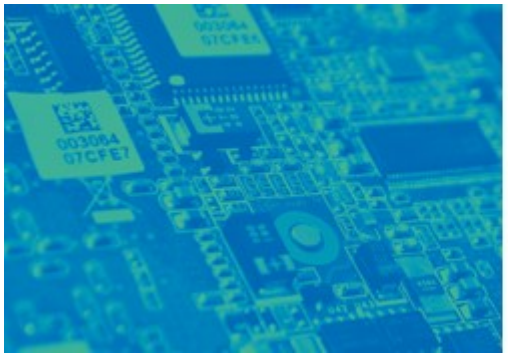
FPGA Programming

Tutorial

Wojciech Jalmuzna
Technical Univeristy of Lodz



LLRF 2011

17-20 October 2011, DESY, Hamburg

INTRODUCTION

- What is an FPGA
- Short FPGA History
- Programming Language vs. Hardware Description Language
- Conclusions ?

Files can be found at:
http://tesla.desy.de/~wjalmuzn/llrf11.tgz

What is FPGA ? (1)

FPGA stands for **F**ield **P**rogrammable **G**ate **A**rray

The best way to show You what is an FPGA is to compare it with micro-controller, DSP or CPU

What is FPGA ? (2)

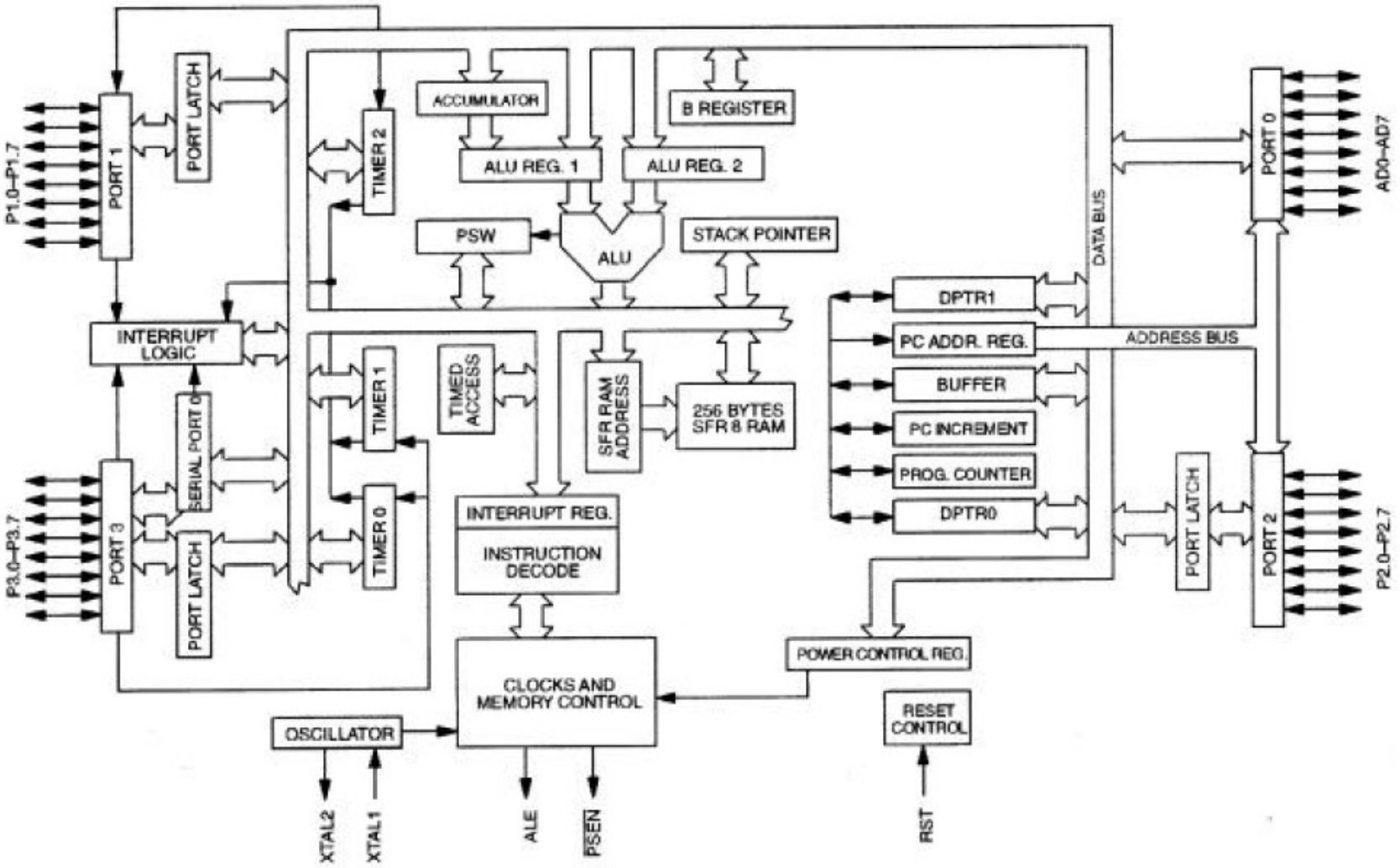


Figure: Maxim 8051-Compatible uProcessor

What is FPGA ? (3)

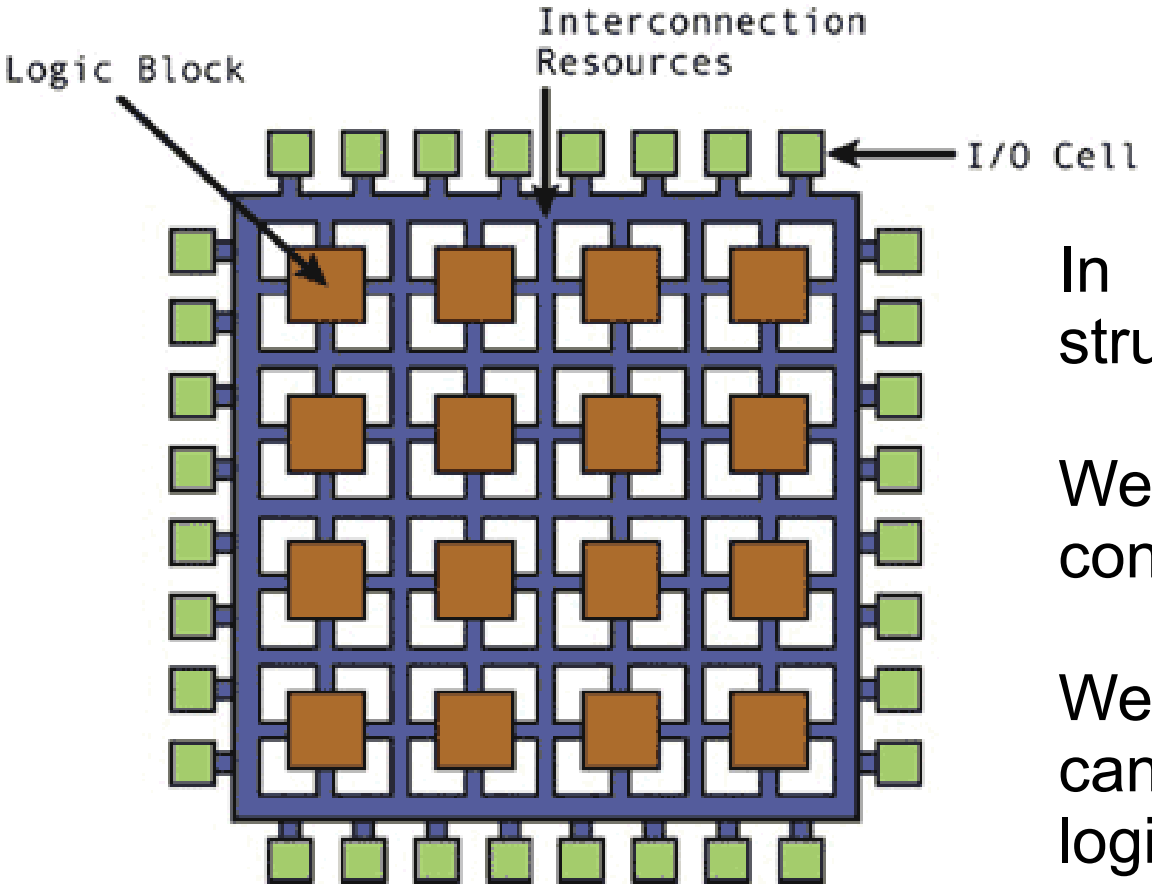


Figure: General FPGA block diagram

In comparison with uProcessor structure of FPGA is regular.

We have logic blocks, which contain “gates” and flip-flops.

We have interconections, which can build connection between logic blocks,

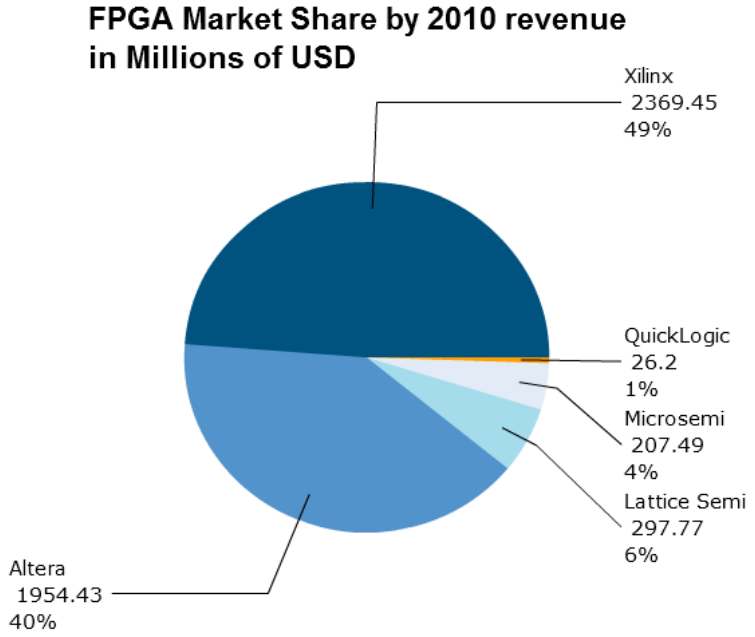
We have IO Cells which are used to interface with outer world.

Short FPGA History

Market size

- 1985: First commercial FPGA technology invented by Xilinx
- 1987: $14 million
- ~1993: >$385 million
- 2005: $1.9 billion
- 2010 estimates: $2.75 billion

Market size corresponds to FPGA size (available logic gates)



Everything started with ~9000 gates and till now this amount increased by almost 1000 times. Additional embedded peripherals are now available (internal memories, fast arithmetic units, gigabit links)

Programming Language vs. HDL

Can we say that we are “programming” FPGA ?

Description written in a programming language is translated into machine code, which is executed sequentially by a “program sequencer” of a target unit (CPU,DSP,uC).

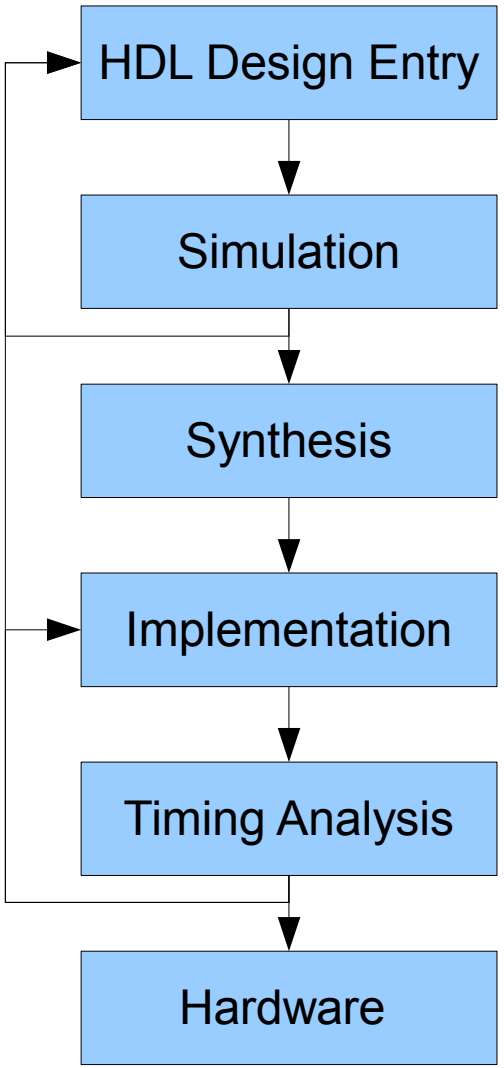
Since FPGA has no predefined higher level structure (only flip-flops and logic gates) “program” should be rather called “structure description”.

Which X is better ?

X	
VENDOR	Altera,Xilinx ?
HDL	VHDL, Verilog, SystemVerilog, SystemC ?
TOOLS	Altera Quartus, Xilinx ISE, Synplicity ?

There is no correct or incorrect answer :)

Design Flow



STAGES	TOOLS
HDL Design Entry	Any text or schematic editor, System Generator
Simulation	ModelSim, ActiveHDL, RivieraPRO, gHDL, **ISIM**
Synthesis	PrecisionRTL, Synplicity, **xst**
Implementation	ngdbuild, map, par
Timing Analysis	Timing Analyzer, PlanAhead
Hardware	Impact, Chipscope

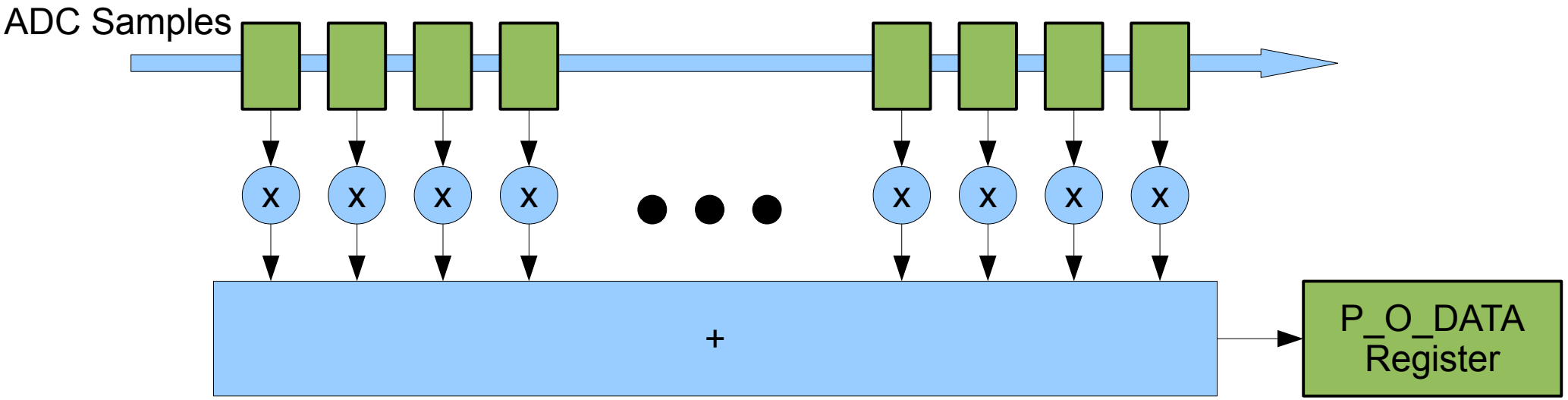
Design Entry - FIR

Describing in VHDL – simple behavioral description (as you would write in high level programming language)

```
...
    elsif rising_edge(P_I_CLK) then

      for I in CON_ORDER-1 downto 1 loop
        SIG_ADC(I) <= SIG_ADC(I-1);
      end loop;
      SIG_ADC(0) <= P_I_DATA;

      v_sum := (others => '0');

      for I in 0 to CON_ORDER-1 loop
        v_mult := SIG_ADC(I)*SIG_COEF(I);
        v_sum  := v_sum + v_mult;
      end loop;

      P_O_DATA <= v_sum;
...
```

VHDL: Simple behavioral FIR implementation

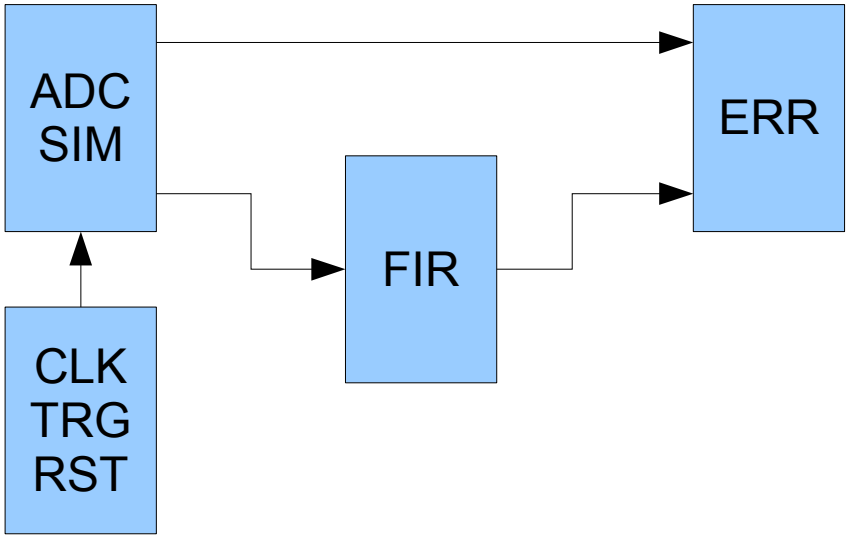
Simulation - FIR



Filter is implemented as an ADC pipeline. Each element is multiplied by a filter coefficient and all results are summed up. The operation takes one clock cycle.

Simulation Testbench

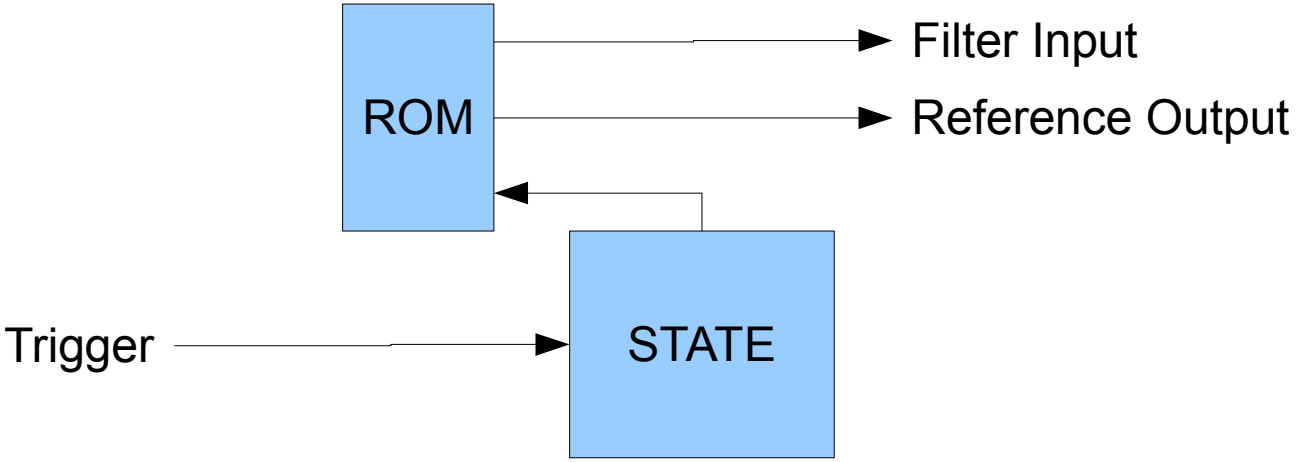
Test-bench preparation.
Some syntax which cannot be synthesized

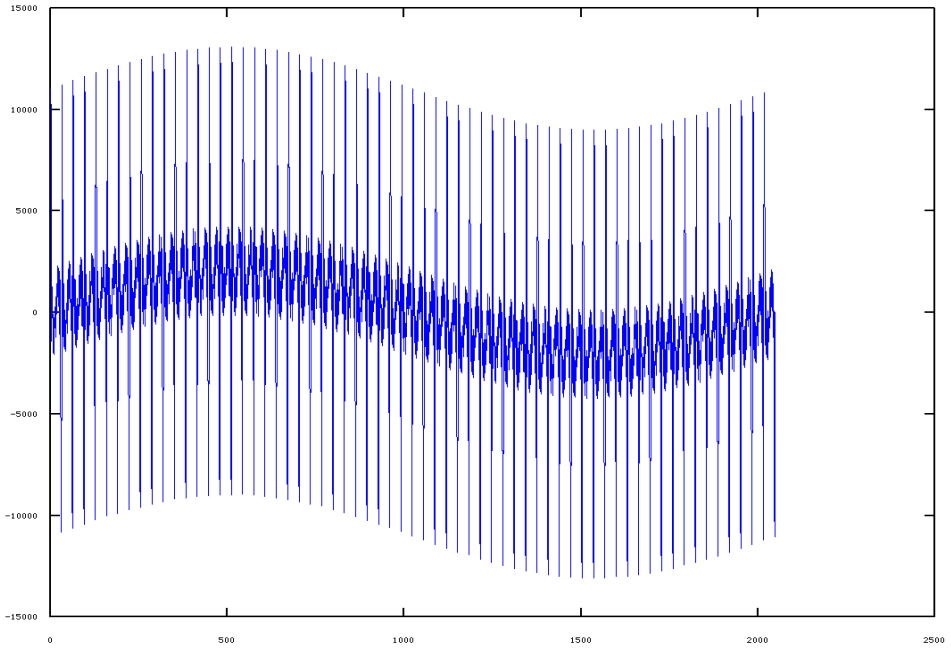


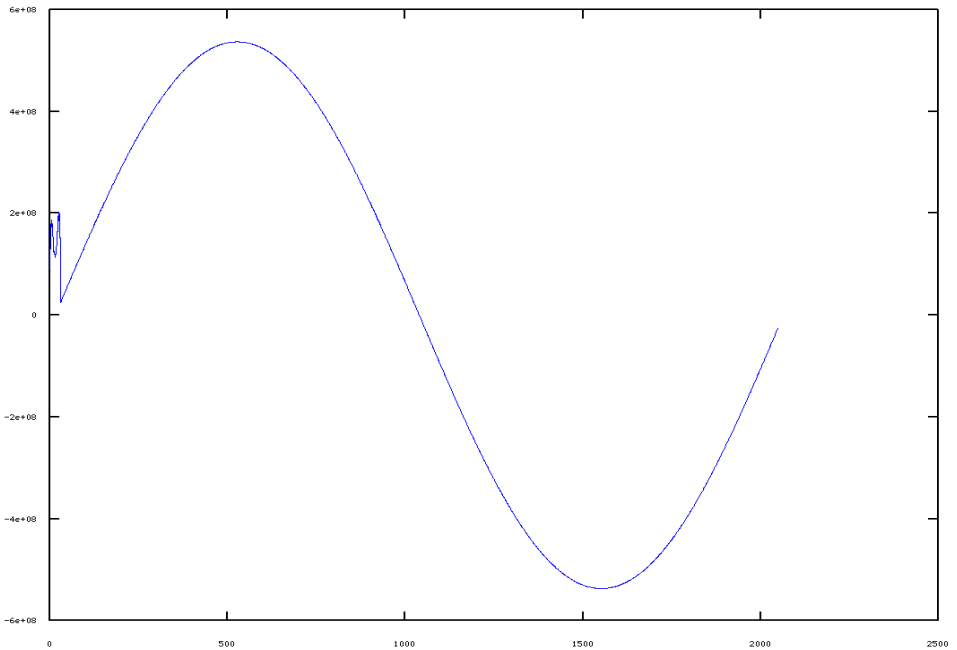
ADC SIM module has been crated to provide data stream

Data from simulator has been exported to VCD file for visualization

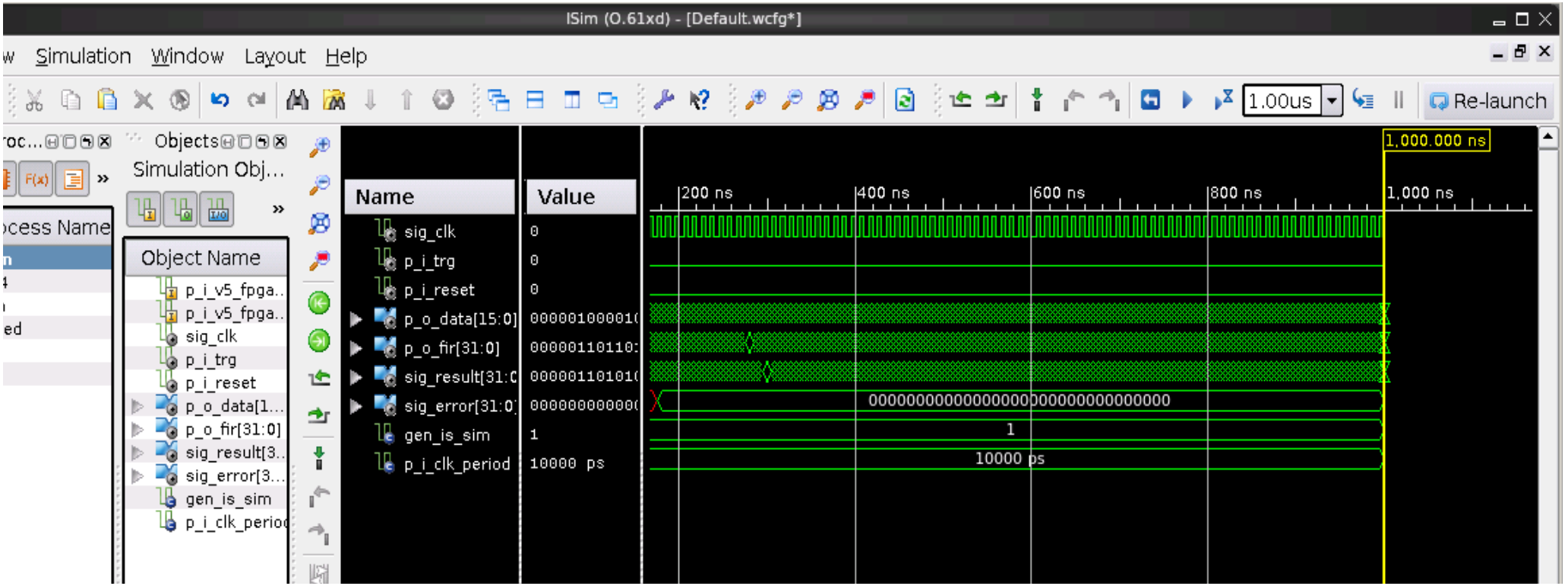
Dedicated Module



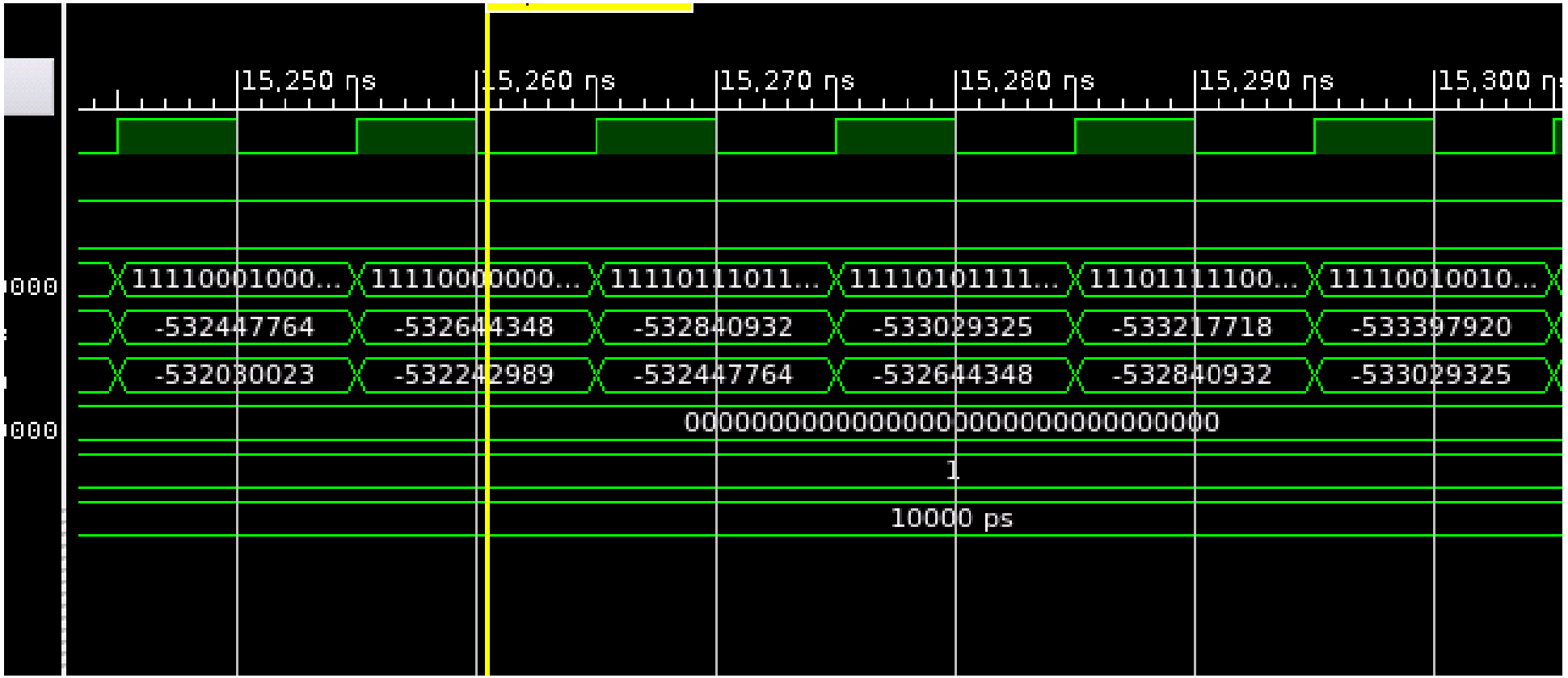




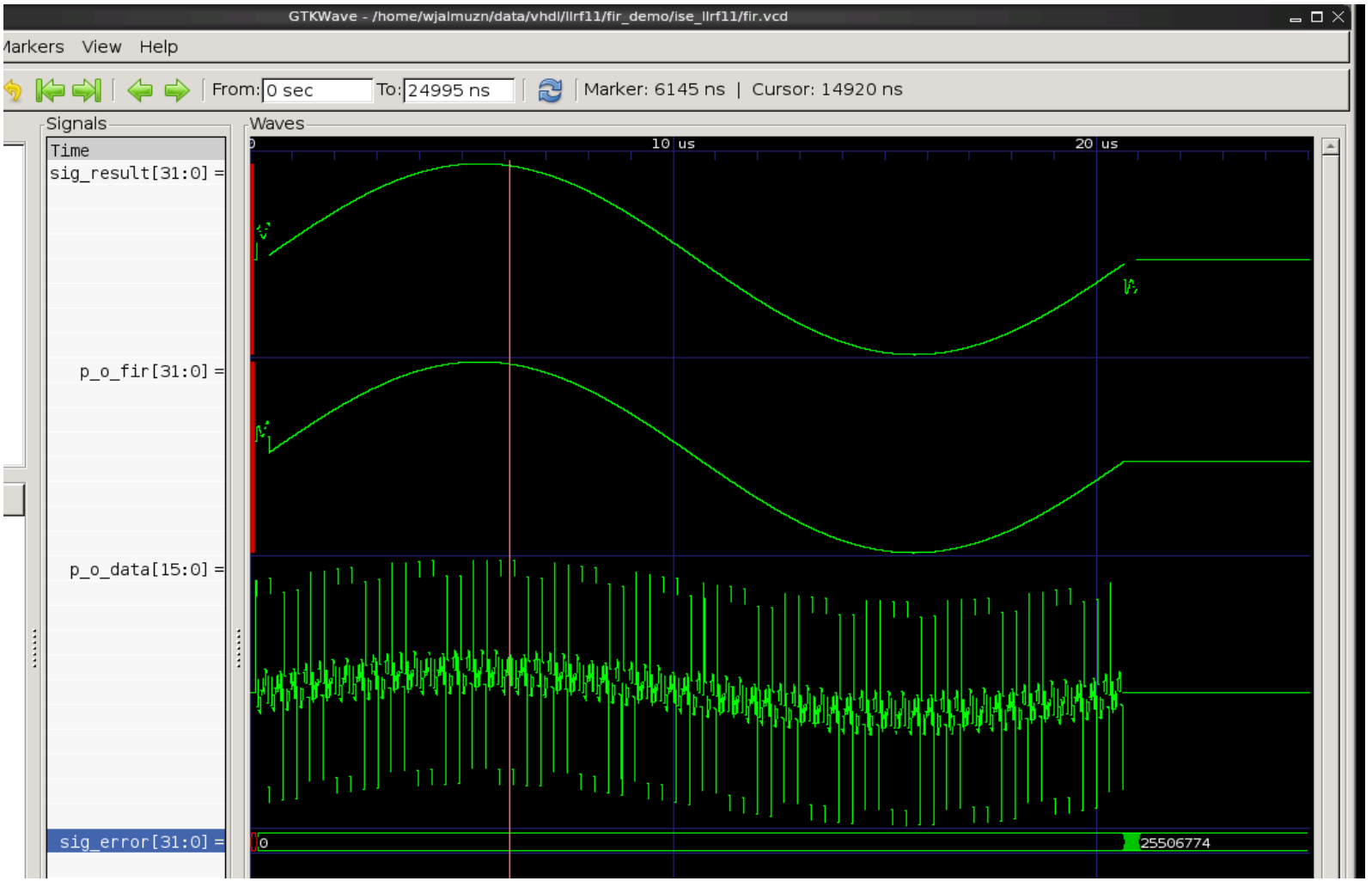
Simulation - FIR



Simulation - FIR



Result Analysis - FIR



Synthesis - FIR

What is Synthesis ? Conversion from some abstract description to logic gates. Special macros and FPGA built-in blocks are also detected and used. This is first step to implement design for FPGA.

- Modifications to test-bench
- Chipscope Pro for debugging

Synthesis Report

```
...
Macro Statistics
# MACs                                                : 31
 16x16-to-32-bit MAC                                  : 31
# Multipliers                                         : 1
 16x16-bit multiplier                                 : 1
# Registers                                           : 512
 Flip-Flops                                           : 512
...
```

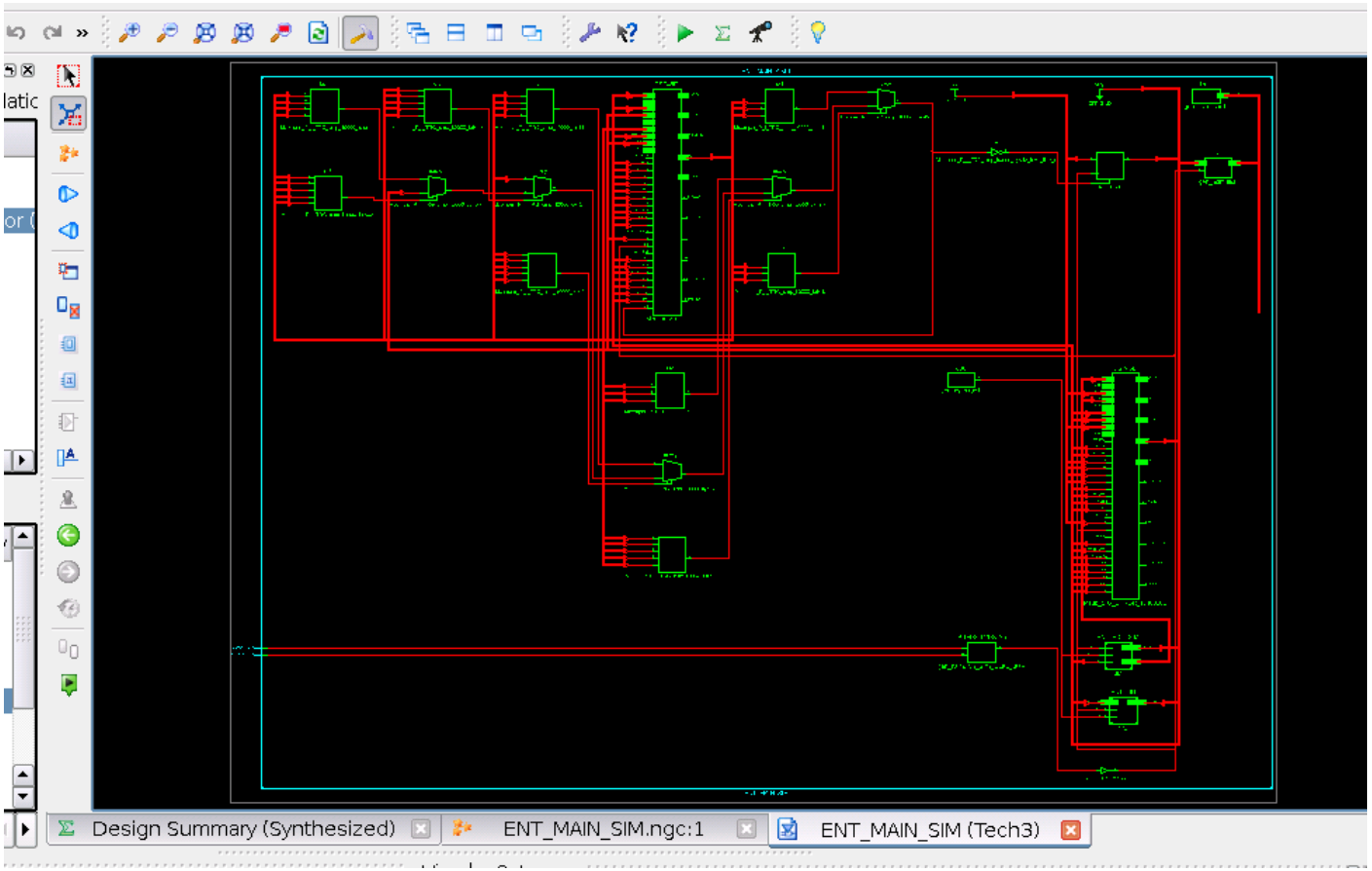
Report: FIR Macro Statistics

```
...
Timing Summary:
---------------
Speed Grade: -1

   Minimum period: 66.385ns (Maximum Frequency: 15.064MHz)
   Minimum input arrival time before clock: 2.670ns
   Maximum output required time after clock: 3.344ns
   Maximum combinational path delay: No path found
...
```

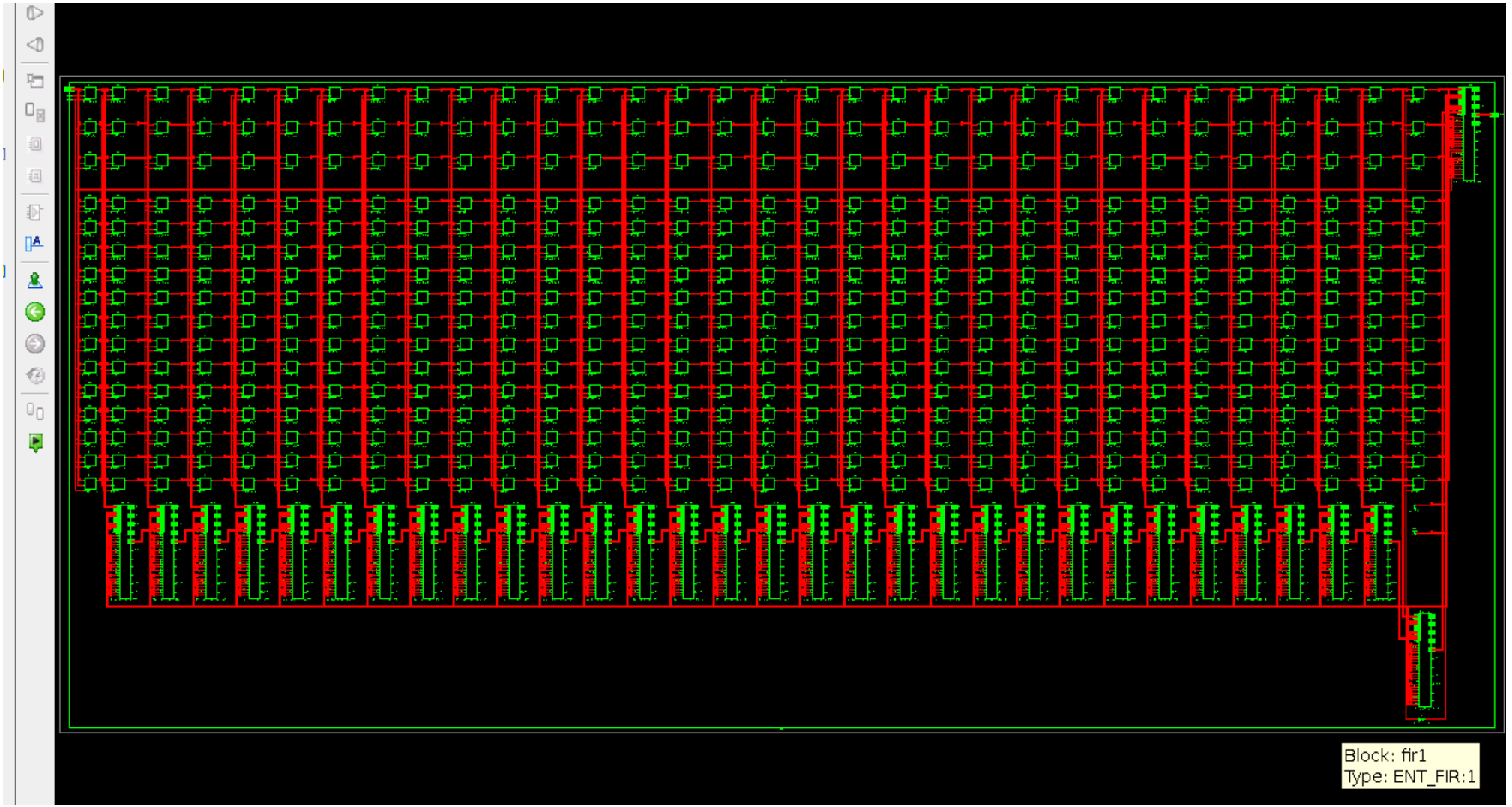
Report: FIR Macro Statistics

Xilinx Schematic Viewer (1)

Limitations: Even for simple design RTL schematic can be huge. The best is to know which path or module part you want to observe.



FIR Schematic Viewer (2)



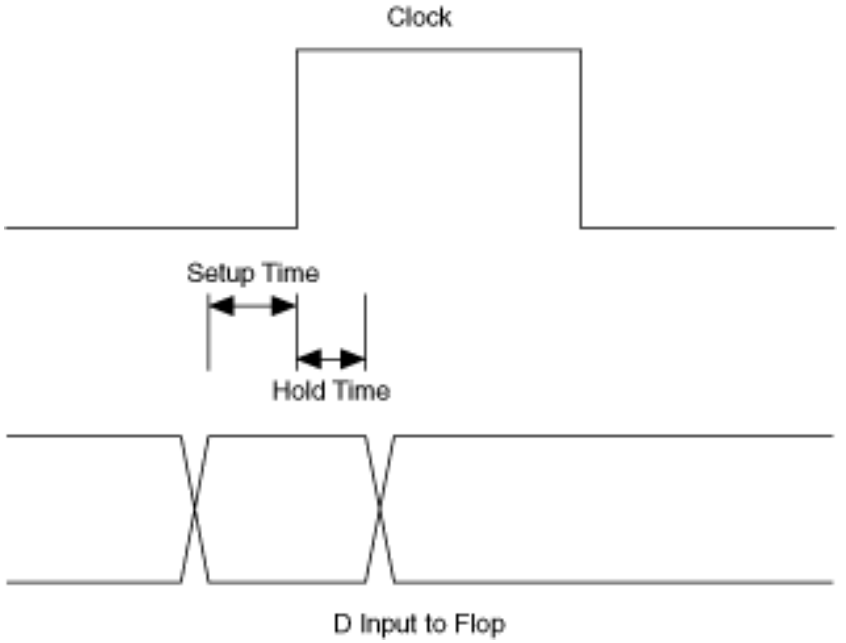
To Hardware

- Timing Constraints
- Implementation process
- Timing reports
- Advanced timing analysis (planAhead)
- Design verification (chipscope)

Timing Constraints

Basic timing constraint is PERIOD.

During timing analysis SETUP and HOLD violations are checked.



They cause flip-flops to go metastable

Implementation

After synthesis, you run design implementation, which comprises the following steps:

Translate - merges the incoming netlists and constraints into a single design file.

Map - fits the design into the available resources on the target device, and optionally, places the design.

Place and Route - places and routes the design to the timing constraints.

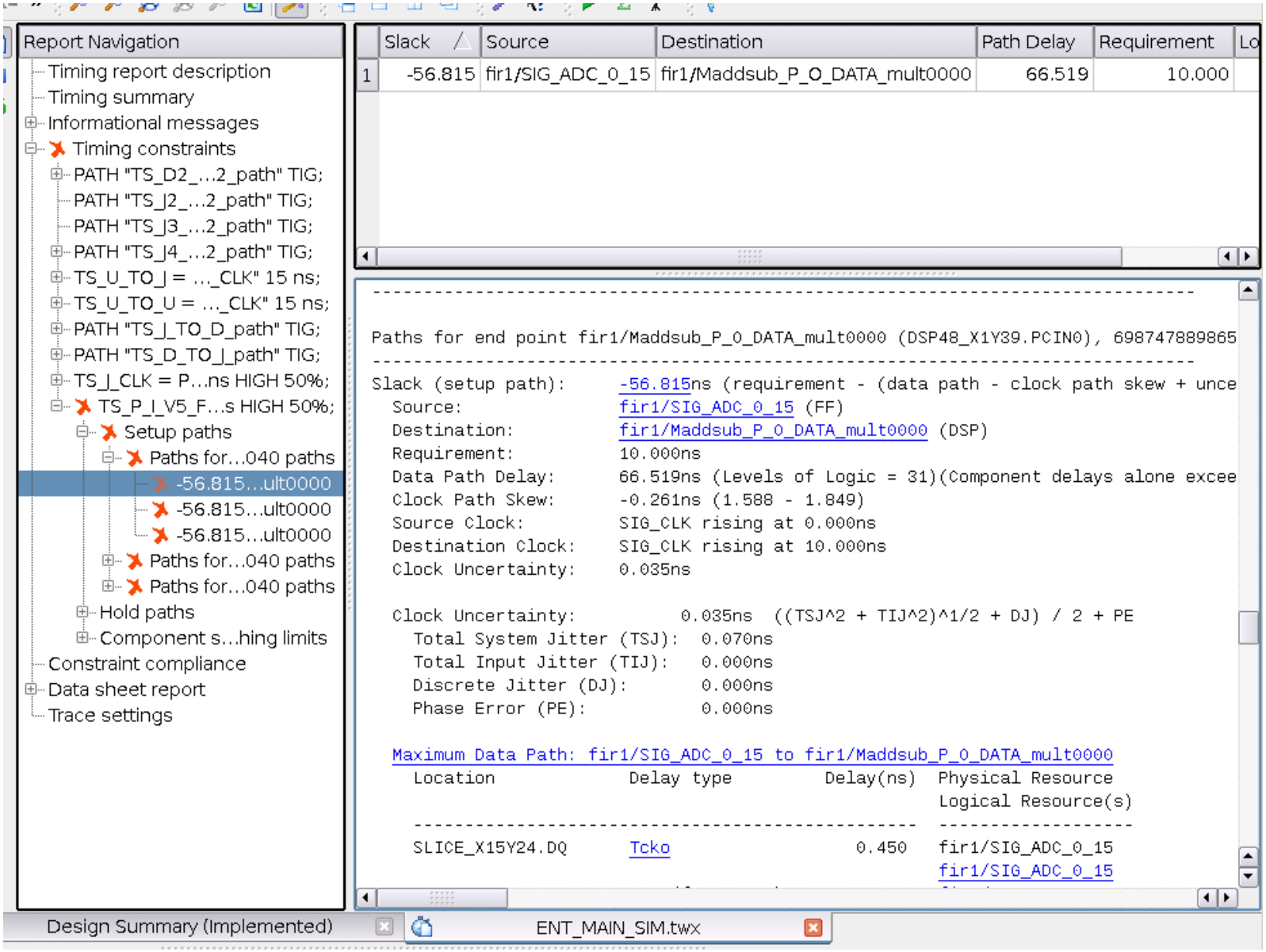
Generate Programming File - creates a bitstream file that can be downloaded to the device.

Timing Analysis (1)

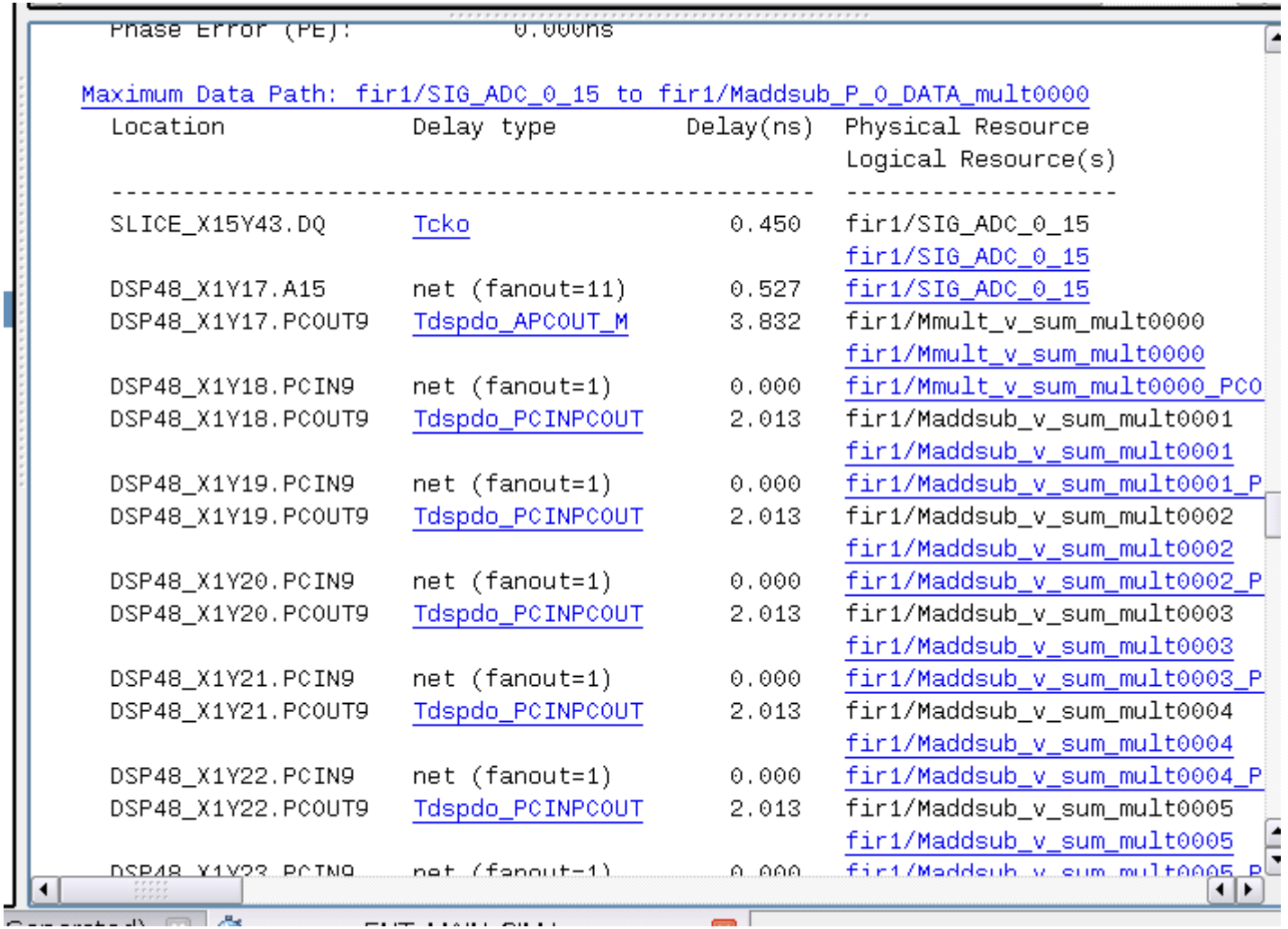
Constraint	Check	Worst Case Slack	Best Case Achievable	Timing Errors	Timing Score
* TS_P_I_V5_FPGA_CLK_P = PERIOD TIMEGRP "TN M_P_I_V5_FPGA_CLK_P" 10 ns HIGH 50%	SETUP	-56.815ns	66.815ns	48	2727120
	HOLD	0.255ns		0	0
TS_U_TO_J = MAXDELAY FROM TIMEGRP "U_CLK" TO TIMEGRP "J_CLK" 15 ns	SETUP	13.034ns	1.966ns	0	0
	HOLD	1.439ns		0	0
TS_U_TO_U = MAXDELAY FROM TIMEGRP "U_CLK" TO TIMEGRP "U_CLK" 15 ns	SETUP	14.103ns	0.897ns	0	0
	HOLD	0.568ns		0	0
TS_J_CLK = PERIOD TIMEGRP "J_CLK" 30 ns HIGH 50%	SETUP	19.371ns	10.629ns	0	0
	HOLD	0.324ns		0	0
PATH "TS_D2_TO_T2_path" TIG	SETUP	N/A	2.783ns	N/A	0
PATH "TS_J2_TO_D2_path" TIG	N/A	N/A	N/A	N/A	N/A
PATH "TS_J3_TO_D2_path" TIG	N/A	N/A	N/A	N/A	N/A
PATH "TS_J4_TO_D2_path" TIG	MAXDELAY	N/A	3.602ns	N/A	0
PATH "TS_J_TO_D_path" TIG	SETUP	N/A	3.917ns	N/A	0
PATH "TS_D_TO_J_path" TIG	SETUP	N/A	3.451ns	N/A	0

VHDL: Simple behavioral FIR implementation

Timing Analysis (2)



Timing Analysis (3)

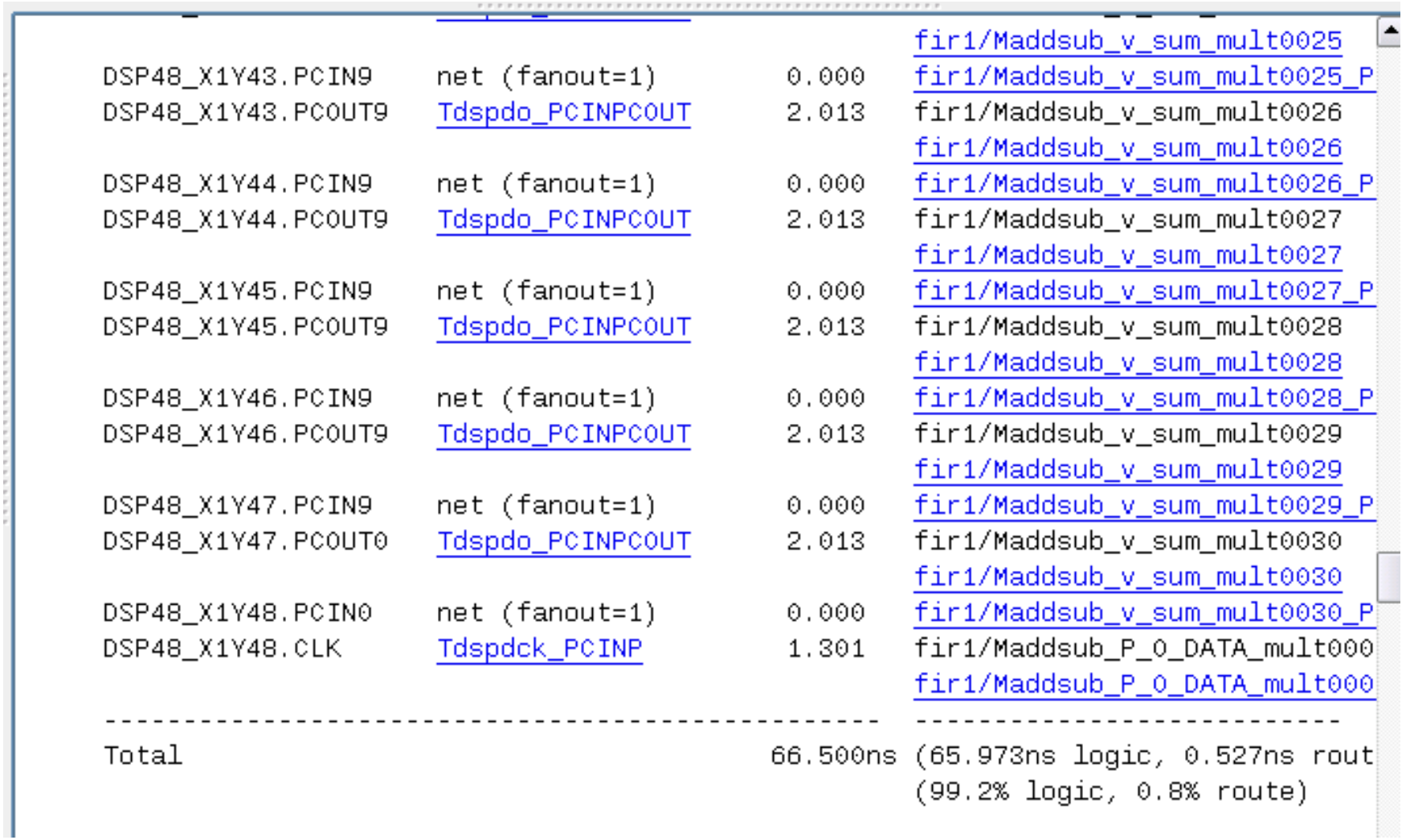


Phase Error (PE): 0.000ns

Maximum Data Path: fir1/SIG_ADC_0_15 to fir1/Maddsub_P_0_DATA_mult0000

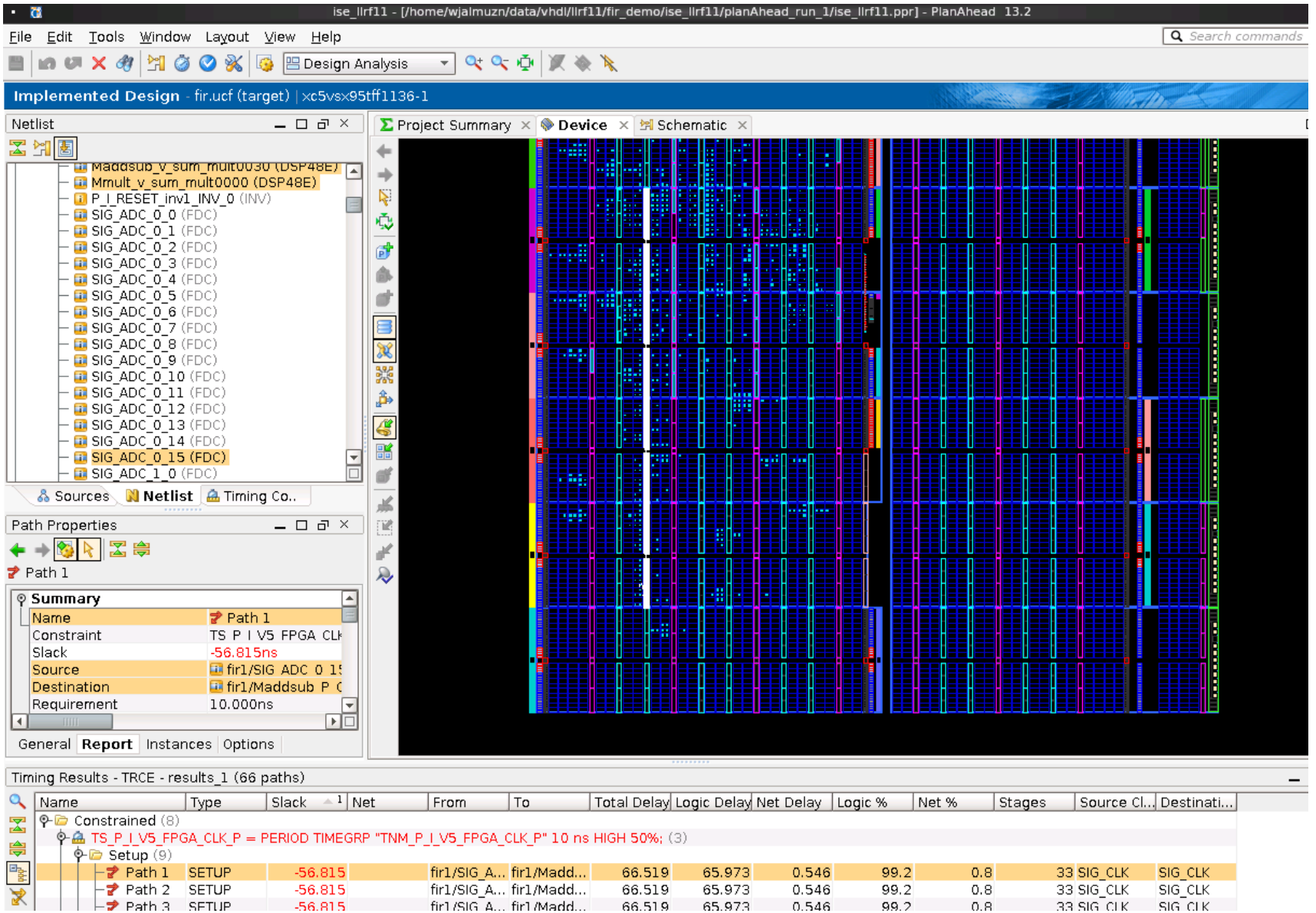
Location	Delay type	Delay(ns)	Physical Resource Logical Resource(s)
SLICE_X15Y43.DQ	Tcko	0.450	fir1/SIG_ADC_0_15 fir1/SIG_ADC_0_15
DSP48_X1Y17.A15	net (fanout=11)	0.527	fir1/SIG_ADC_0_15
DSP48_X1Y17.PCOUT9	Tdspdo_APCOUT_M	3.832	fir1/Mmult_v_sum_mult0000 fir1/Mmult_v_sum_mult0000
DSP48_X1Y18.PCIN9	net (fanout=1)	0.000	fir1/Mmult_v_sum_mult0000_PC0
DSP48_X1Y18.PCOUT9	Tdspdo_PCINPCOUT	2.013	fir1/Maddsub_v_sum_mult0001 fir1/Maddsub_v_sum_mult0001
DSP48_X1Y19.PCIN9	net (fanout=1)	0.000	fir1/Maddsub_v_sum_mult0001_P
DSP48_X1Y19.PCOUT9	Tdspdo_PCINPCOUT	2.013	fir1/Maddsub_v_sum_mult0002 fir1/Maddsub_v_sum_mult0002
DSP48_X1Y20.PCIN9	net (fanout=1)	0.000	fir1/Maddsub_v_sum_mult0002_P
DSP48_X1Y20.PCOUT9	Tdspdo_PCINPCOUT	2.013	fir1/Maddsub_v_sum_mult0003 fir1/Maddsub_v_sum_mult0003
DSP48_X1Y21.PCIN9	net (fanout=1)	0.000	fir1/Maddsub_v_sum_mult0003_P
DSP48_X1Y21.PCOUT9	Tdspdo_PCINPCOUT	2.013	fir1/Maddsub_v_sum_mult0004 fir1/Maddsub_v_sum_mult0004
DSP48_X1Y22.PCIN9	net (fanout=1)	0.000	fir1/Maddsub_v_sum_mult0004_P
DSP48_X1Y22.PCOUT9	Tdspdo_PCINPCOUT	2.013	fir1/Maddsub_v_sum_mult0005 fir1/Maddsub_v_sum_mult0005
DSP48_X1Y23.PCIN9	net (fanout=1)	0.000	fir1/Maddsub_v_sum_mult0005_P

Timing Analysis (4)

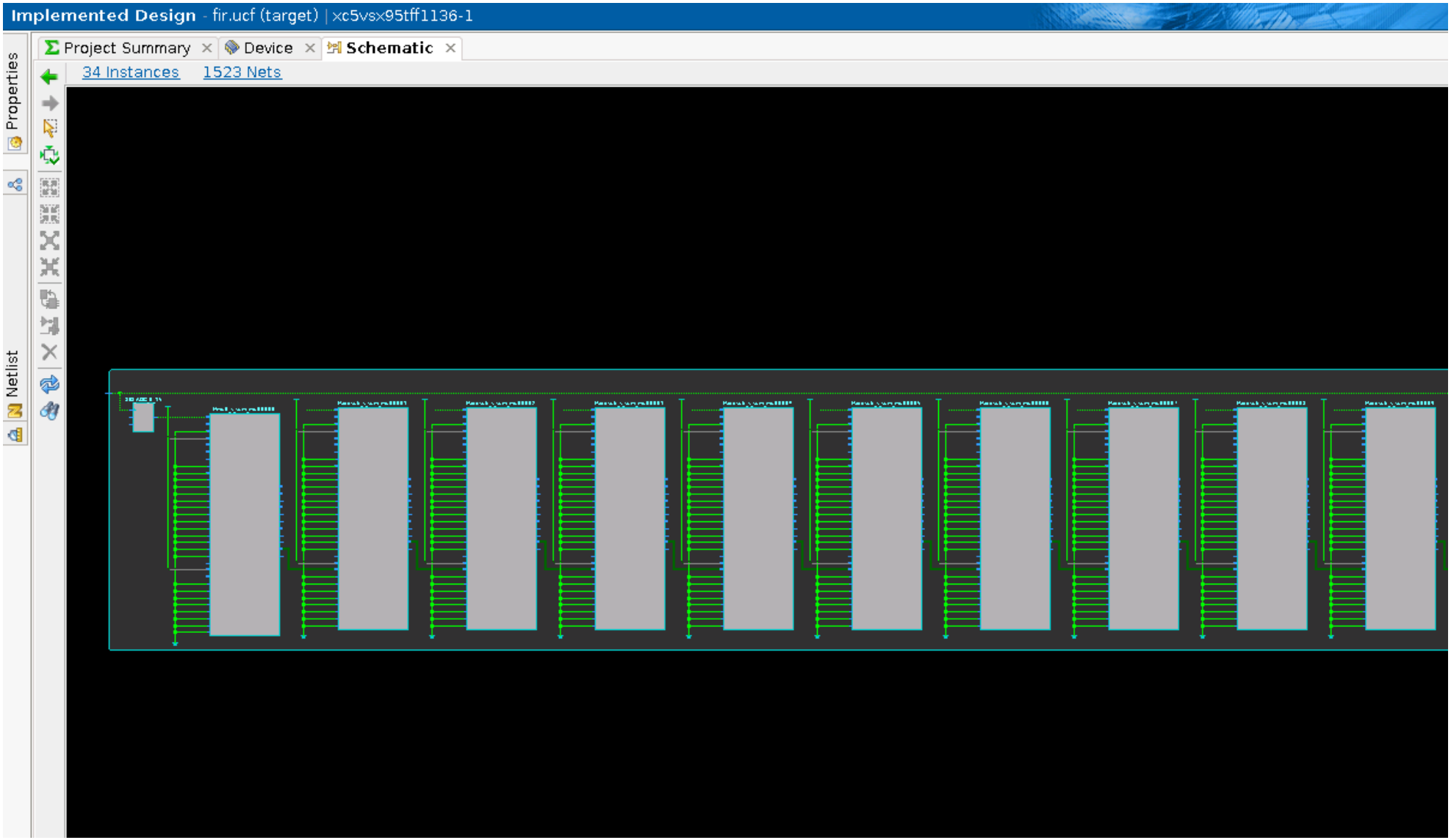


```
                                                          fir1/Maddsub_v_sum_mult0025
DSP48_X1Y43.PCIN9     net (fanout=1)        0.000       fir1/Maddsub_v_sum_mult0025_P
DSP48_X1Y43.PCOUT9    Tdspdo_PCINPCOUT      2.013       fir1/Maddsub_v_sum_mult0026
                                                          fir1/Maddsub_v_sum_mult0026
DSP48_X1Y44.PCIN9     net (fanout=1)        0.000       fir1/Maddsub_v_sum_mult0026_P
DSP48_X1Y44.PCOUT9    Tdspdo_PCINPCOUT      2.013       fir1/Maddsub_v_sum_mult0027
                                                          fir1/Maddsub_v_sum_mult0027
DSP48_X1Y45.PCIN9     net (fanout=1)        0.000       fir1/Maddsub_v_sum_mult0027_P
DSP48_X1Y45.PCOUT9    Tdspdo_PCINPCOUT      2.013       fir1/Maddsub_v_sum_mult0028
                                                          fir1/Maddsub_v_sum_mult0028
DSP48_X1Y46.PCIN9     net (fanout=1)        0.000       fir1/Maddsub_v_sum_mult0028_P
DSP48_X1Y46.PCOUT9    Tdspdo_PCINPCOUT      2.013       fir1/Maddsub_v_sum_mult0029
                                                          fir1/Maddsub_v_sum_mult0029
DSP48_X1Y47.PCIN9     net (fanout=1)        0.000       fir1/Maddsub_v_sum_mult0029_P
DSP48_X1Y47.PCOUT0    Tdspdo_PCINPCOUT      2.013       fir1/Maddsub_v_sum_mult0030
                                                          fir1/Maddsub_v_sum_mult0030
DSP48_X1Y48.PCIN0     net (fanout=1)        0.000       fir1/Maddsub_v_sum_mult0030_P
DSP48_X1Y48.CLK       Tdspdck_PCINP         1.301       fir1/Maddsub_P_O_DATA_mult000
                                                          fir1/Maddsub_P_O_DATA_mult000
----------------------------------------------------  ---------------------------
Total                                      66.500ns (65.973ns logic, 0.527ns rout
                                                    (99.2% logic, 0.8% route)
```

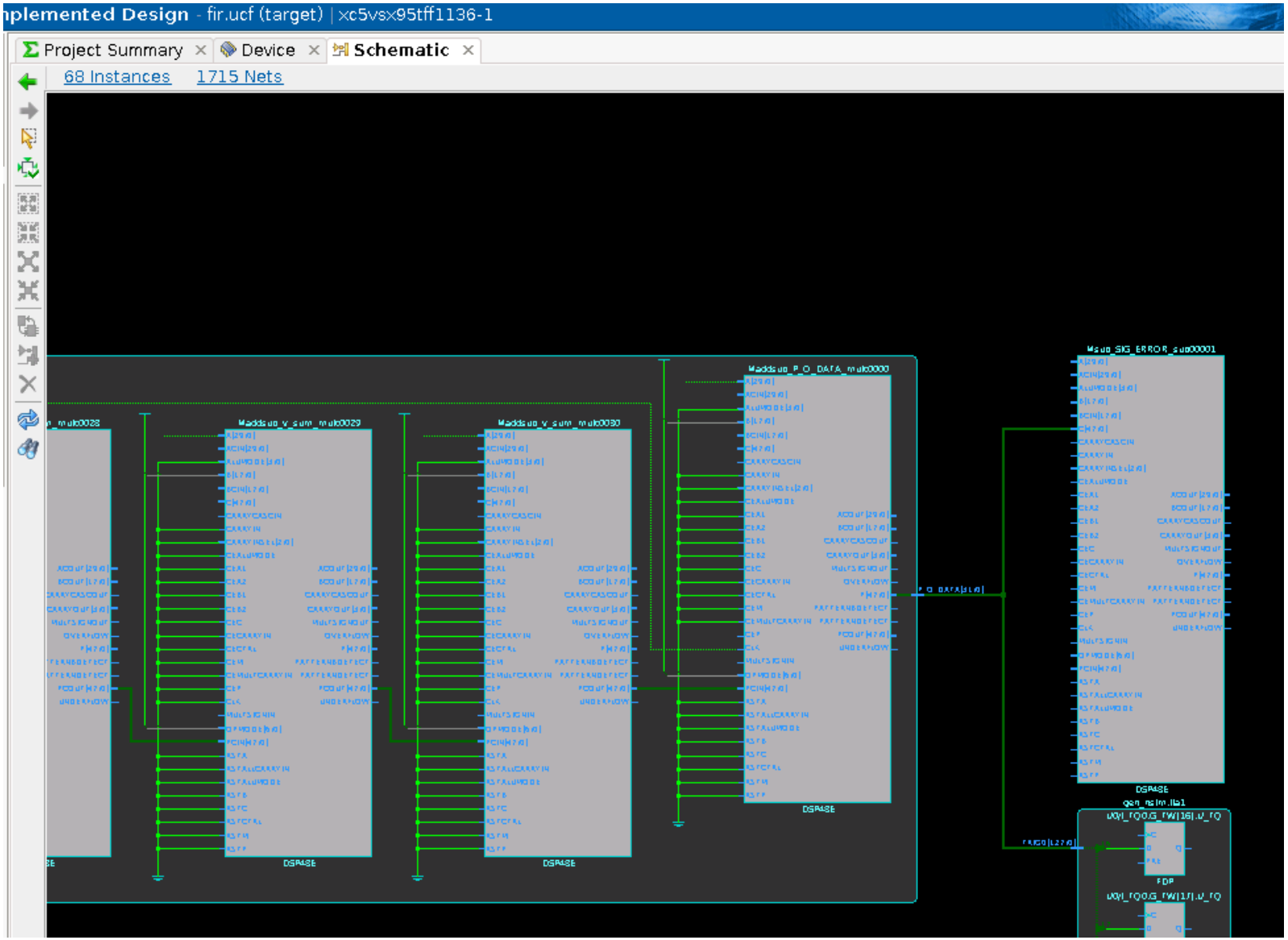
Timing Analysis (5)



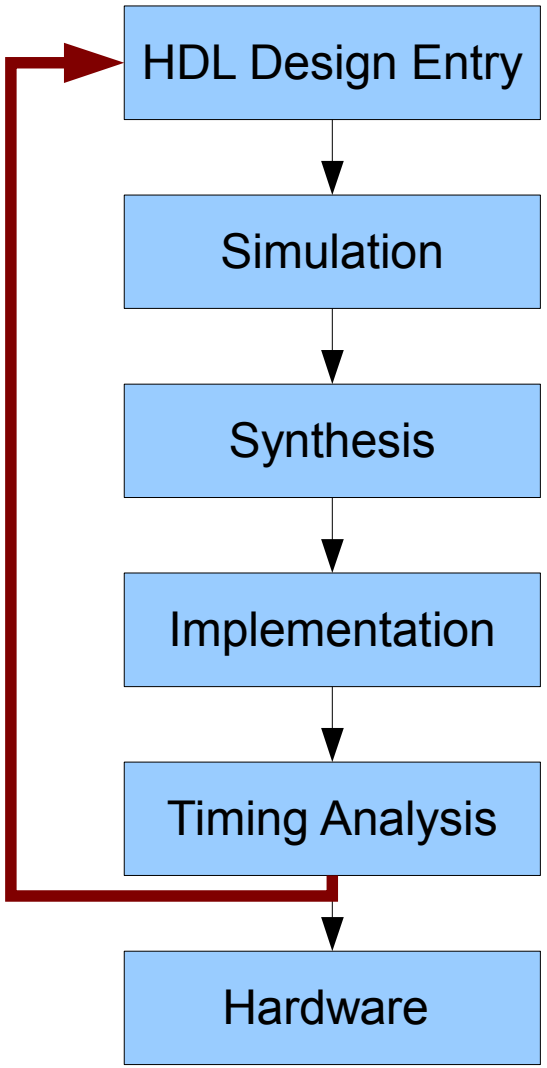
Timing Analysis (6)



Timing Analysis (7)



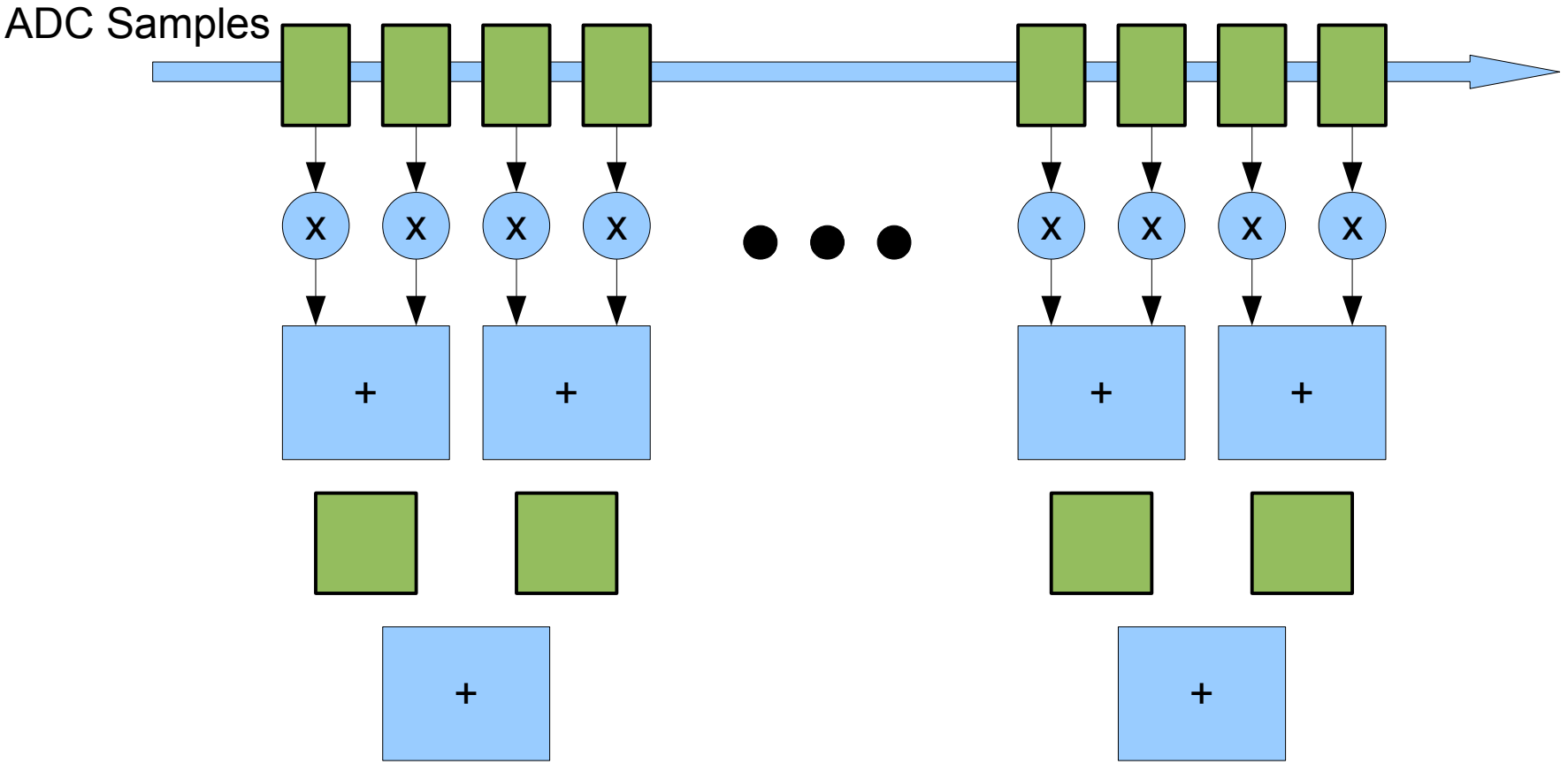
How to solve timing problems ?



If timing violation is not big (range of 1/10 of nanosecond) it may be enough to change optimization options or rerun implementation several times.

In case of our violation (~56 ns) the only solution is to go back to design entry stage.

FIR – Final Structure



P_O_DATA Register

32-input adder has been implemented as a cascade of 2 input adders (5 levels – 5 clock cycles of additional delay)

Chipscope Pro for HW Verification

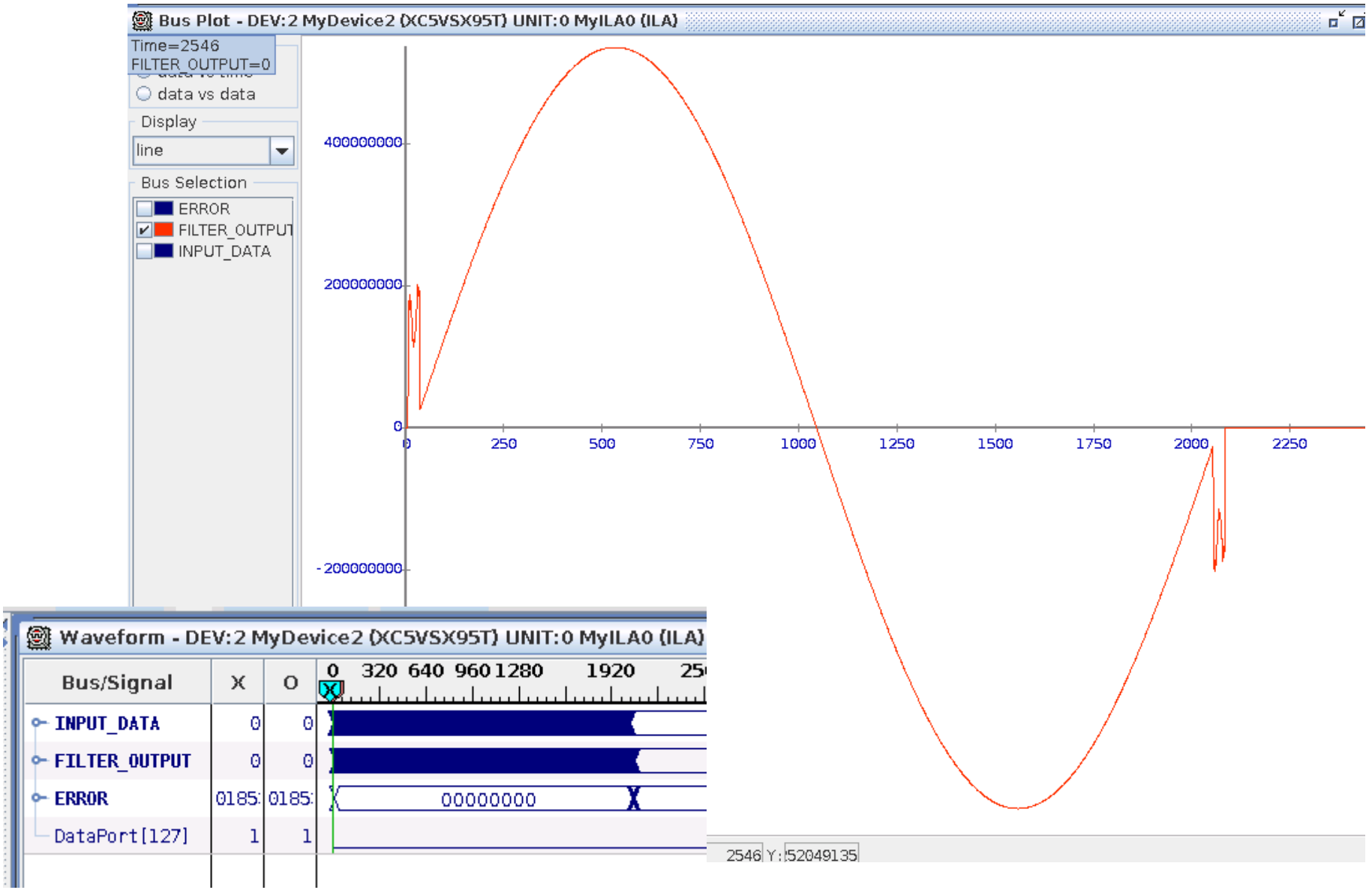
Internal Logic Analyzer – its functionality can be compared to big and expensive logic analyzers, but it is much simple to use.

During design entry stage you connect signals you want to monitor.

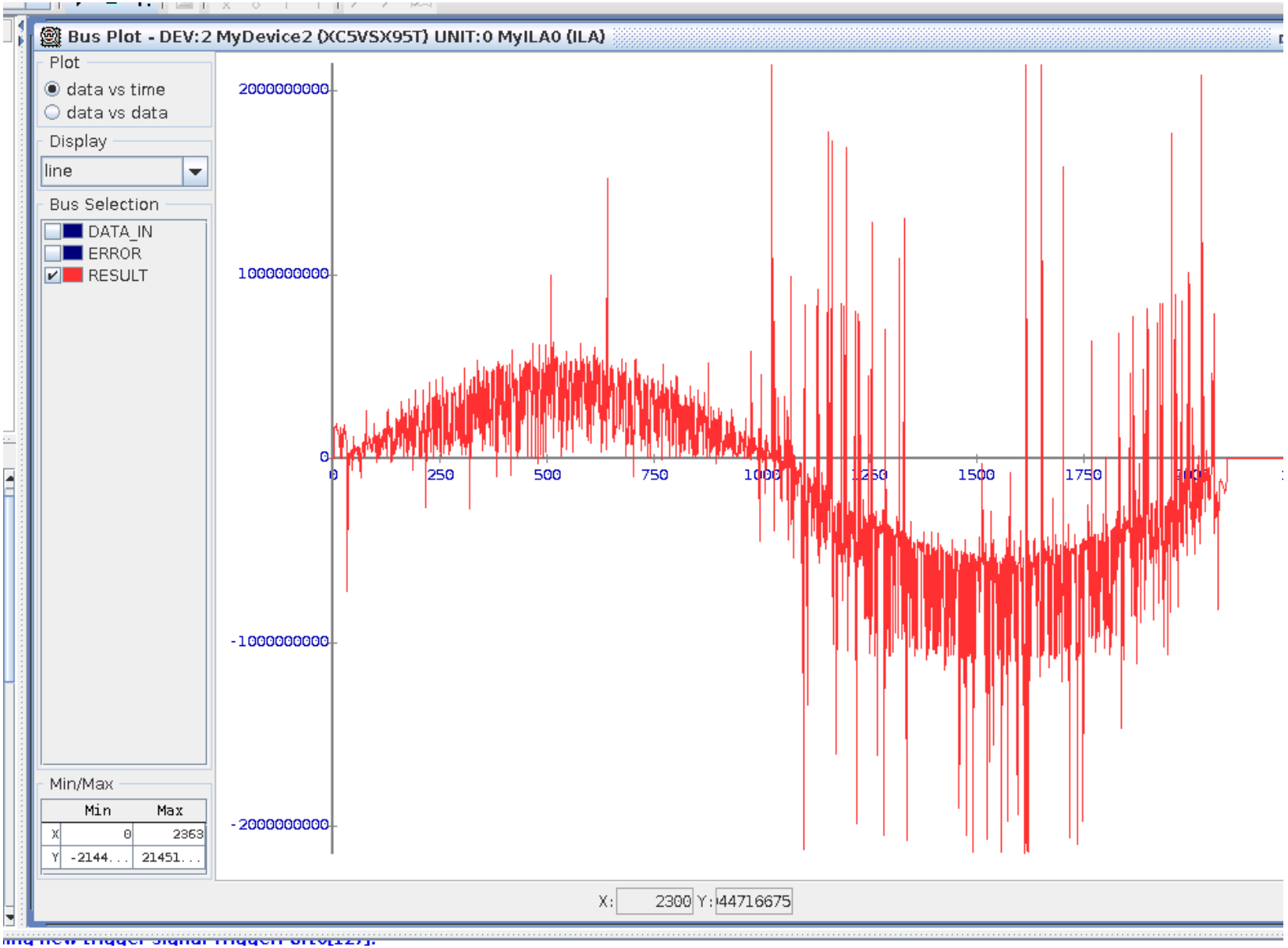
After FPGA is programmed you define trigger conditions and examine your data :)

Be careful – it occupies resources and can change circuit behavior especially when timing constraints are defined in a wrong way.

Chipscope Pro for HW Verification



Chipscope Pro for HW Verification



Advanced Issues

- More timing constraints (OFFSET, multi-cycle paths)
- Source synchronous data transfers
- Multiple clock domains

Thank You for Your Attention

Files can be found at:
http://tesla.desy.de/~wjalmuzn/llrf11.tgz