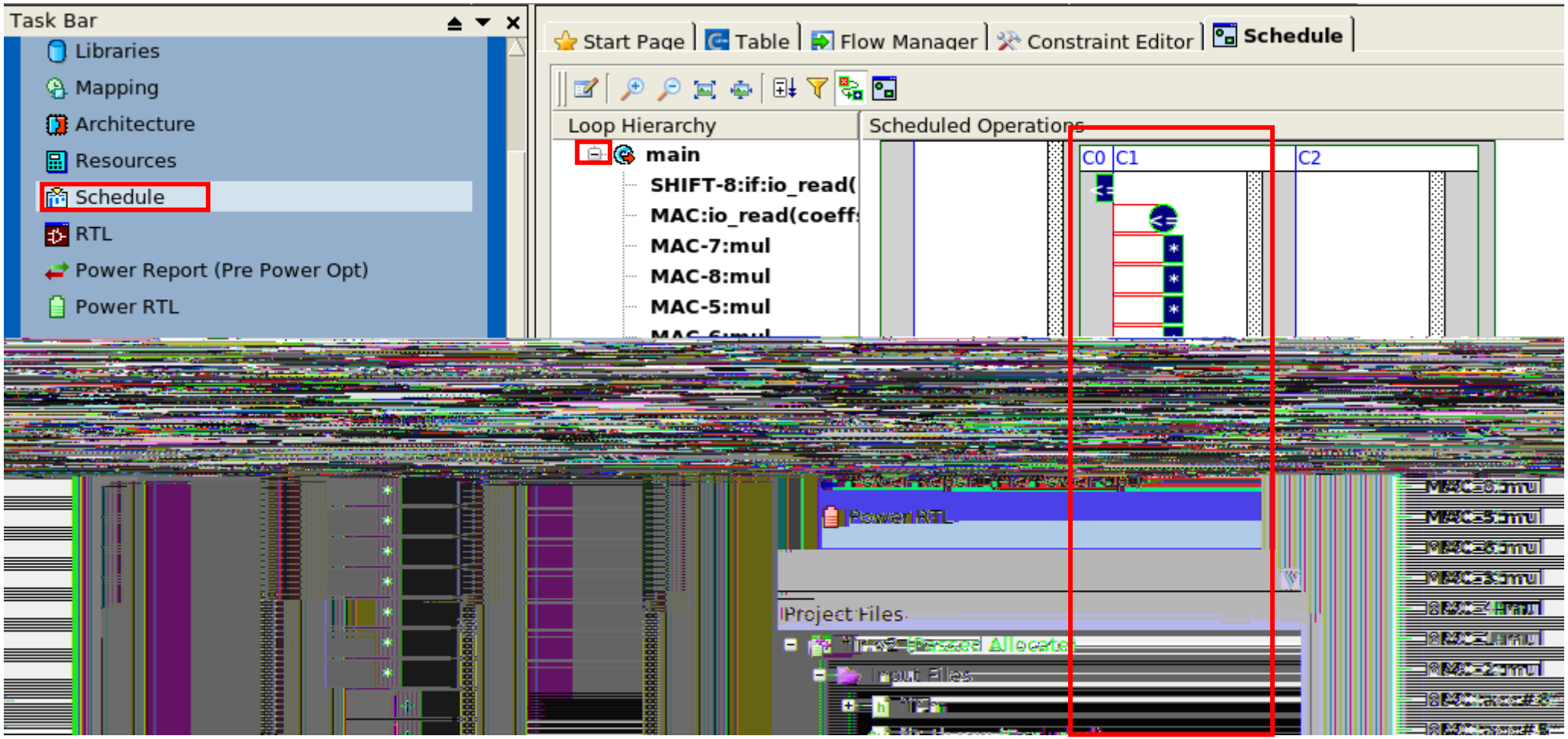


# HLS Synthesis – Architecture



# HLS Synthesis – Scheduling

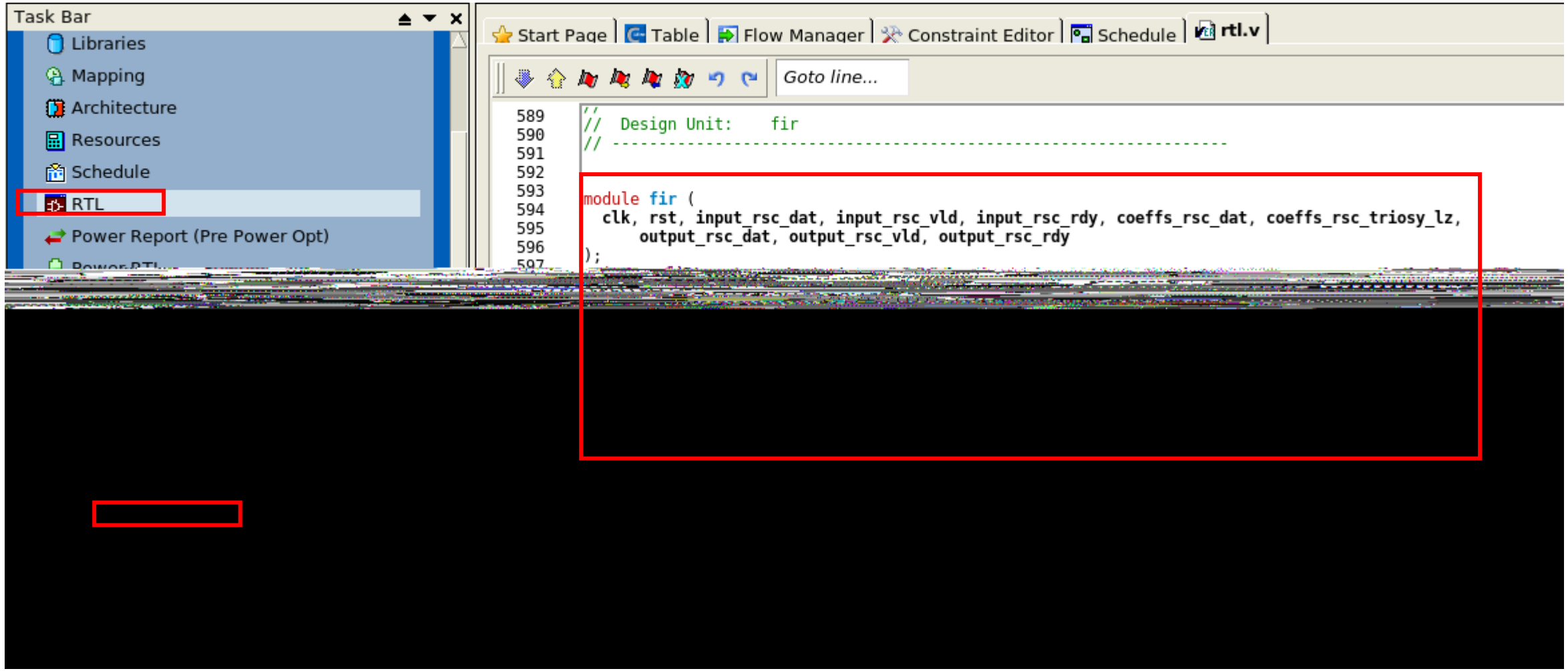






SIEMENS

# HLS Synthesis – RTL

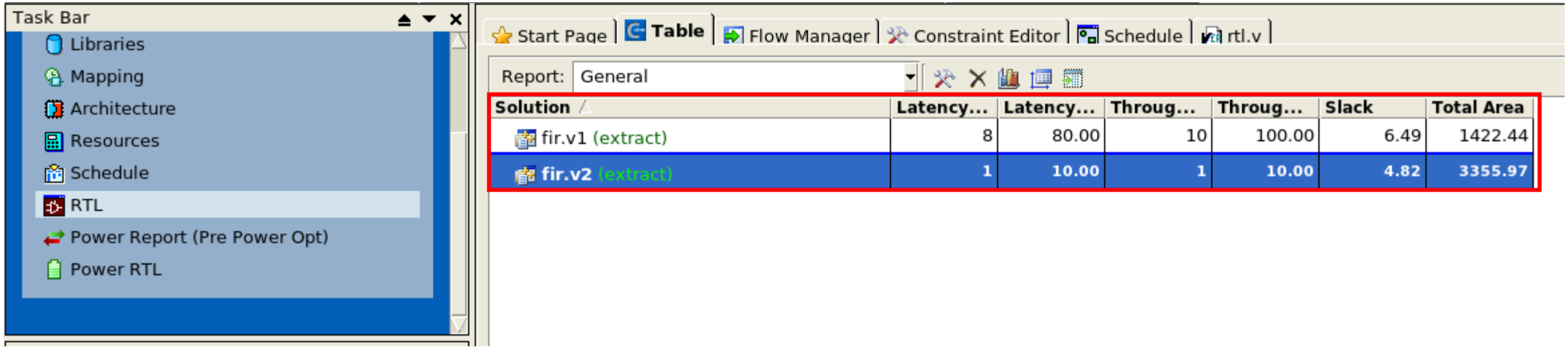






SIEMENS

# HLS Synthesis – Report



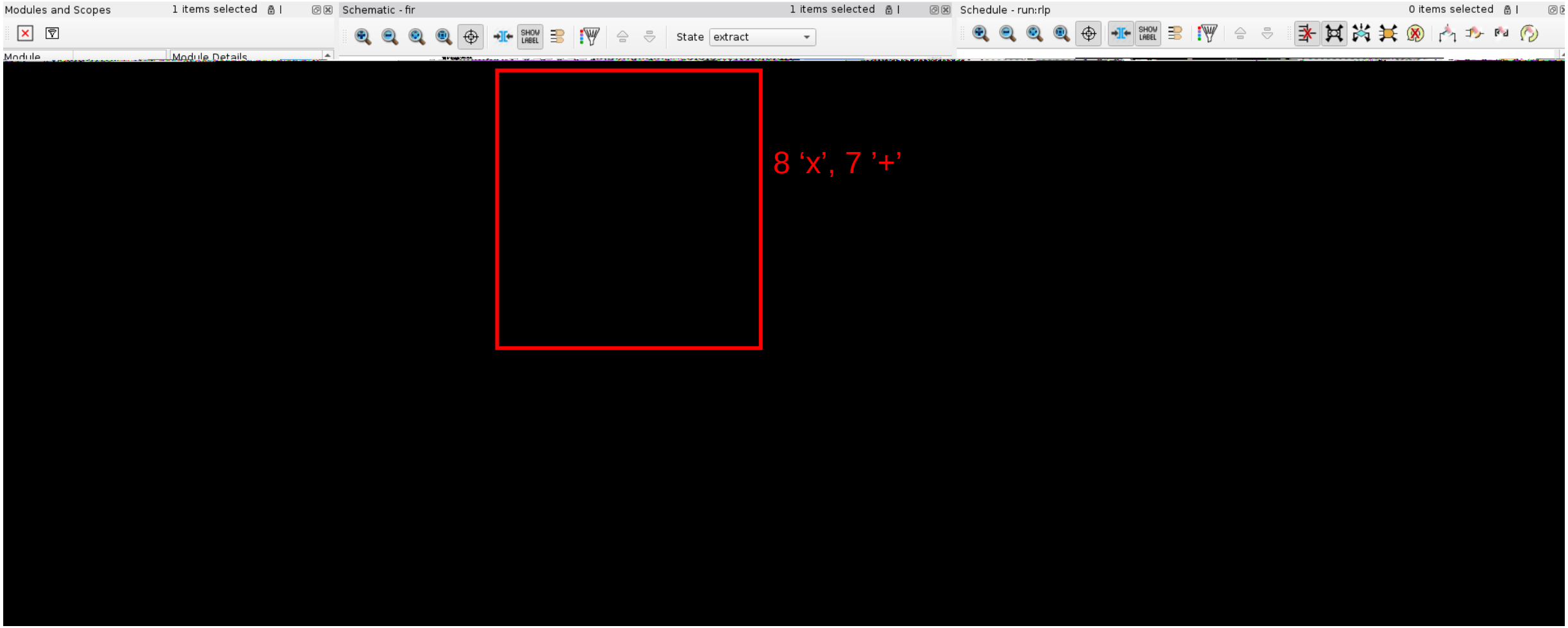

| Solution | Latency... | Latency... | Throug... | Throug... | Slack | Total Area |
|---|---|---|---|---|---|---|
| fir.v1 (extract) | 8 | 80.00 | 10 | 100.00 | 6.49 | 1422.44 |
| **fir.v2 (extract)** | **1** | **10.00** | **1** | **10.00** | **4.82** | **3355.97** |




SIEMENS

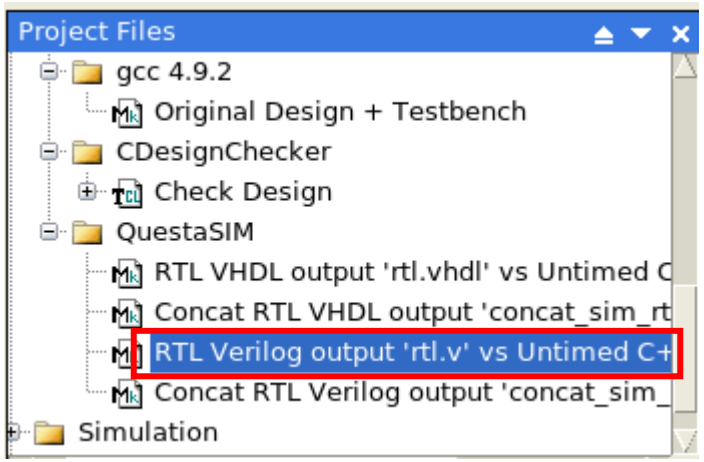
# Design Analyzer

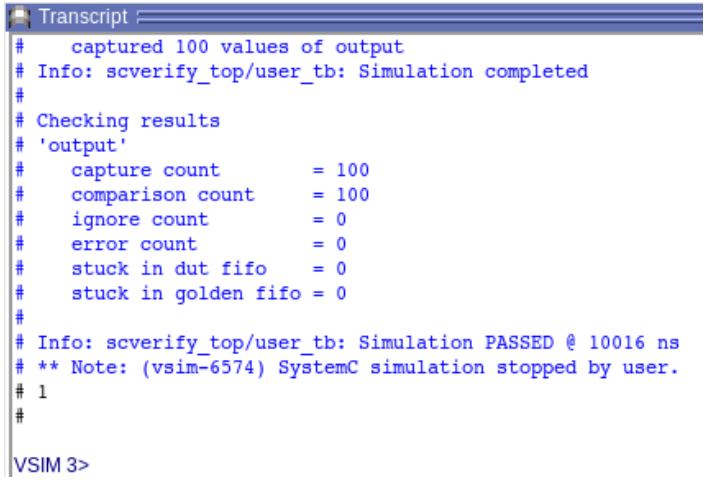
## Analyze the design in various perspectives

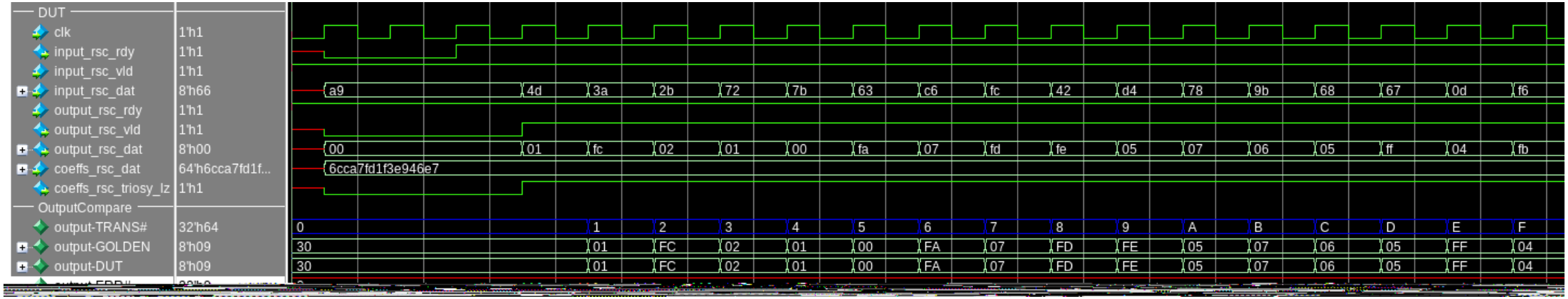


# RTL Verification

Automatically compare the outputs of HLC C and RTL





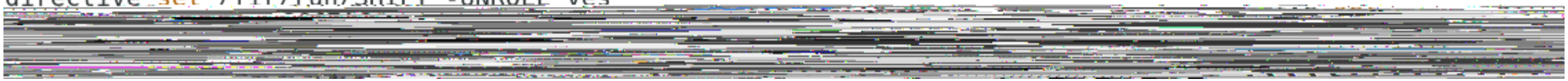


SIEMENS

# 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 -- -rtlsyntool DesignCompiler -vendor Nangate -technology 045nm
go libraries
directive set -CLOCKS {clk {-CLOCK_PERIOD 10.0 -CLOCK_EDGE rising -CLOCK_UNCERTAINTY 0.0 -CLOCK_HIGH_TI
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 &




SIEMENS

# 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);
}
```




SIEMENS

# 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);
}
```

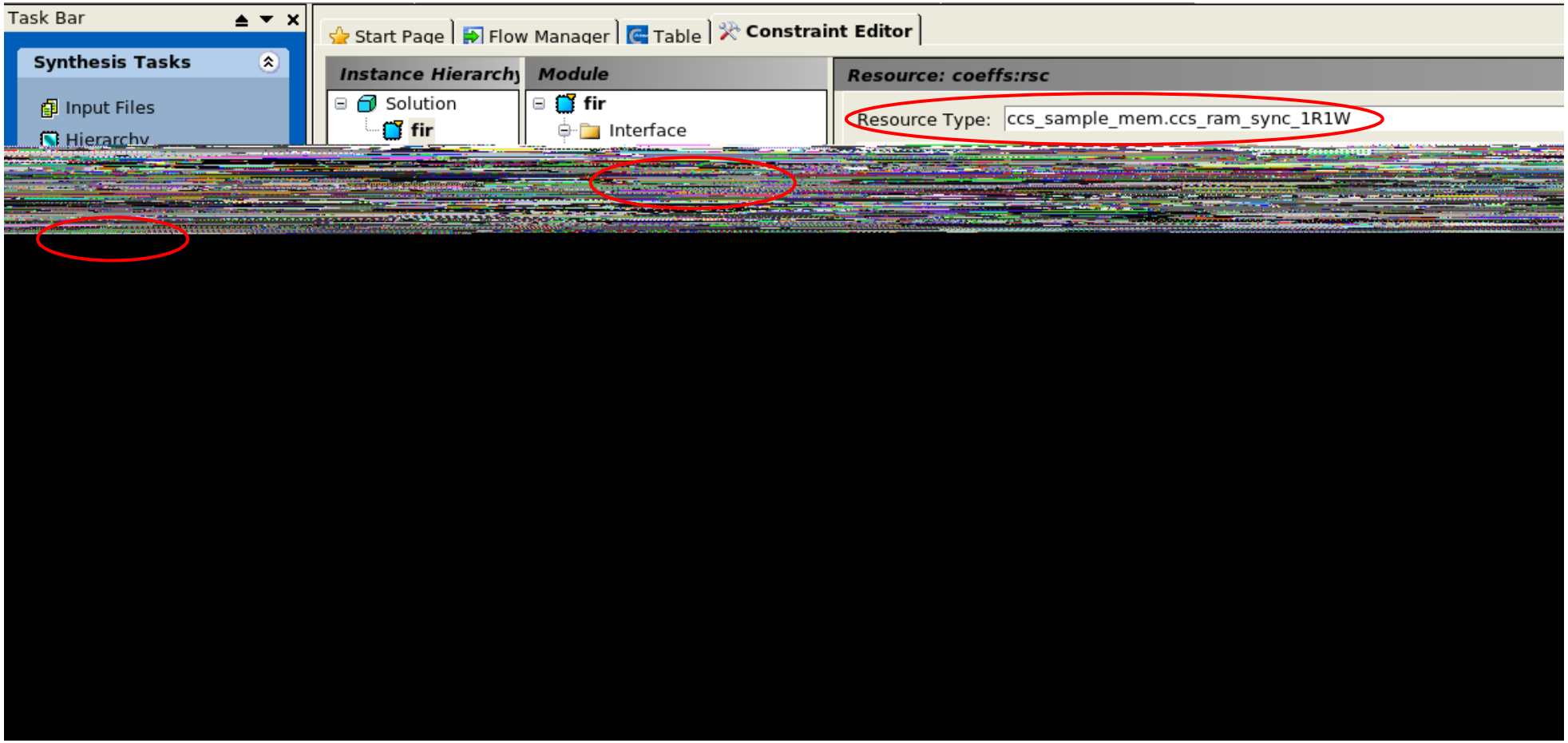



SIEMENS

# Start with running directives.tcl

catapult –f directives.tcl

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



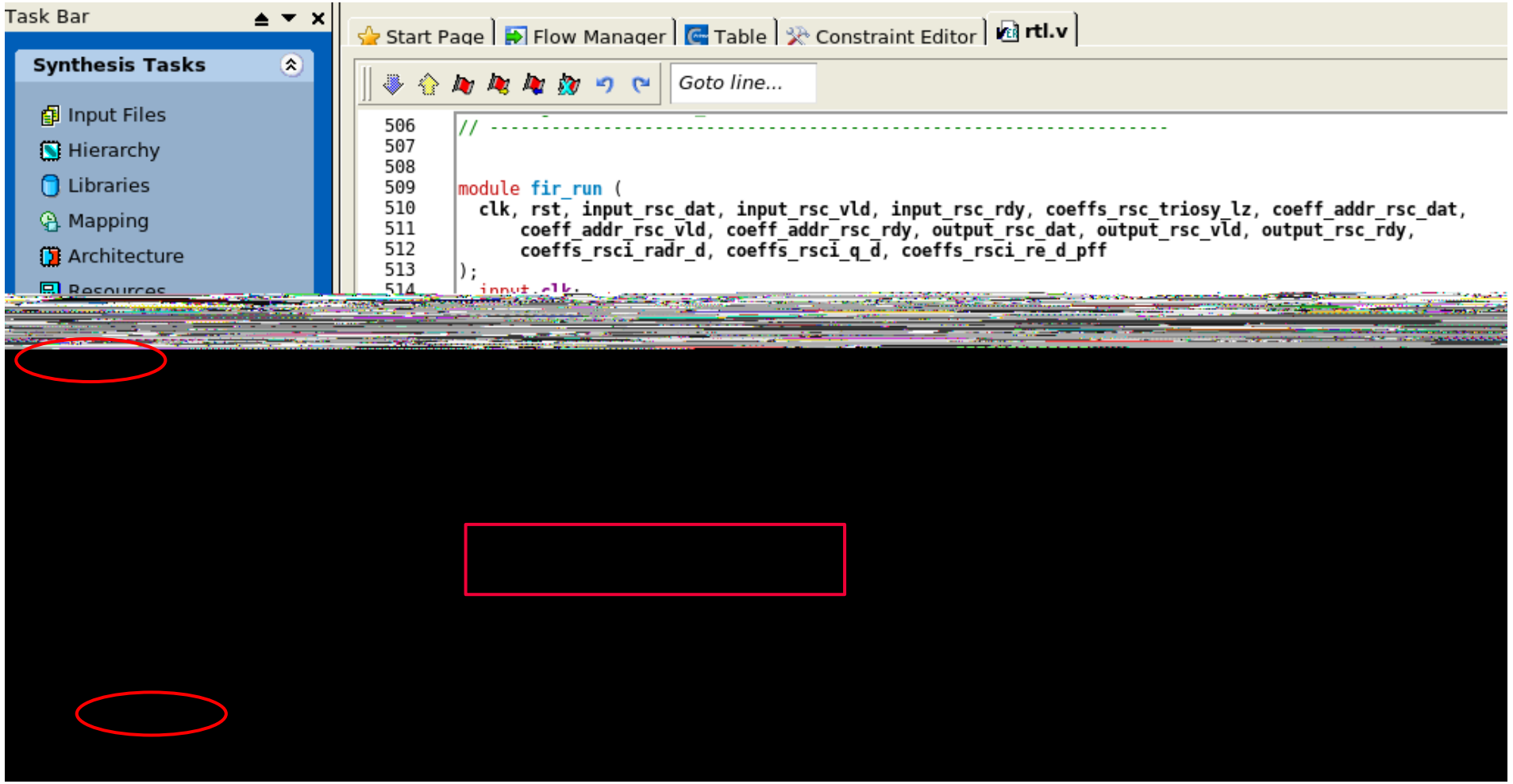




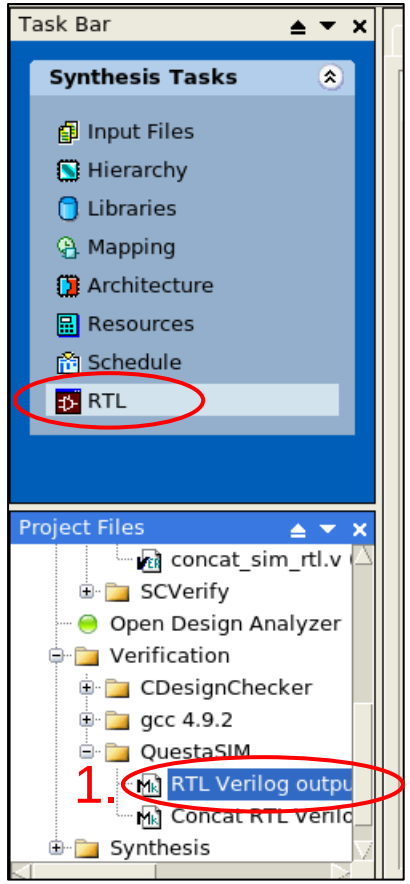
SIEMENS

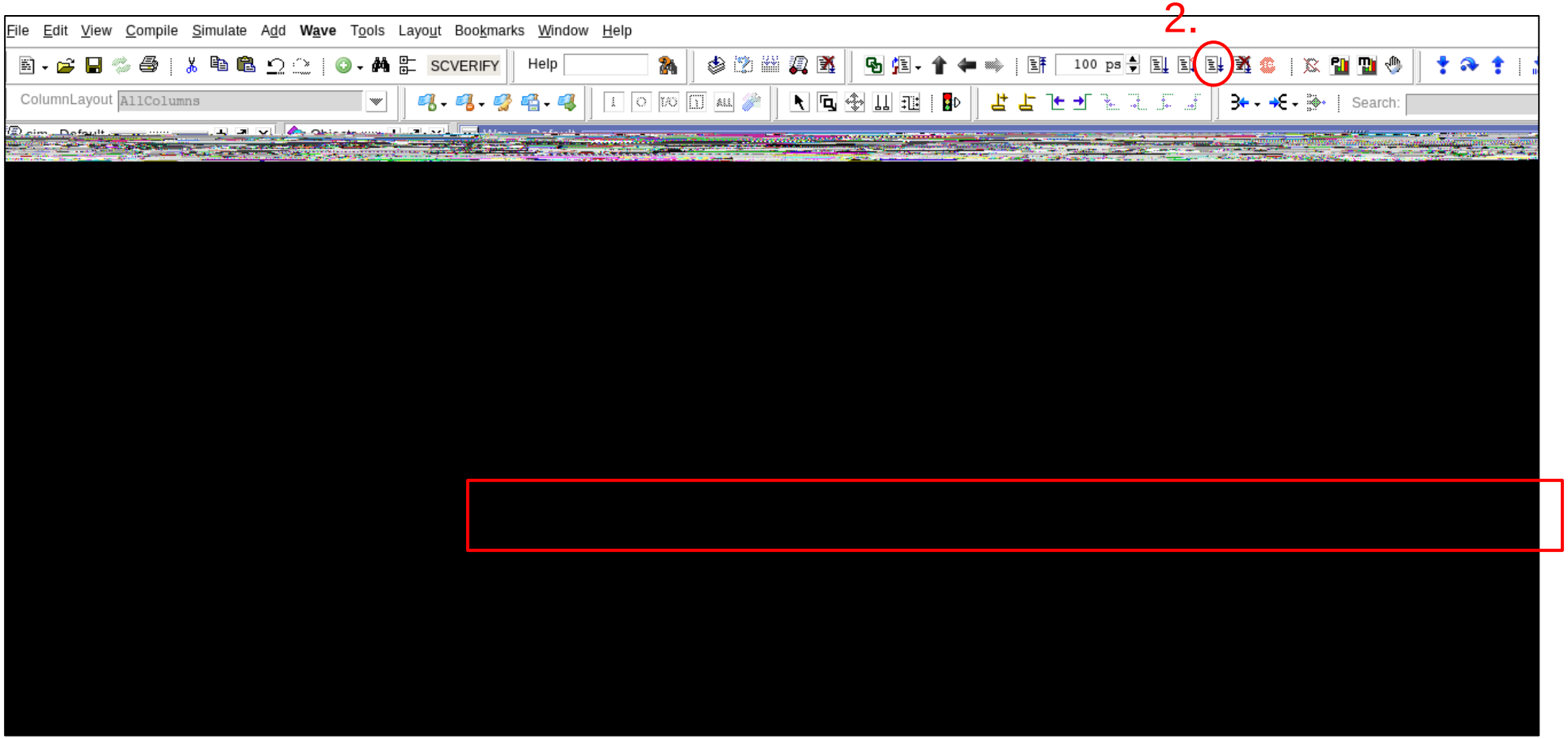
# Check SRAM Interface in RTL

*_re_*: ram read enable; *_radr_*: ram read address; *_d”: ram read data



# Check SRAM Interface in Waveform



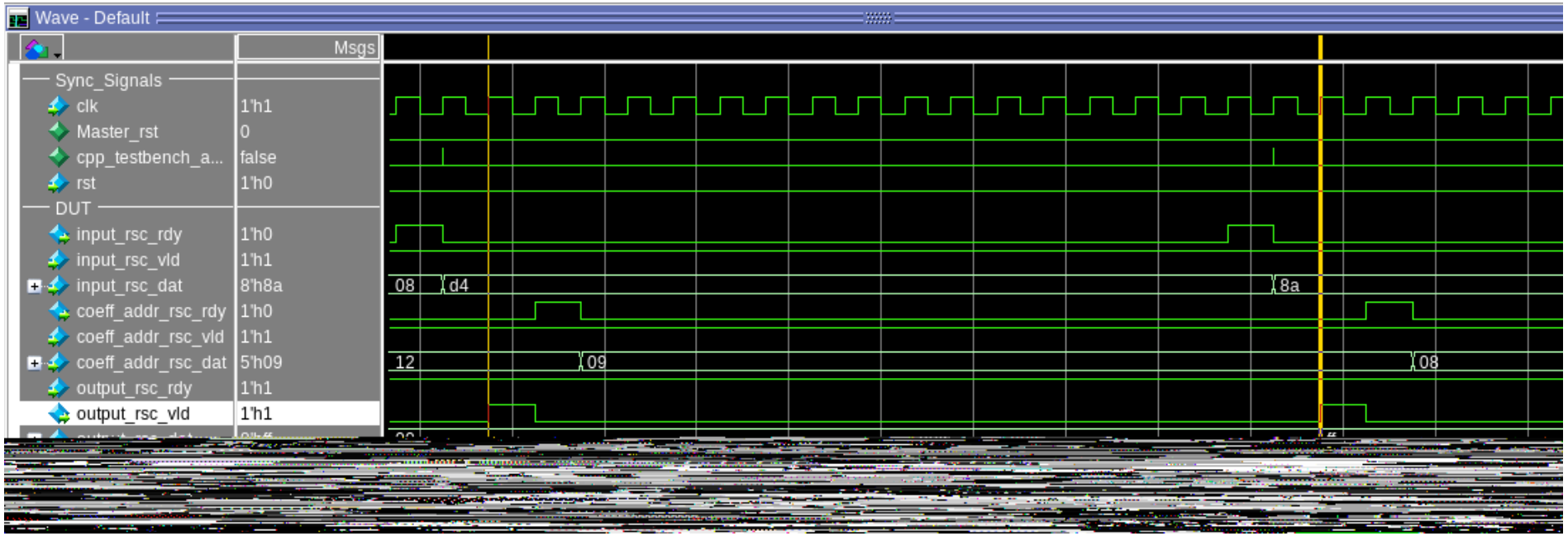



# How do we improve the throughput

Current throughput is 18

Report: General

| 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 |







SIEMENS

# 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);
 }
```

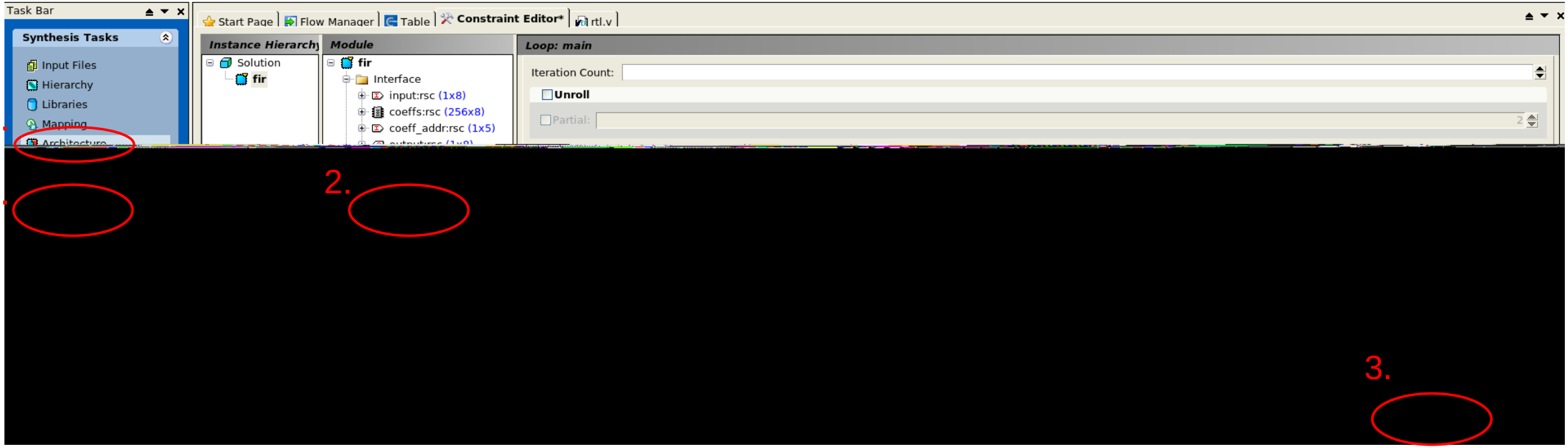
Make convolution parallel

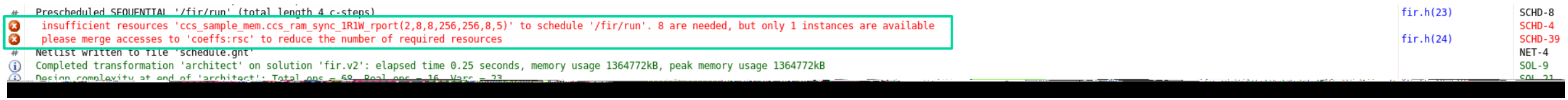



SIEMENS

# Change the Architecture

## Unroll ‘SHIFT’ and ‘MAC’, and Pipeline ‘main’



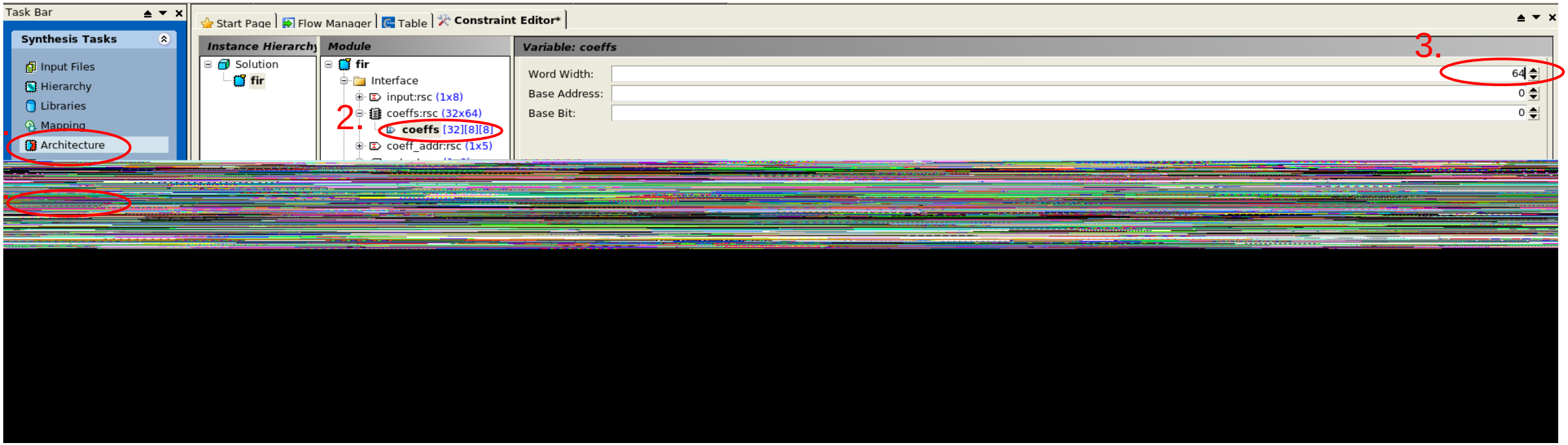




SIEMENS

# Increase Memory Data Width

Word Width = 64 (why ?)




```
#   Generating SCVerify 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



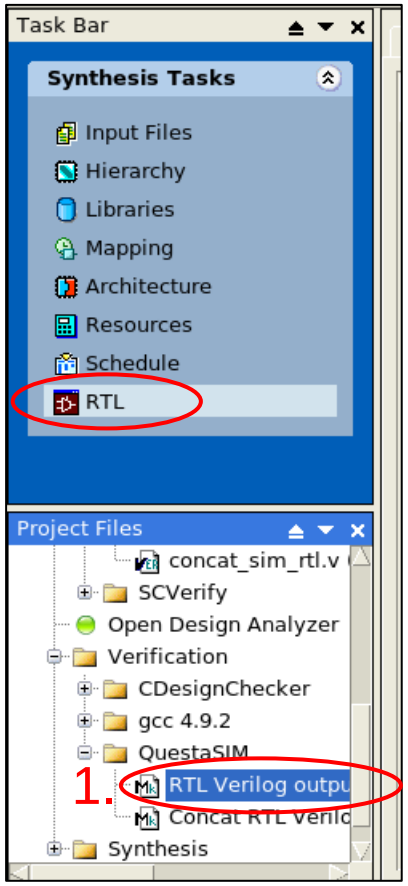

SIEMENS

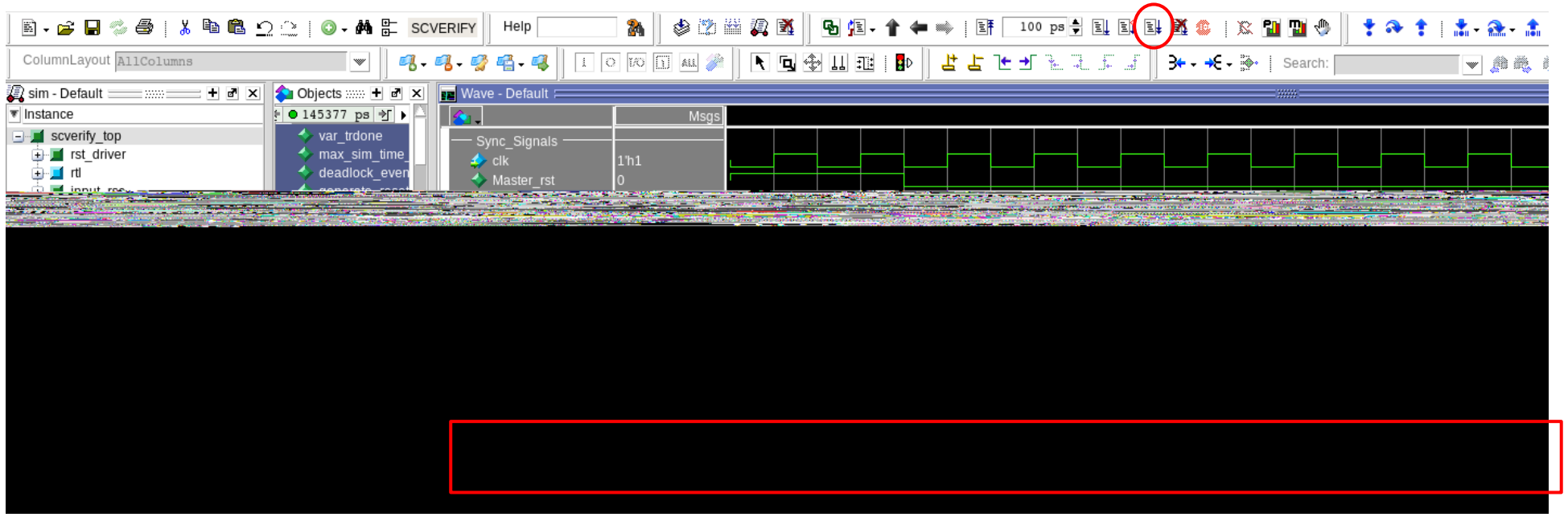
# Check Waveform

Throughput = 1

Report: General

| 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 |
| **fir.v2** (extract) | **2** | **20.00** | **1** | **10.00** | **7.58** | **4024.42** |



SIEMENS

# Multiple Blocks

A Design with Two 8-Tap FIRs and One Decimator




SIEMENS

# 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);
  …
```




SIEMENS

# 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; }
  }
 …
```

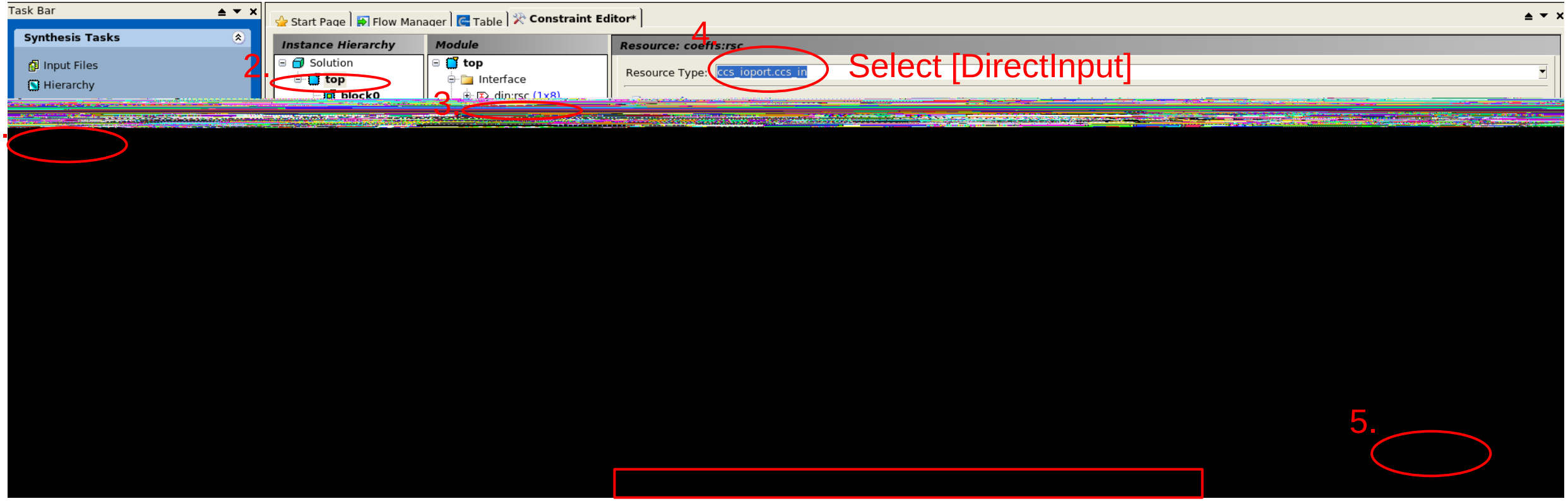
```
#pragma hls_design interface
  void CCS_BLOCK(run)(ac_channel< ac_int<8> > &input,
                      ac_int<8> coeffs[8],
                      ac_channel< ac_int<8> > &output) {
    ac_int<19> temp = 0;
    SHIFT:for (int i = 7; i>=0; i--) {
      if ( i == 0 ) {
        regs[i] = input.read();
      } 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 &

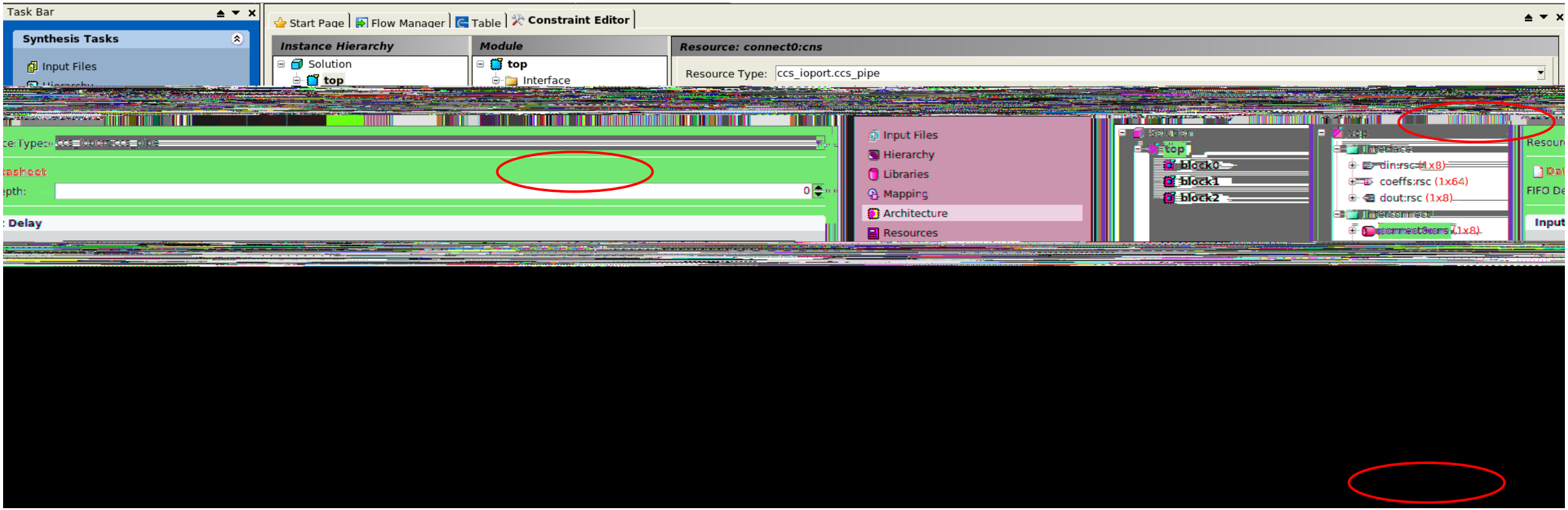
There are three design blocks: block0, block1, block2

Set ‘DirectInput’ for the coeffs



# Set FIFO Depth as ‘0’

Set FIFO depth as ‘0’ for connect0 and connect1





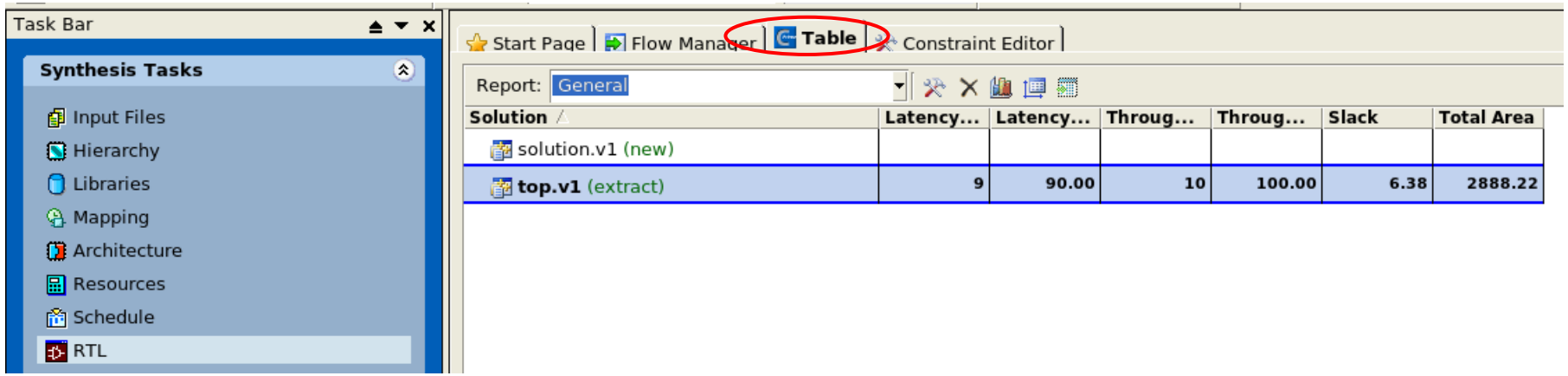


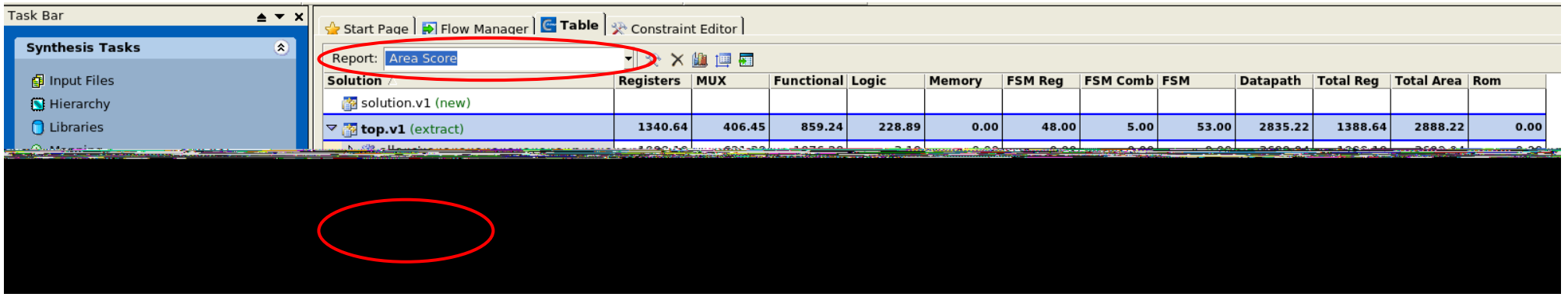
SIEMENS

# Look Report

Latency = ?, Throughput = ?

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



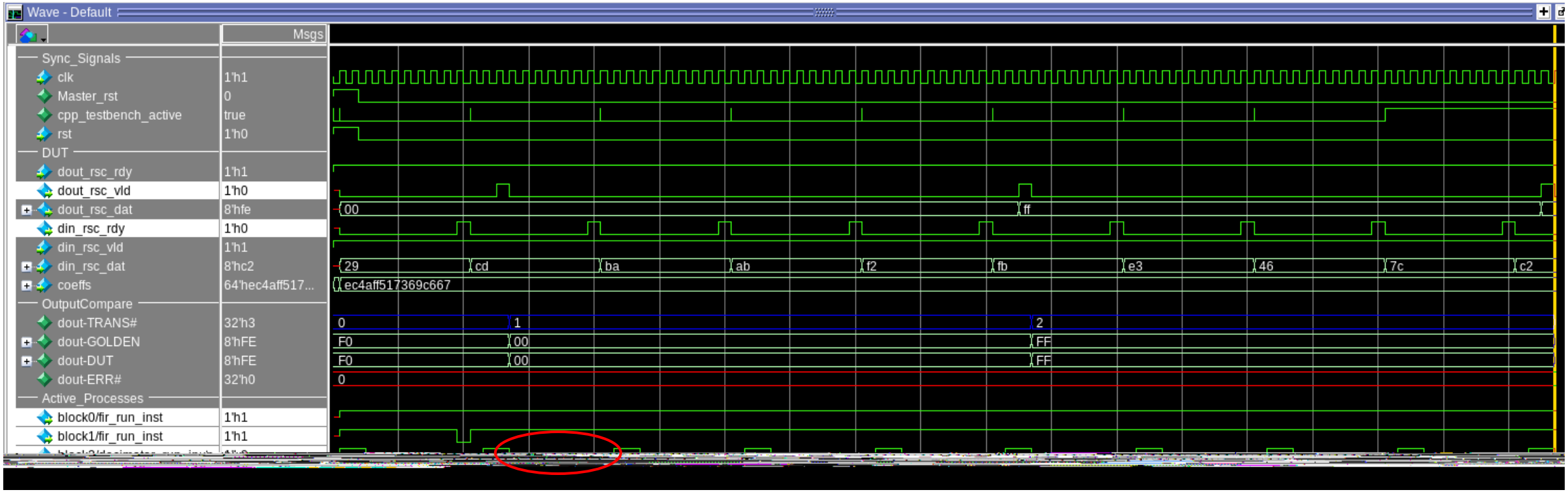







SIEMENS

# RTL Simulation



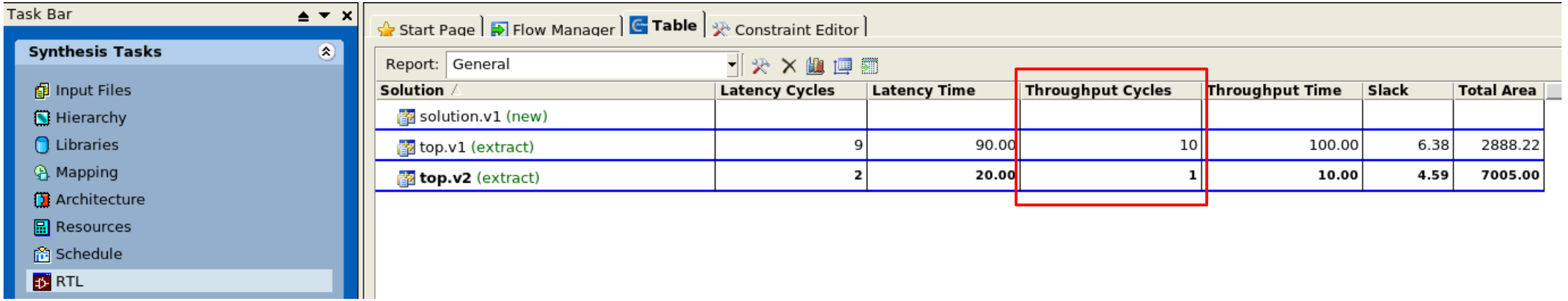

why idle ?




SIEMENS

# Improve Throughput

Improve the throughput from 10 to 1 by yourself



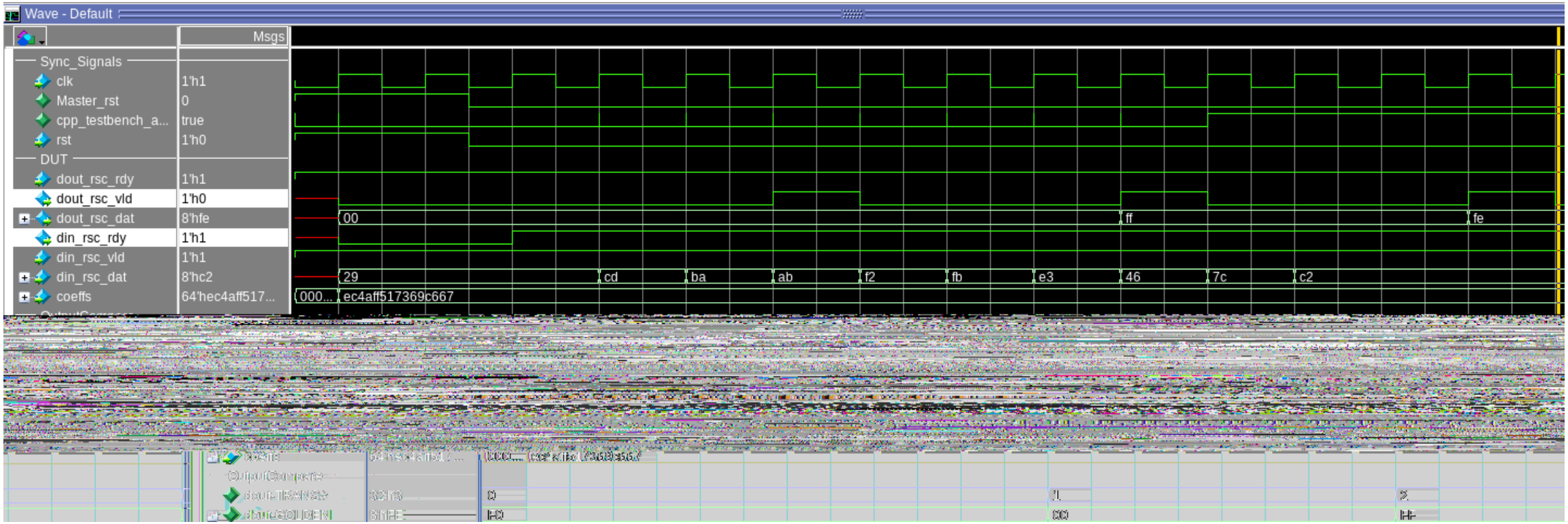

| Solution | Latency Cycles | Latency Time | Throughput Cycles | Throughput Time | Slack | Total Area |
|---|---|---|---|---|---|---|
| solution.v1 (new) | | | | | | |
| top.v1 (extract) | 9 | 90.00 | 10 | 100.00 | 6.38 | 2888.22 |
| **top.v2** (extract) | **2** | **20.00** | **1** | **10.00** | **4.59**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




SIEMENS

# RTL Simulation (Throughput=1)

One dout every four din because of the decimation







SIEMENS