



# ЗНАКОМСТВО С САПР. СОЗДАЕМ СВОЙ ПРОЕКТ

**Проектирование цифровой  
техники с применением ПЛИС  
и аппаратного языка разработки  
System Verilog**

**Н. Г. Зайцев**

Кандидат технических наук, преподаватель кафедры КИПР ТУСУР,  
начальник сектора цифровой электроники ООО «ЛЭМЗ-Т»

**М. В. Кулешов**

Ведущий инженер-электроник ООО «ЛЭМЗ-Т»

**Я. В. Непомнящих**

Ведущий инженер-программист ООО «ЛЭМЗ-Т»

# Знакомство с САПР. Создаем свой проект



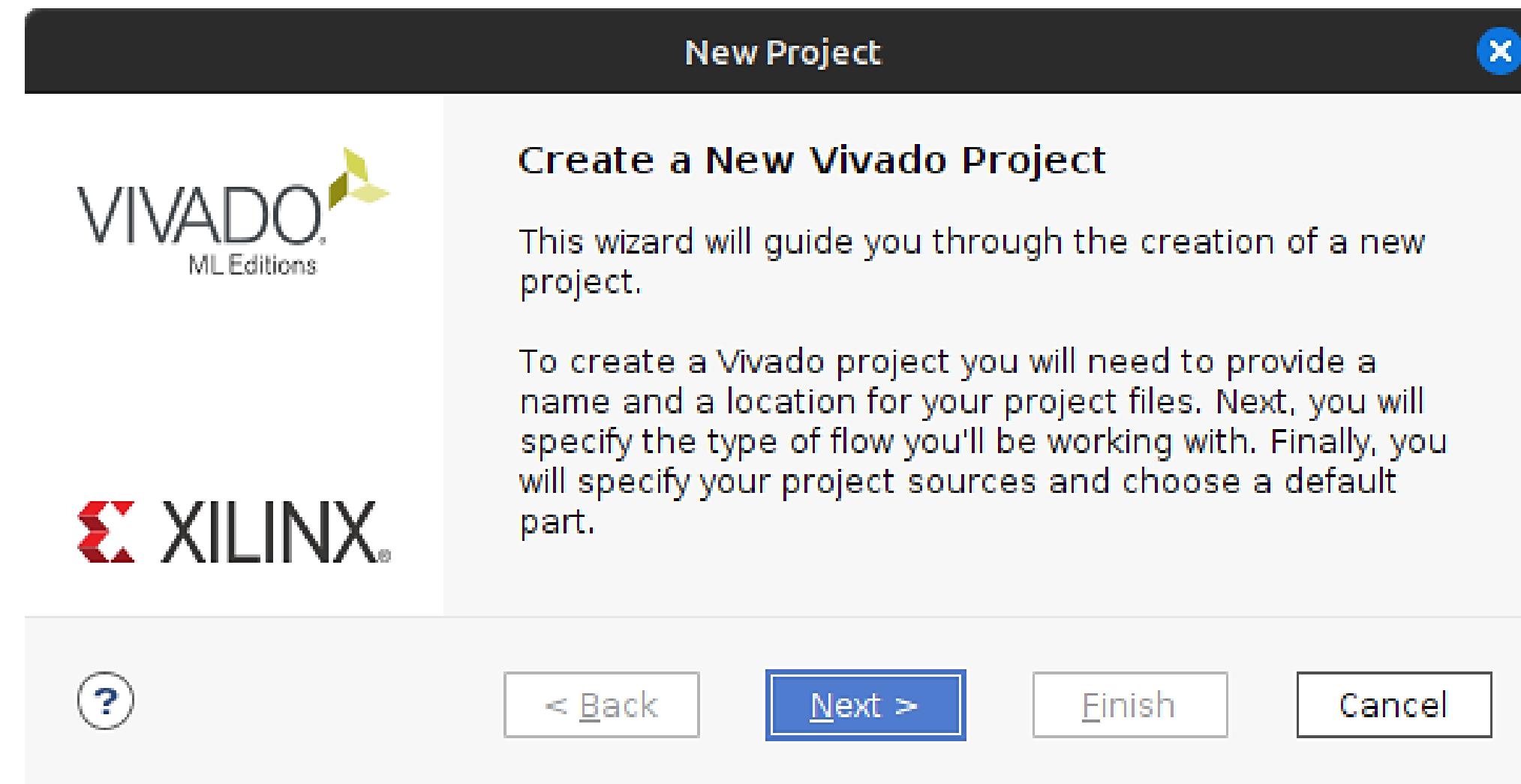
## Quick Start

[Create Project >](#)

[Open Project >](#)

[Open Example Project >](#)

# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект



New Project X

**Project Name**

Enter a name for your project and specify a directory where the project data files will be stored.



Project name:  X

Project location:  X ...

Create project subdirectory

Project will be created at: /home/neya/tmp/project\_2

? < Back Next > Finish Cancel

# Знакомство с САПР. Создаем свой проект



New Project X

**Project Type**  
Specify the type of project to create.



RTL Project  
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.  
 Do not specify sources at this time  
 Project is an extensible Vitis platform

Post-synthesis Project  
You will be able to add sources, view device resources, run design analysis, planning and implementation.  
 Do not specify sources at this time

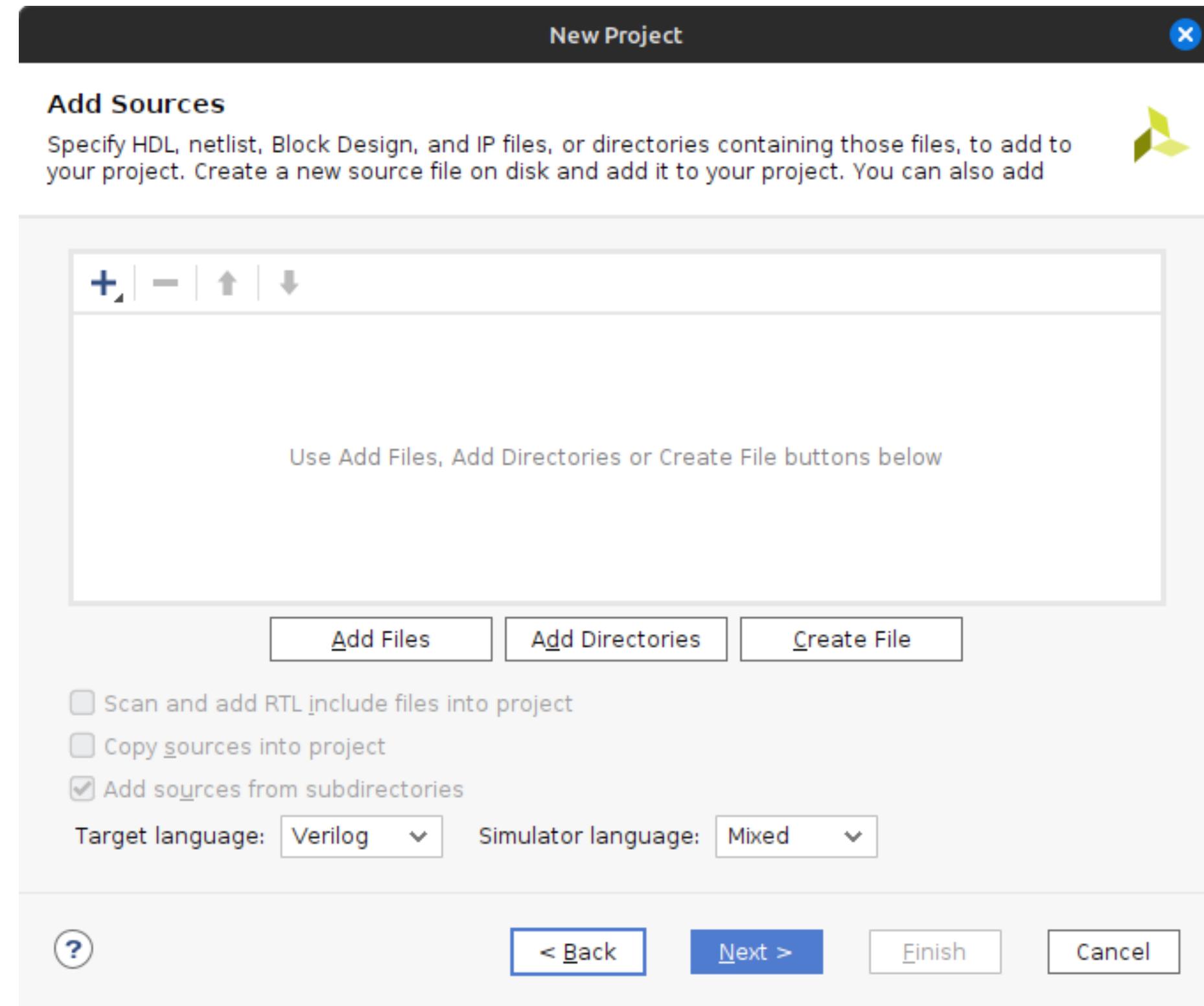
I/O Planning Project  
Do not specify design sources. You will be able to view part/package resources.

Imported Project  
Create a Vivado project from a Synplify Project File.

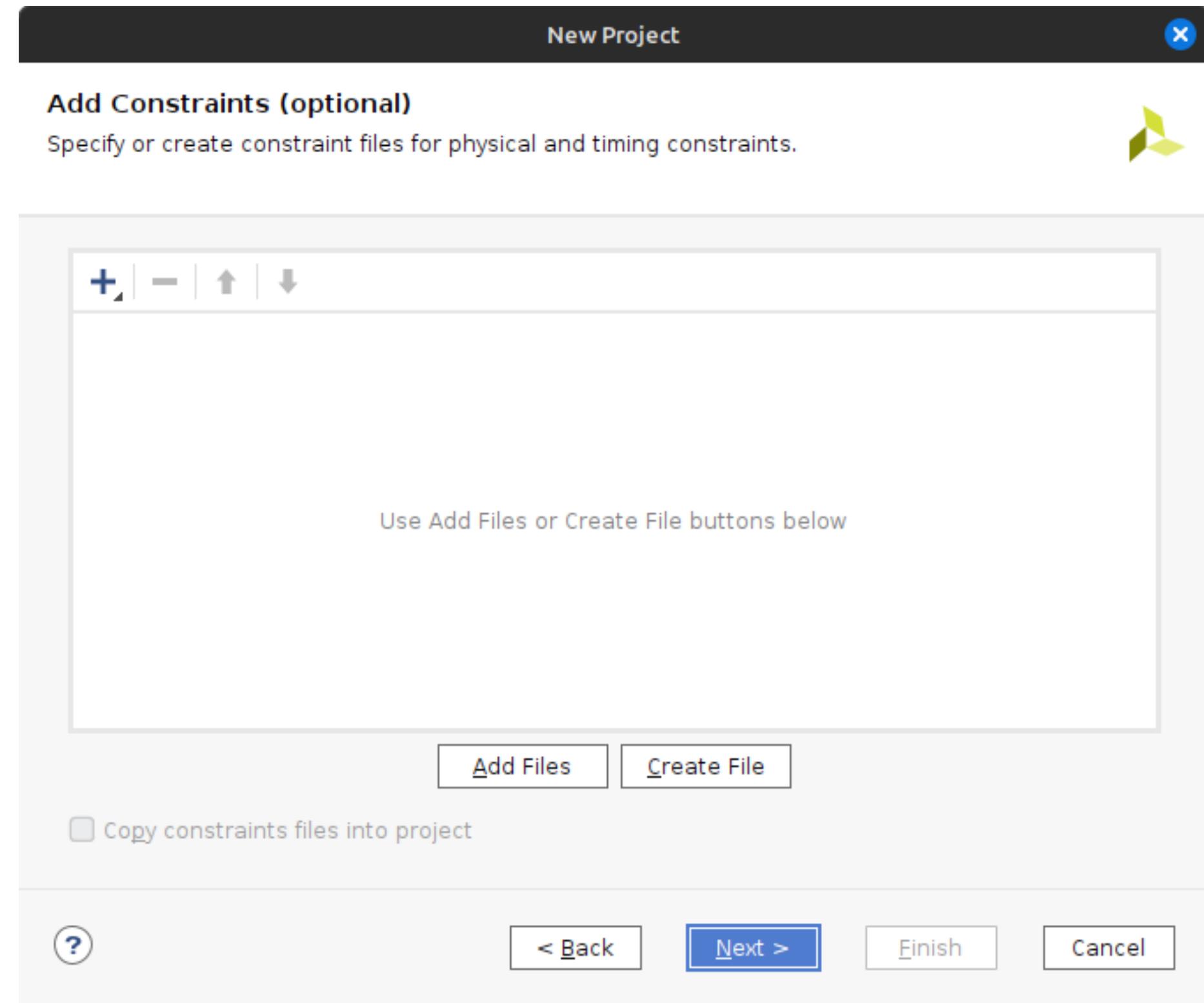
Example Project  
Create a new Vivado project from a predefined template.

? < Back Next > Finish Cancel

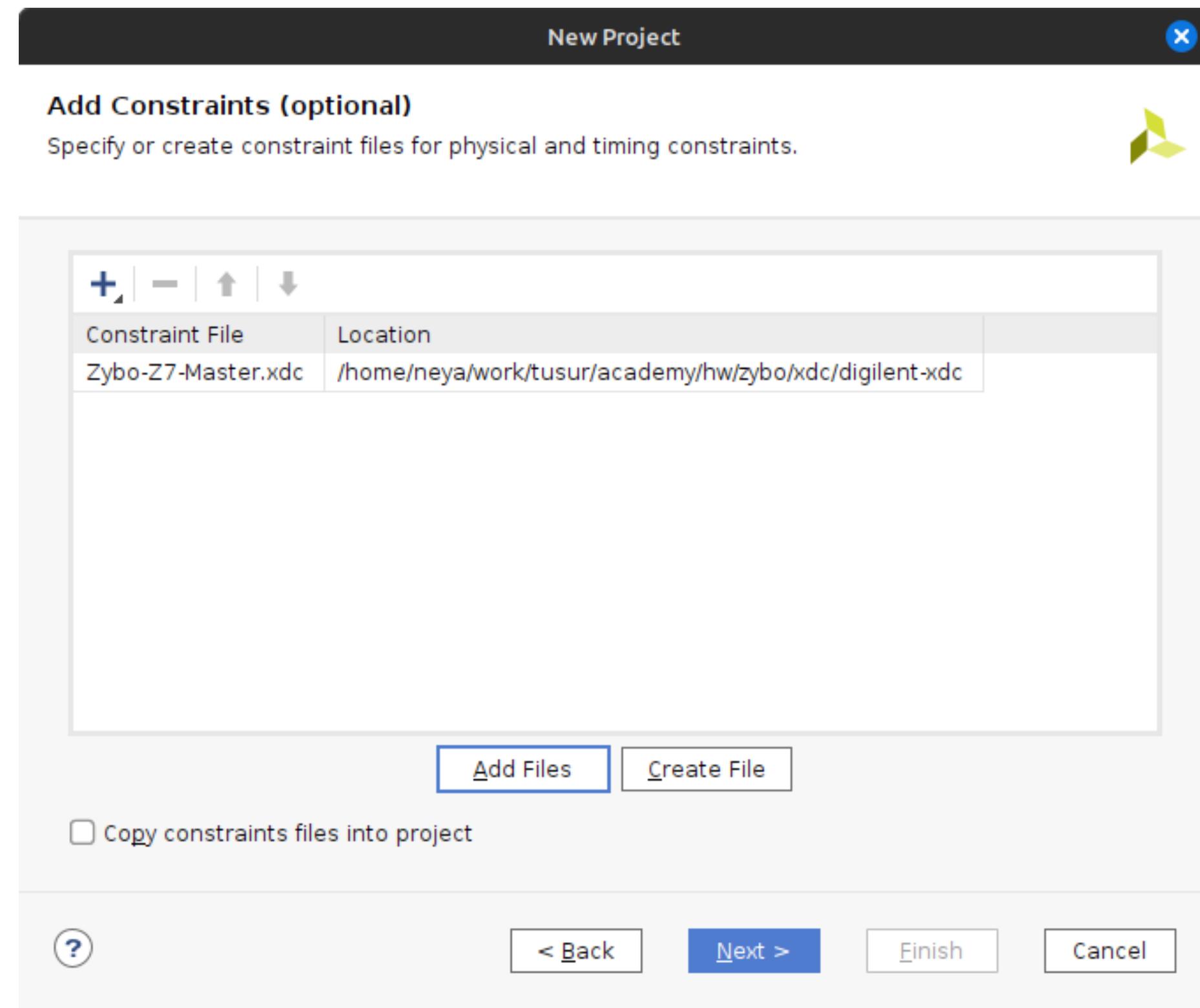
# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект



New Project

Default Part

Choose a default Xilinx part or board for your project.

Parts | Boards

To fetch the latest available boards from git repository, click on 'Refresh' button. [Dismiss](#)

Reset All Filters

Vendor: All Name: All Board Rev: Latest

Display Name	Preview	Status	Vendor	File Version	Part
Artix-7 AC701 Evaluation Platform Add Companion Card <a href="#">Connections</a>		Installed	xilinx.com	1.4	xc7a200tfg676
Kria K26C SOM Add Companion Card <a href="#">Connections</a>		Installed	xilinx.com	1.4	Commercial tem
Kria K26I SOM Add Companion Card <a href="#">Connections</a>		Installed	xilinx.com	1.4	Industrial tempe
Kintex-7 KC705 Evaluation Platform Add Companion Card <a href="#">Connections</a>		Installed	xilinx.com	1.6	xc7k325tffg900-
Kintex-UltraScale KCU105 Evaluation Platform Add Companion Card <a href="#">Connections</a>		Installed	xilinx.com	1.7	xciku040-fvval115

Refresh

< Back Next > Finish Cancel

# Знакомство с САПР. Создаем свой проект



New Project X

**Default Part**  
Choose a default Xilinx part or board for your project.

**Parts | Boards**

To fetch the latest available boards from git repository, click on 'Refresh' button. [Dismiss](#)

[Reset All Filters](#)

Vendor: All Name: All Board Rev: Latest

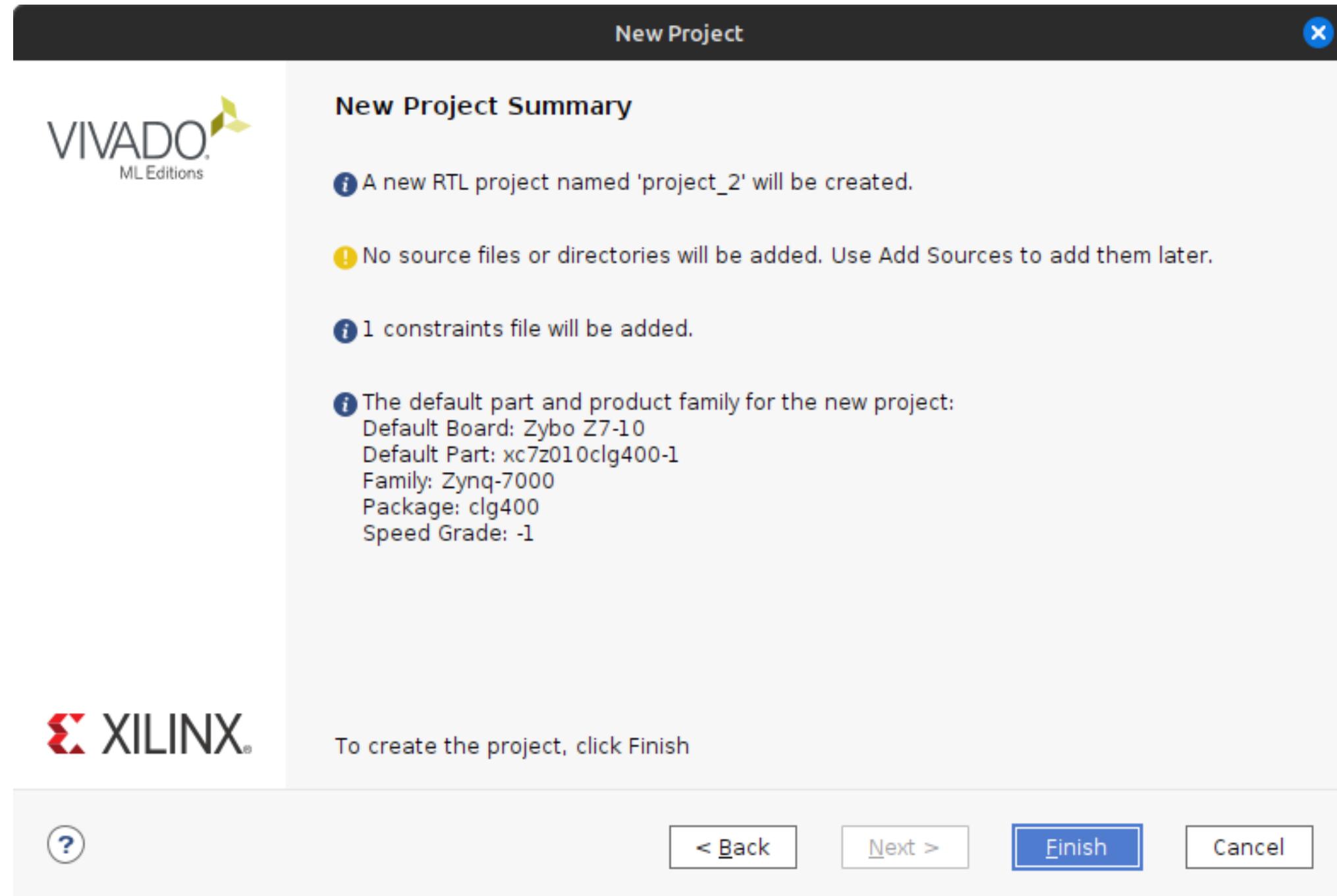
Search:  (3 matches)

Display Name	Preview	Status	Vendor	File Version	Part
Zybo Z7-10		Installed	diligentinc.com	1.2	xc7z010clg400-1
Zybo Z7-20		Installed	diligentinc.com	1.2	xc7z020clg400-1
Zybo		Installed	diligentinc.com	2.0	xc7z010clg400-1

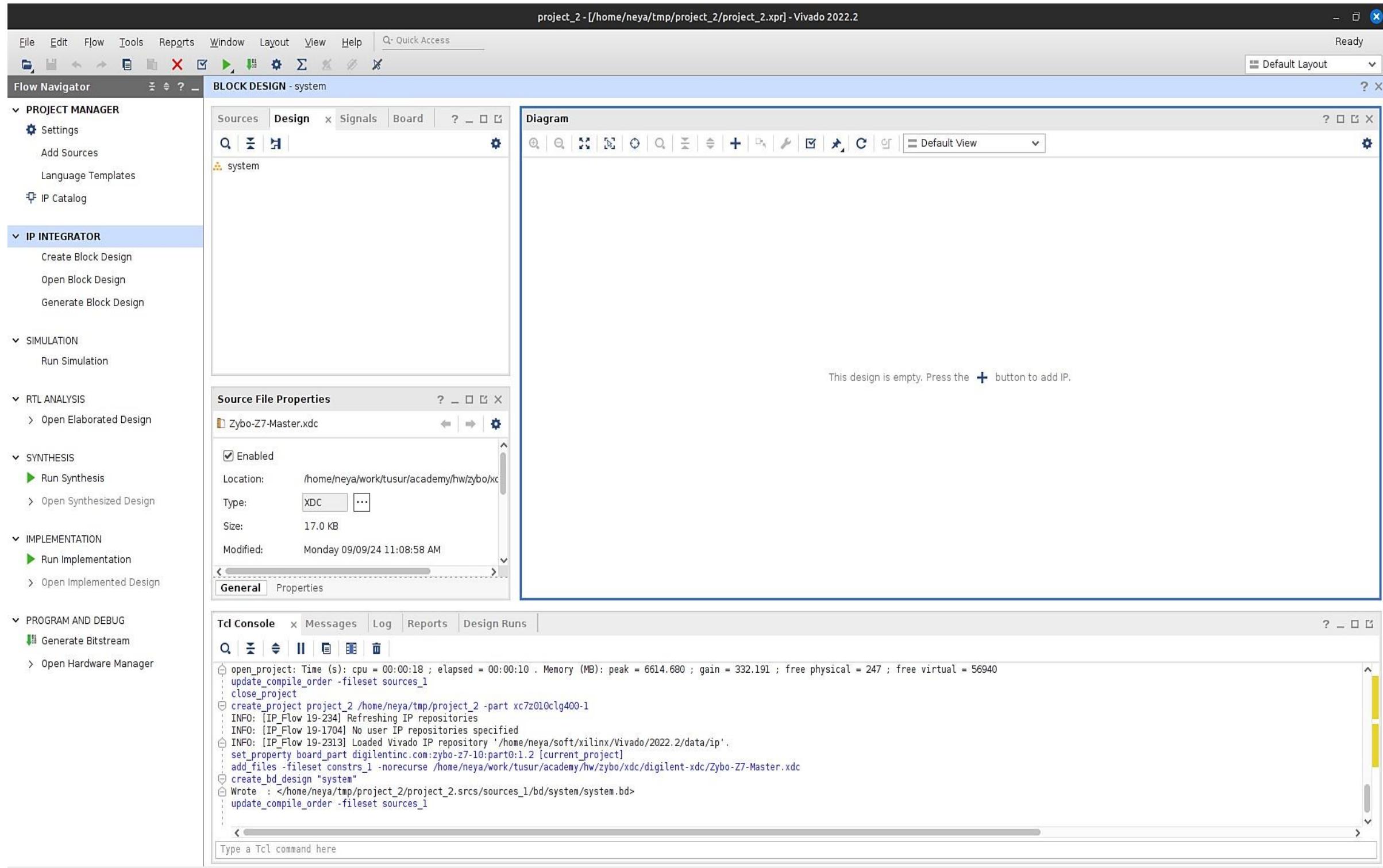
[Refresh](#)

[?](#) < Back [Next >](#) [Finish](#) [Cancel](#)

# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект



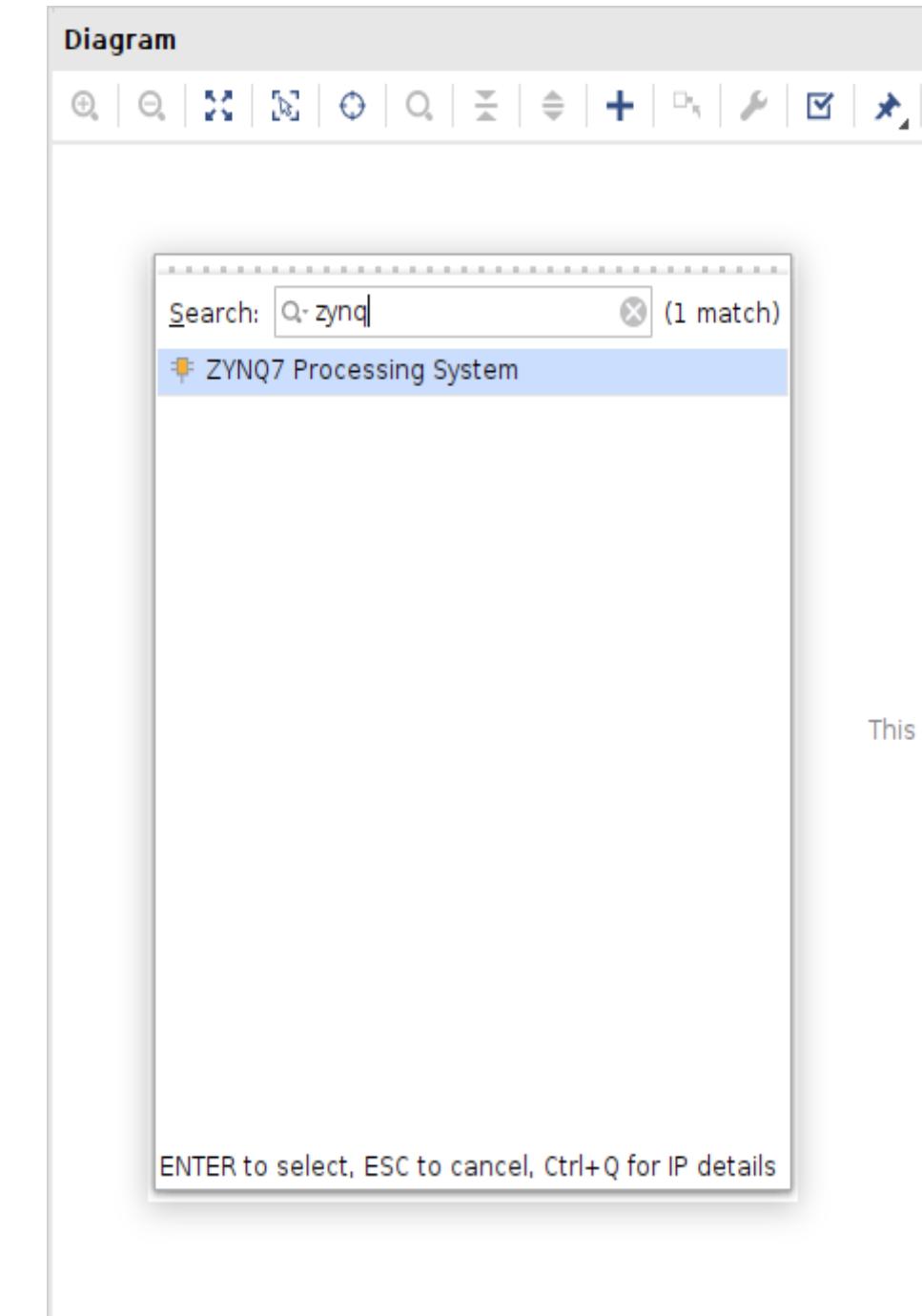
# Знакомство с САПР. Создаем свой проект



```
Zybo-Z7-Master.xdc
/home/neya/work/tusur/academy/hw/zybo/xdc/digilent-xdc/Zybo-Z7-Master.xdc
Q | F | < | > | X | D | // | E | ? |
1 ## This file is a general .xdc for the Zybo Z7 Rev. B
2 ## It is compatible with the Zybo Z7-20 and Zybo Z7-10
3 ## To use it in a project:
4 ## - uncomment the lines corresponding to used pins
5 ## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project
6
7 ##Clock signal
8 #set_property -dict { PACKAGE_PIN K17 IOSTANDARD LVCMS33 } [get_ports { sysclk }]; #IO_L12P_T1_MRCC_35 Sch=sysclk
9 #create_clock -add -name sys_clk_pin -period 8.00 -waveform {0 4} [get_ports { sysclk }];
10
11
12 ##Switches
13 #set_property -dict { PACKAGE_PIN G15 IOSTANDARD LVCMS33 } [get_ports { sw[0] }]; #IO_L19N_T3_VREF_35 Sch=sw[0]
14 #set_property -dict { PACKAGE_PIN P15 IOSTANDARD LVCMS33 } [get_ports { sw[1] }]; #IO_L24P_T3_34 Sch=sw[1]
15 #set_property -dict { PACKAGE_PIN W13 IOSTANDARD LVCMS33 } [get_ports { sw[2] }]; #IO_L4N_T0_34 Sch=sw[2]
16 #set_property -dict { PACKAGE_PIN T16 IOSTANDARD LVCMS33 } [get_ports { sw[3] }]; #IO_L9P_T1_DQS_34 Sch=sw[3]
17
18
19 ##Buttons
20 set_property -dict { PACKAGE_PIN K18 IOSTANDARD LVCMS33 } [get_ports { btn_tri_i[0] }]; #IO_L12N_T1_MRCC_35 Sch=btn[0]
21 set_property -dict { PACKAGE_PIN P16 IOSTANDARD LVCMS33 } [get_ports { btn_tri_i[1] }]; #IO_L24N_T3_34 Sch=btn[1]
22 set_property -dict { PACKAGE_PIN K19 IOSTANDARD LVCMS33 } [get_ports { btn_tri_i[2] }]; #IO_L10P_T1_AD11P_35 Sch=btn[2]
23 set_property -dict { PACKAGE_PIN Y16 IOSTANDARD LVCMS33 } [get_ports { btn_tri_i[3] }]; #IO_L7P_T1_34 Sch=btn[3]
24
25
26 ##LEDs
27 set_property -dict { PACKAGE_PIN M14 IOSTANDARD LVCMS33 } [get_ports { led_tri_o[0] }]; #IO_L23P_T3_35 Sch=led[0]
28 set_property -dict { PACKAGE_PIN M15 IOSTANDARD LVCMS33 } [get_ports { led_tri_o[1] }]; #IO_L23N_T3_35 Sch=led[1]
29 set_property -dict { PACKAGE_PIN G14 IOSTANDARD LVCMS33 } [get_ports { led_tri_o[2] }]; #IO_0_35 Sch=led[2]
30 set_property -dict { PACKAGE_PIN D18 IOSTANDARD LVCMS33 } [get_ports { led_tri_o[3] }]; #IO_L3N_T0_DQS_AD1N_35 Sch=led[3]
31
32
33 ##RGB LED 5 (Zybo Z7-20 only)
34 #set_property -dict { PACKAGE_PIN Y11 IOSTANDARD LVCMS33 } [get_ports { led5_r }]; #IO_L18N_T2_13 Sch=led5_r
35 #set_property -dict { PACKAGE_PIN T5 IOSTANDARD LVCMS33 } [get_ports { led5_g }]; #IO_L19P_T3_13 Sch=led5_g
36 #set_property -dict { PACKAGE_PIN V12 IOSTANDARD LVCMS33 } [get_ports { led5_b }]; #IO_L19N_T3_13 Sch=led5_b
```

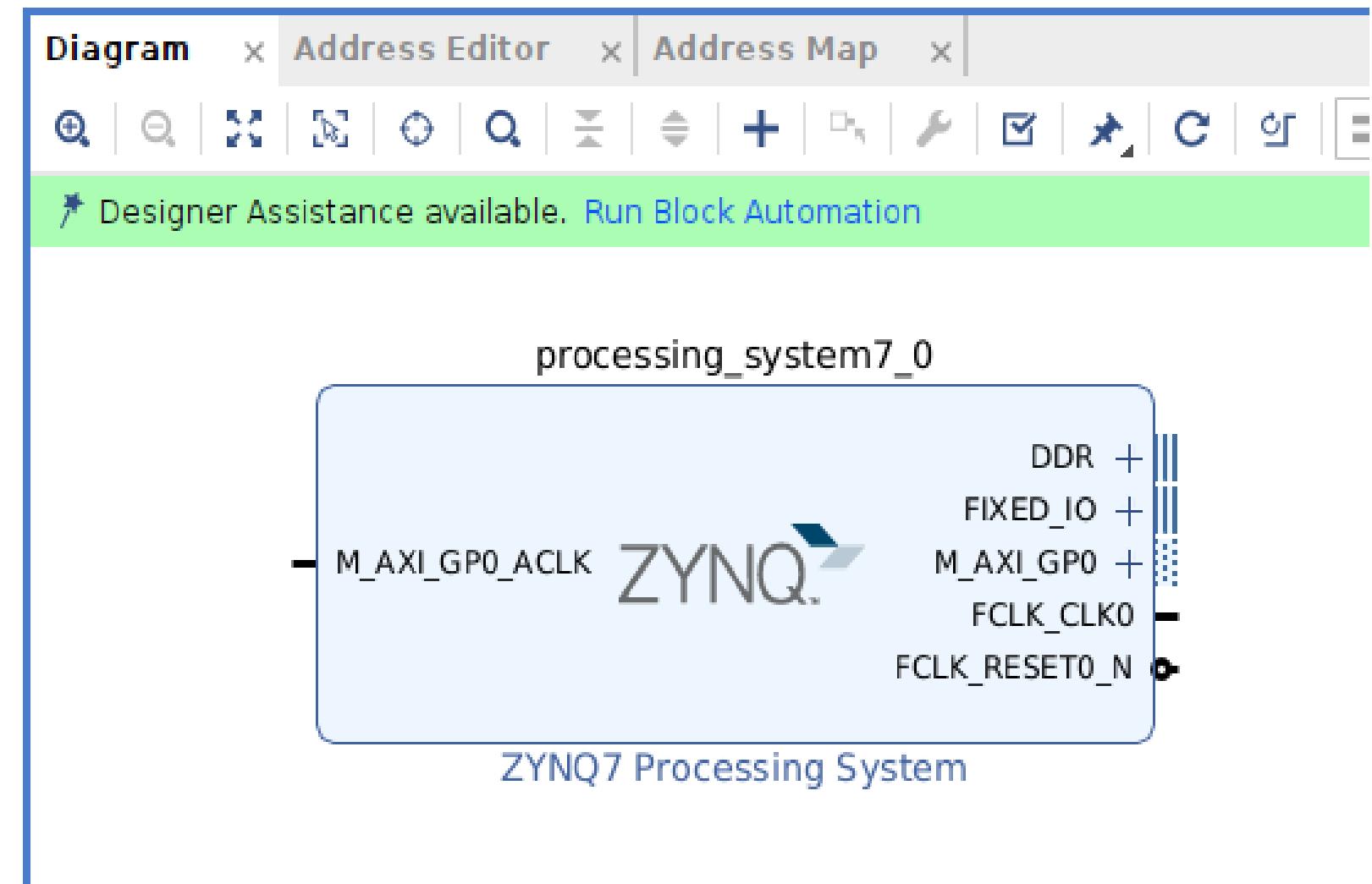
В проекте пока ничего нет кроме файла ограничений с описанием LED-индикаторов и клавиш.

# Знакомство с САПР. Создаем свой проект



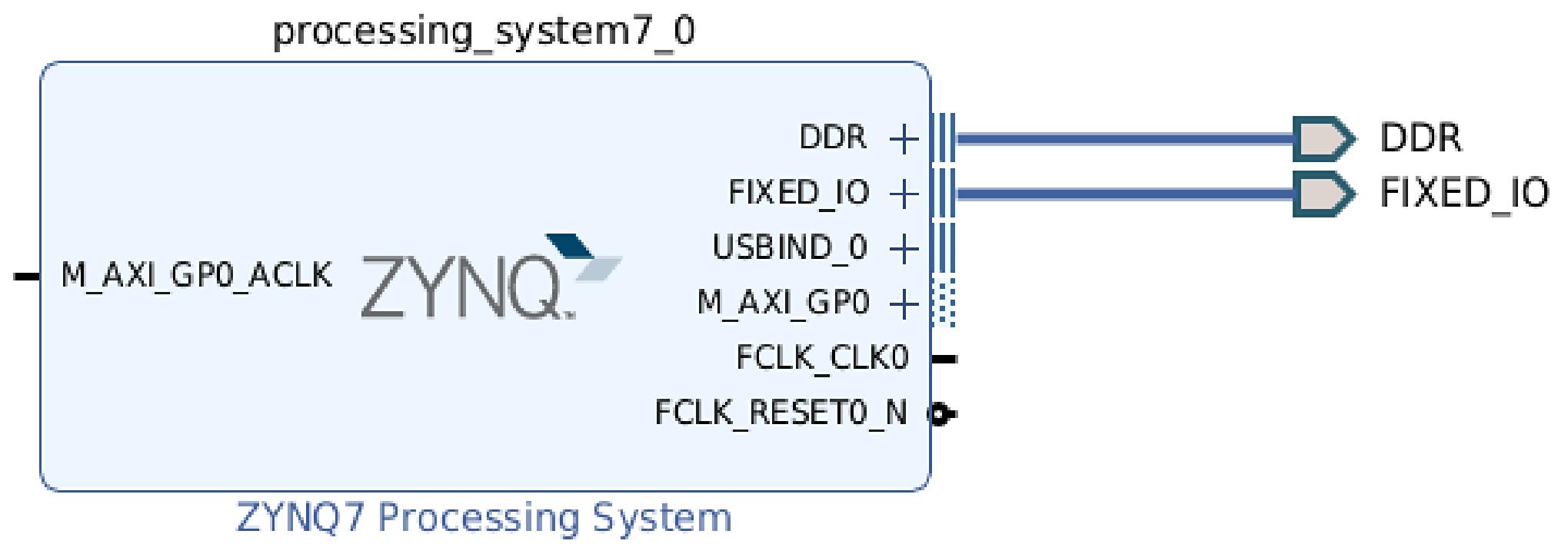
Добавляем описание процессорной системы и ее периферии.

# Знакомство с САПР. Создаем свой проект

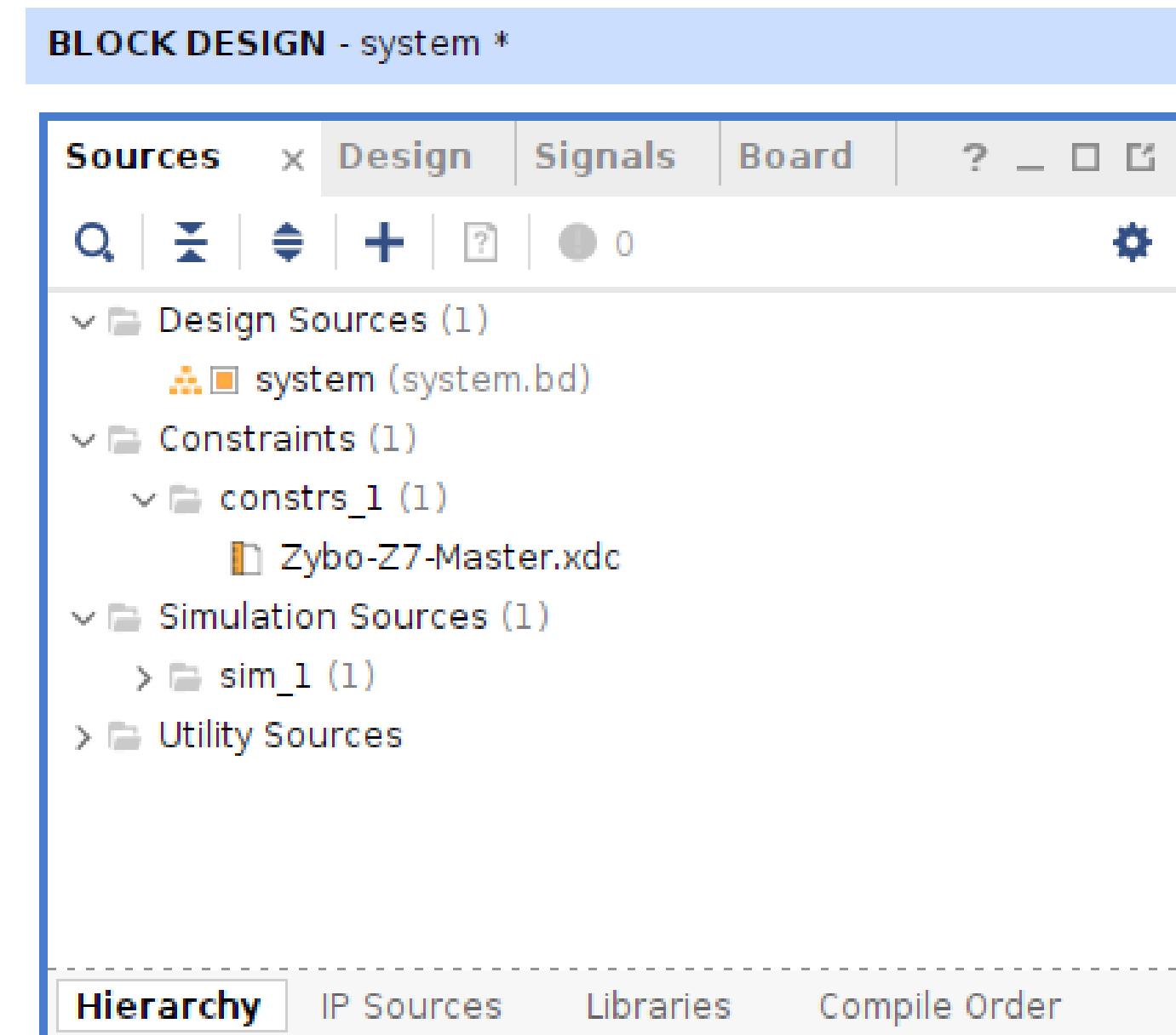


Пользуемся ПОМОЩНИКОМ.

# Знакомство с САПР. Создаем свой проект

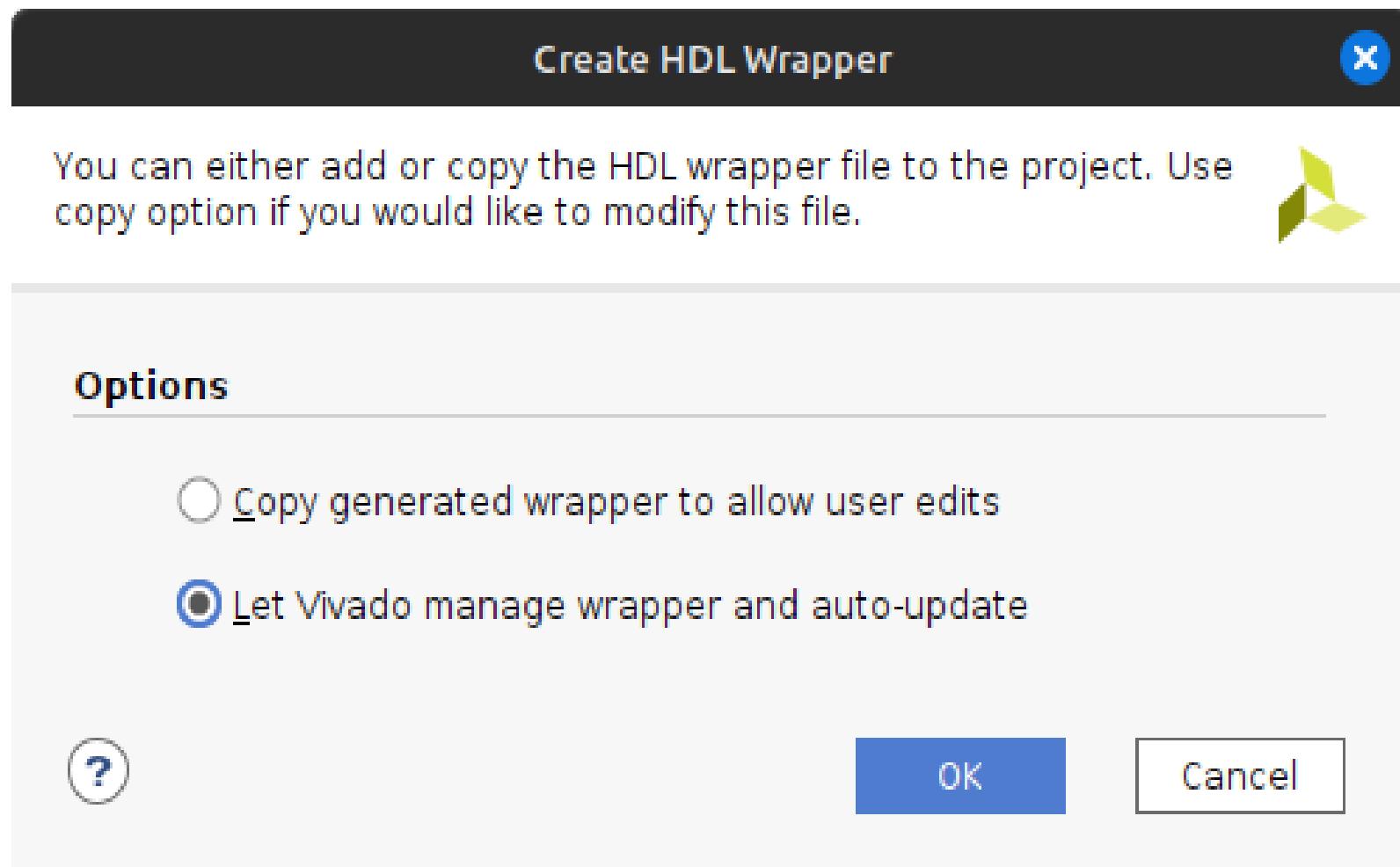


# Знакомство с САПР. Создаем свой проект

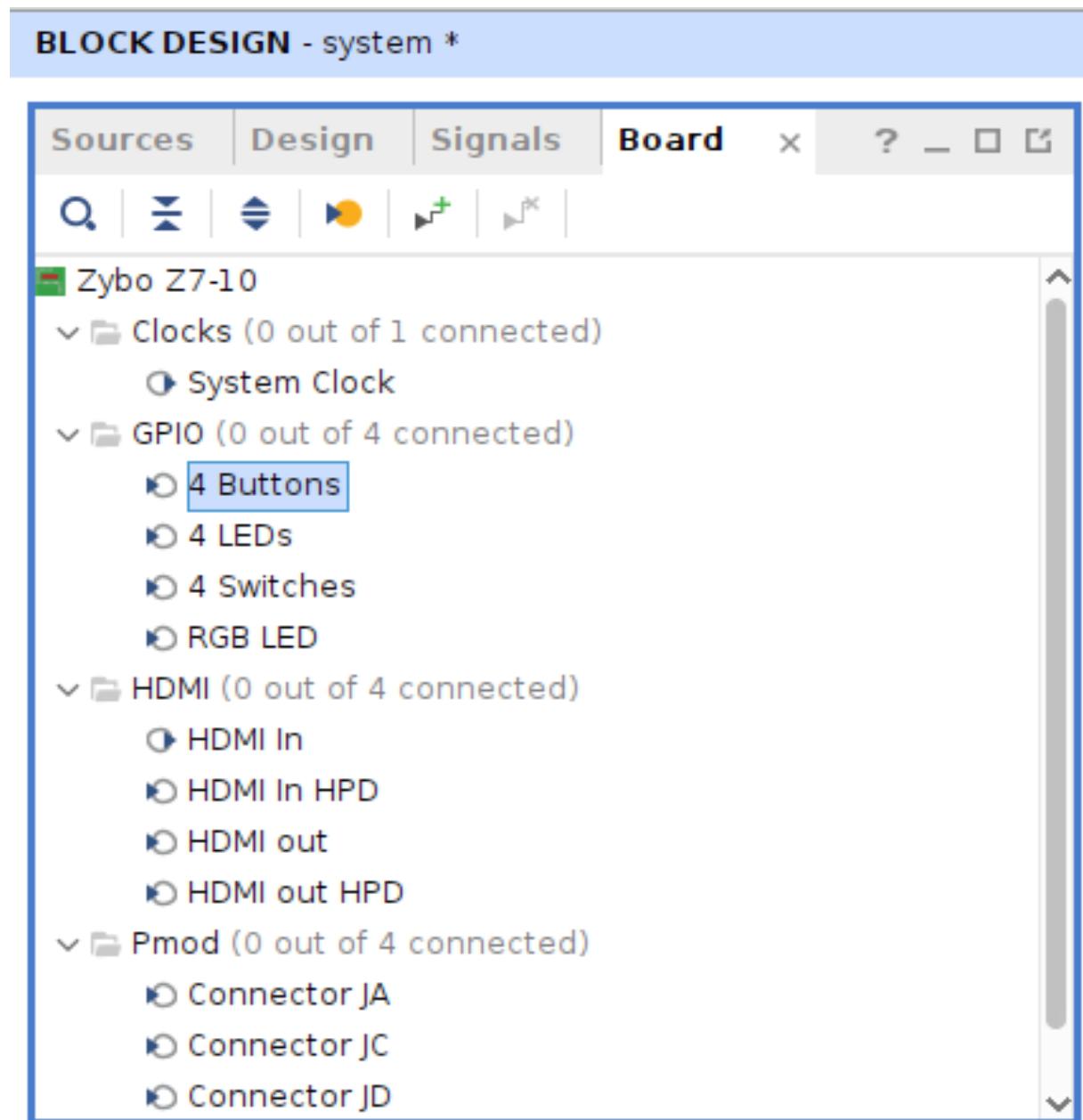


Создаем файл-обертку для Block Design.  
В контекстном меню на system.bd есть нужная команда.

# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект



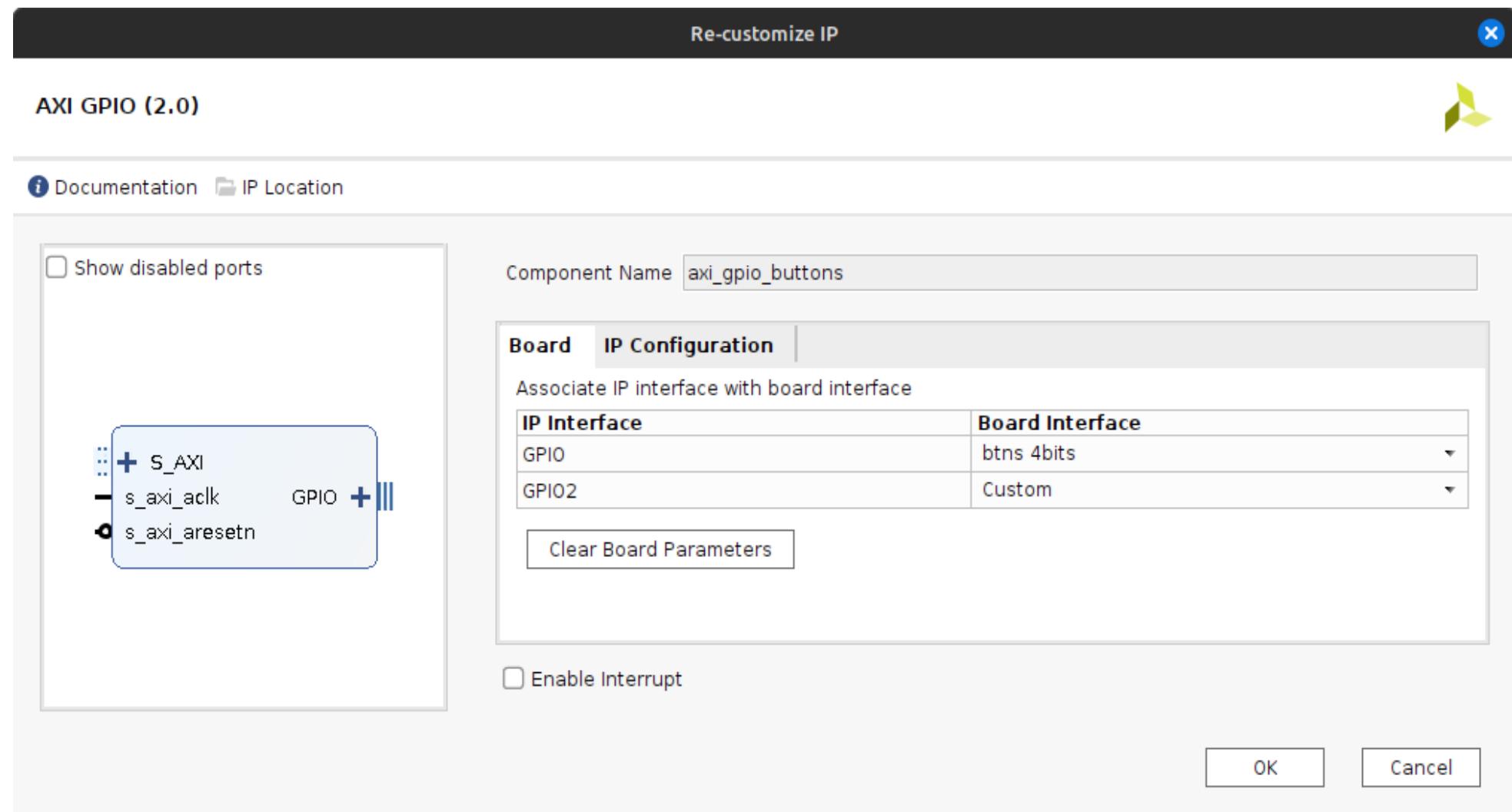
В контекстном меню выбираем Connect Board Component.

# Знакомство с САПР. Создаем свой проект



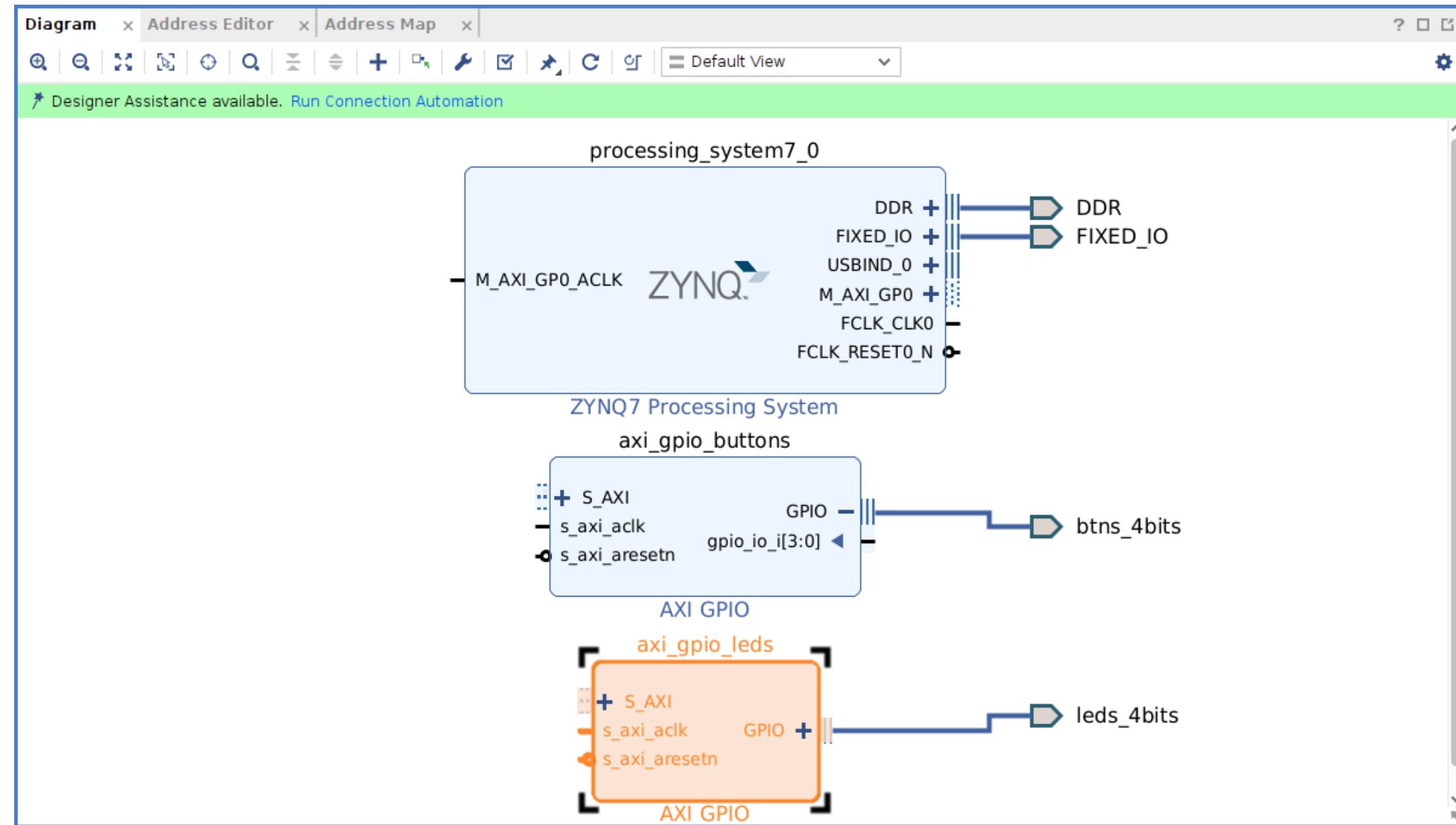
Добавлен компонент, позволяющий программе использовать и создавать пользовательские входы и выходы (сейчас они виртуальные).

# Знакомство с САПР. Создаем свой проект



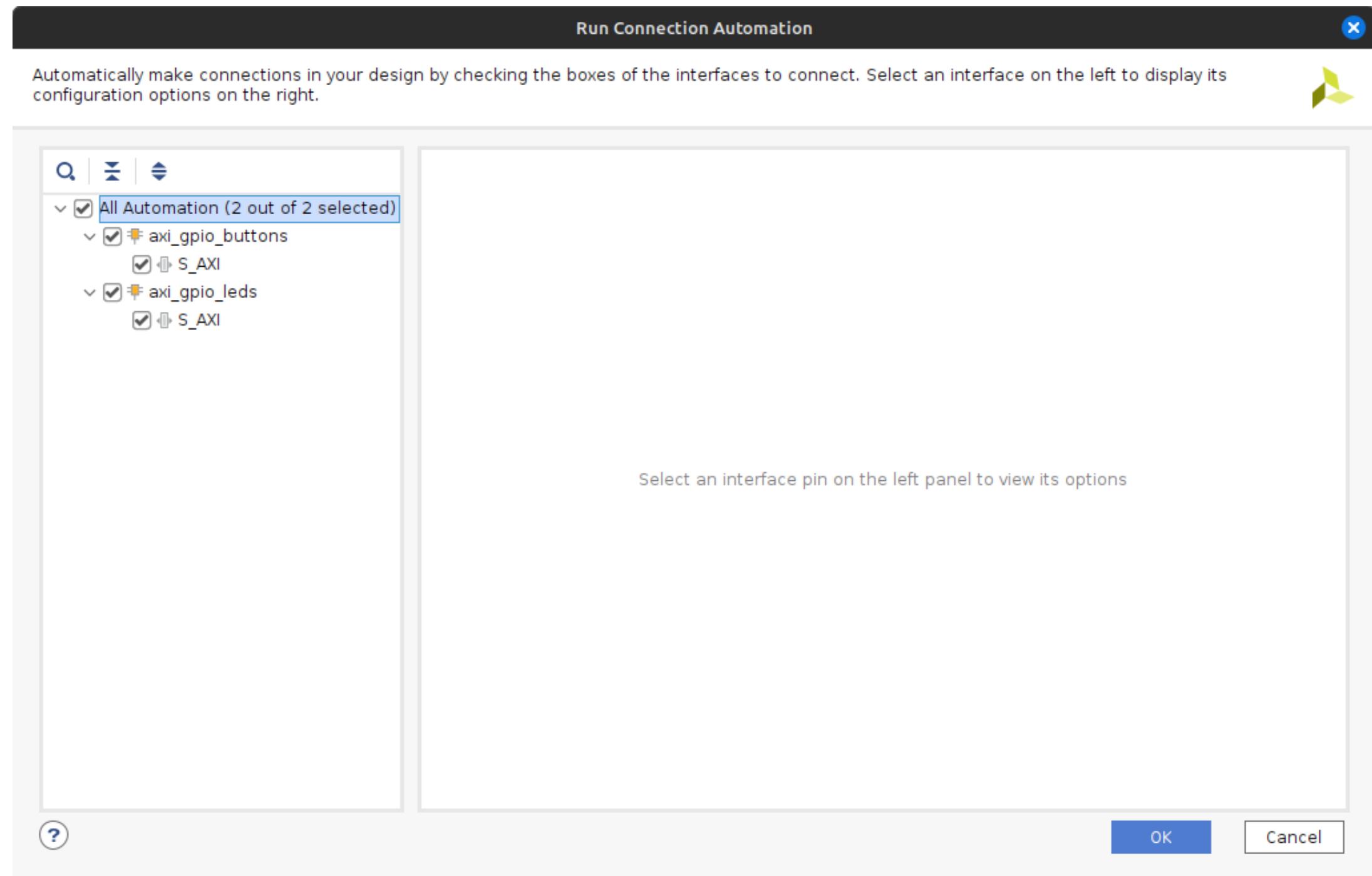
Настраивать пока ничего не будем.  
Вперед!

# Знакомство с САПР. Создаем свой проект

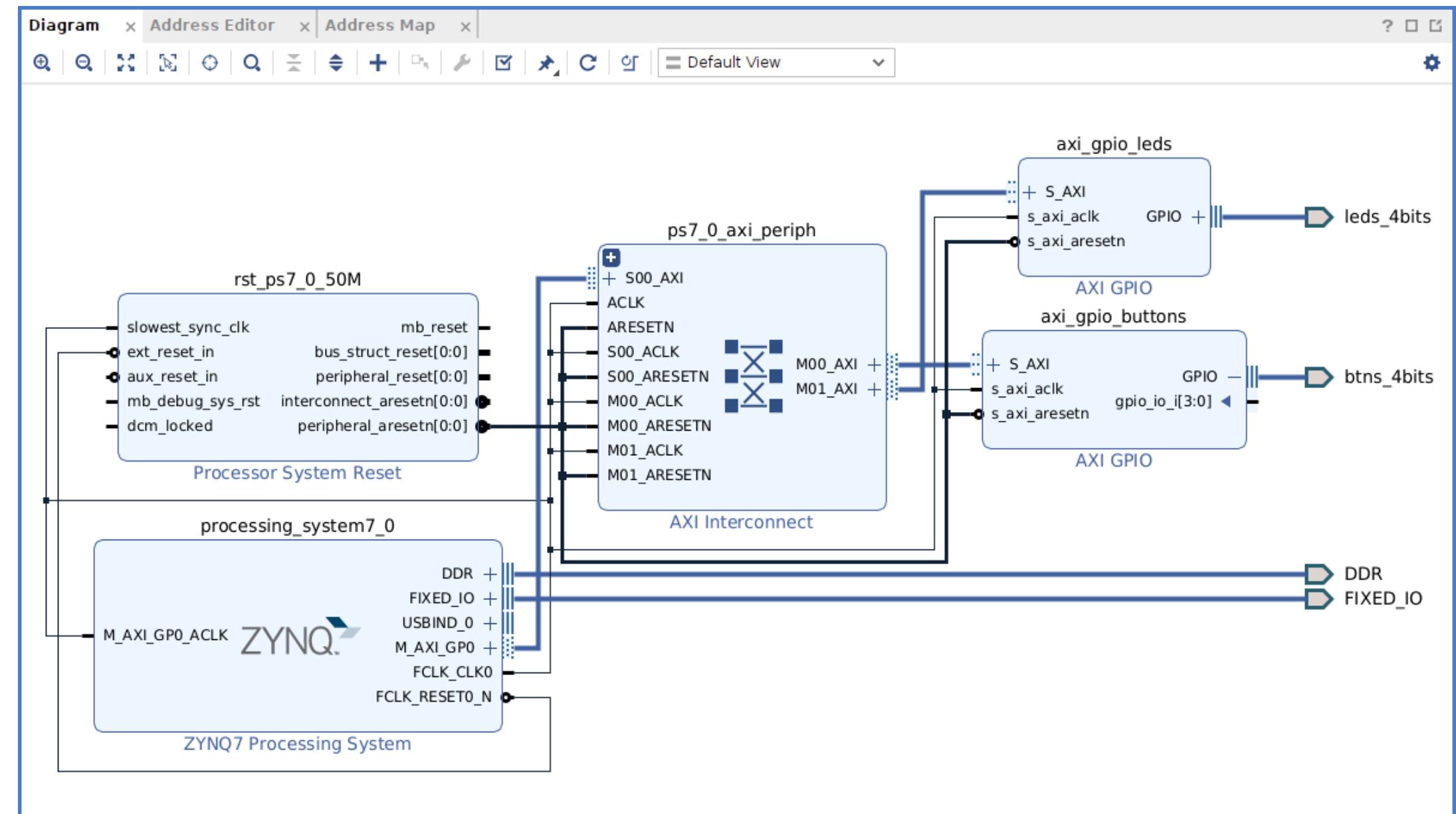


Снова ПОМОЩНИК в деле.

# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект



Итог наших созидательных движений.

# Знакомство с САПР. Создаем свой проект



```
36  inout [14:0]DDR_addr;
37  inout [2:0]DDR_ba;
38  inout DDR_cas_n;
39  inout DDR_ck_n;
40  inout DDR_ck_p;
41  inout DDR_cke;
42  inout DDR_cs_n;
43  inout [3:0]DDR_dm;
44  inout [31:0]DDR_dq;
45  inout [3:0]DDR_dqs_n;
46  inout [3:0]DDR_dqs_p;
47  inout DDR_odt;
48  inout DDR_ras_n;
49  inout DDR_reset_n;
50  inout DDR_we_n;
51  inout FIXED_IO_ddr_vrn;
52  inout FIXED_IO_ddr_vrp;
53  inout [53:0]FIXED_IO_mio;
54  inout FIXED_IO_ps_clk;
55  inout FIXED_IO_ps_porb;
56  inout FIXED_IO_ps_srstb;
57  input [3:0]btns_4bits_tri_i;
58  output [3:0]leds_4bits_tri_o;
--
```

Файл-обертка сформировался автоматически.  
Синтаксис - Verilog.  
Интерфейс соответствует Block Design.

# Знакомство с САПР. Создаем свой проект



```
#Buttons
set_property -dict { PACKAGE_PIN K18 IOSTANDARD LVCMOS33 } [get_ports { btms_4bits_tri_i[0] }]; #IO_L12N_T1_MRCC_35 Sch=btn[0]
set_property -dict { PACKAGE_PIN P16 IOSTANDARD LVCMOS33 } [get_ports { btms_4bits_tri_i[1] }]; #IO_L24N_T3_34 Sch=btn[1]
set_property -dict { PACKAGE_PIN K19 IOSTANDARD LVCMOS33 } [get_ports { btms_4bits_tri_i[2] }]; #IO_L10P_T1_AD11P_35 Sch=btn[2]
set_property -dict { PACKAGE_PIN Y16 IOSTANDARD LVCMOS33 } [get_ports { btms_4bits_tri_i[3] }]; #IO_L7P_T1_34 Sch=btn[3]

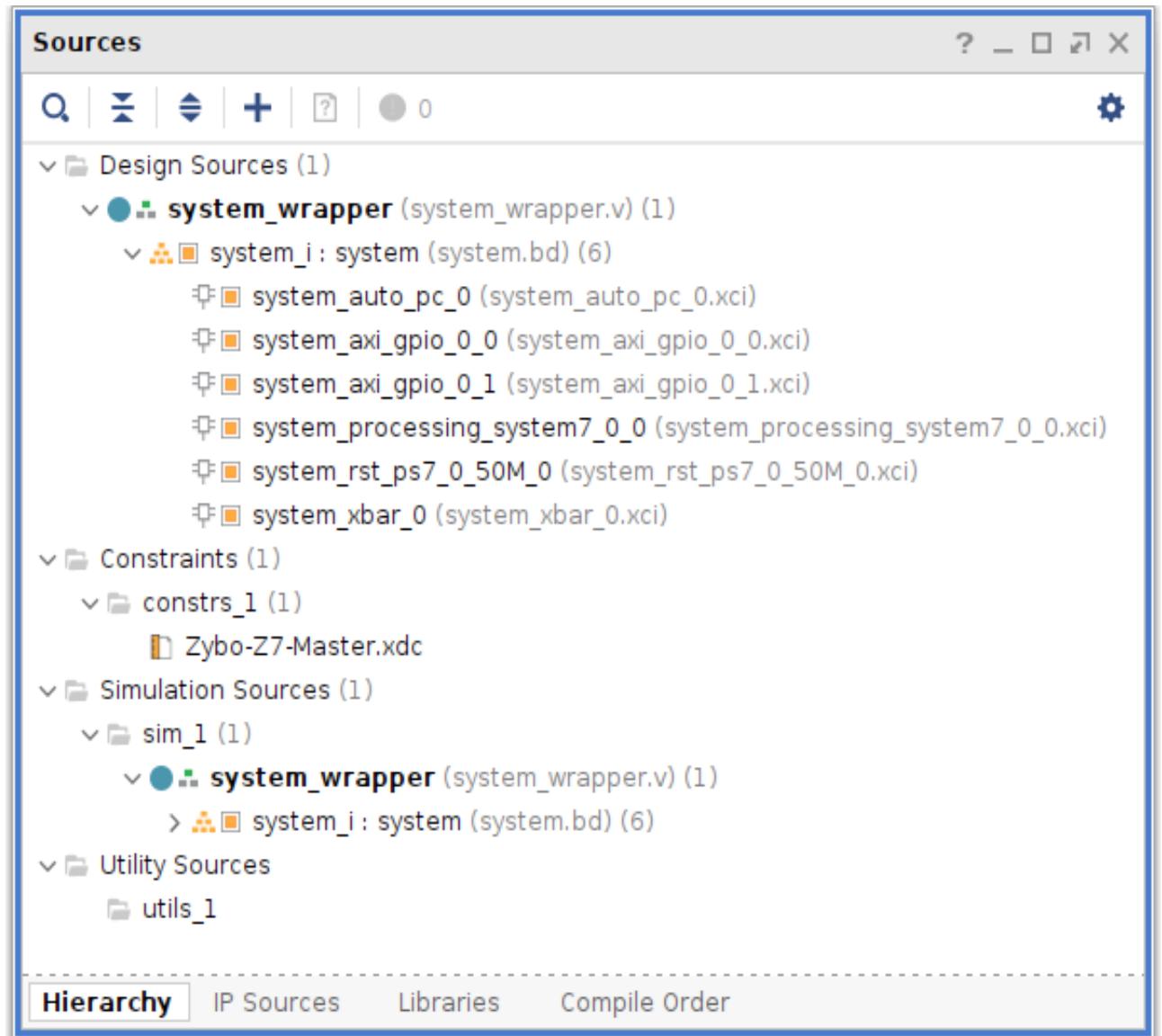
#LEDs
set_property -dict { PACKAGE_PIN M14 IOSTANDARD LVCMOS33 } [get_ports { leds_4bits_tri_o[0] }]; #IO_L23P_T3_35 Sch=led[0]
set_property -dict { PACKAGE_PIN M15 IOSTANDARD LVCMOS33 } [get_ports { leds_4bits_tri_o[1] }]; #IO_L23N_T3_35 Sch=led[1]
set_property -dict { PACKAGE_PIN G14 IOSTANDARD LVCMOS33 } [get_ports { leds_4bits_tri_o[2] }]; #IO_O_35 Sch=led[2]
set_property -dict { PACKAGE_PIN D18 IOSTANDARD LVCMOS33 } [get_ports { leds_4bits_tri_o[3] }]; #IO_L3N_T0_DQS_ADIN_35 Sch=led[3]
```

Зачем нам эти ограничения?

Здесь дружится логика наших бизнес-задач с холодным бездушным железом!

Разогреем его градусов до 70?

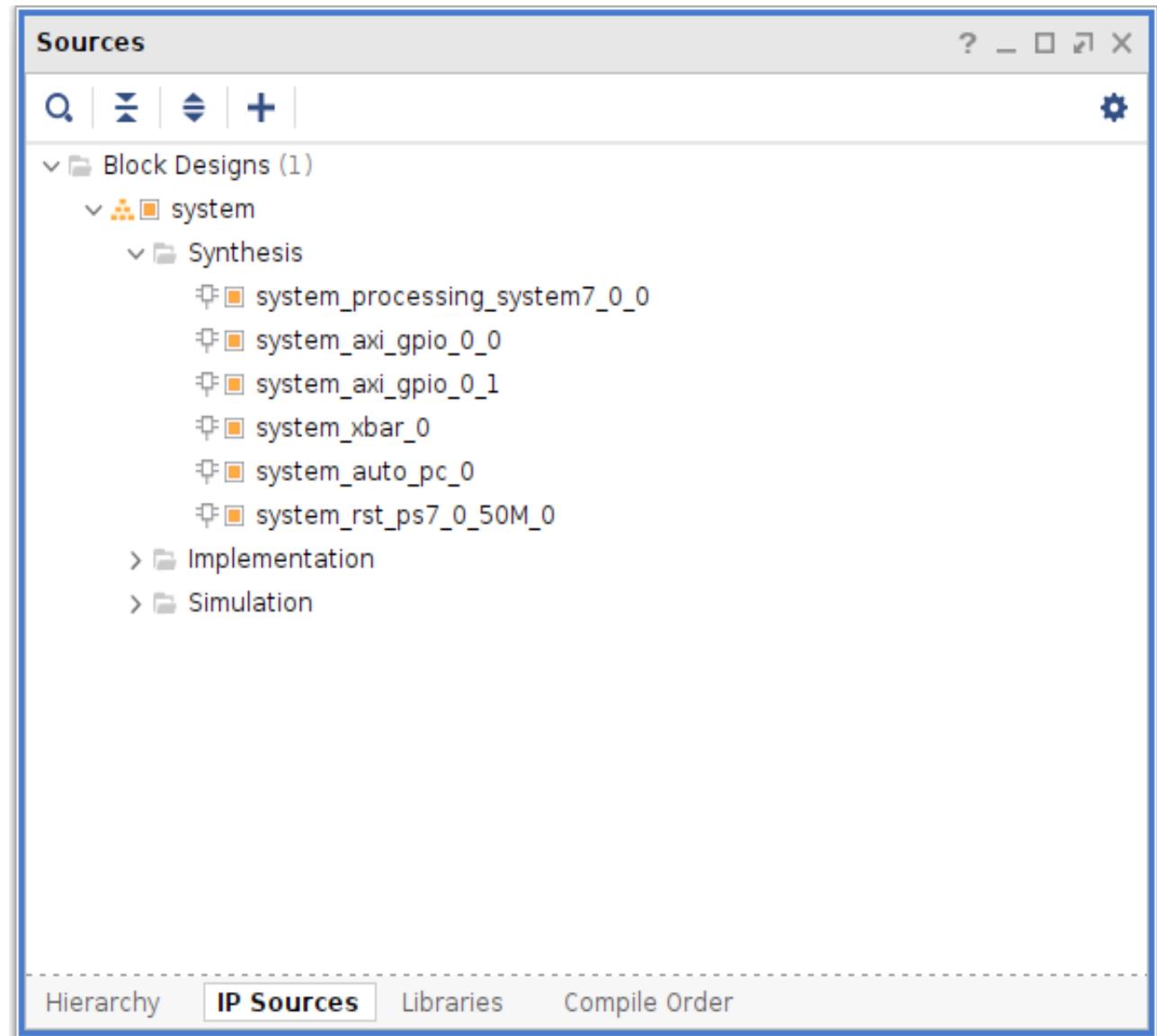
# Знакомство с САПР. Создаем свой проект



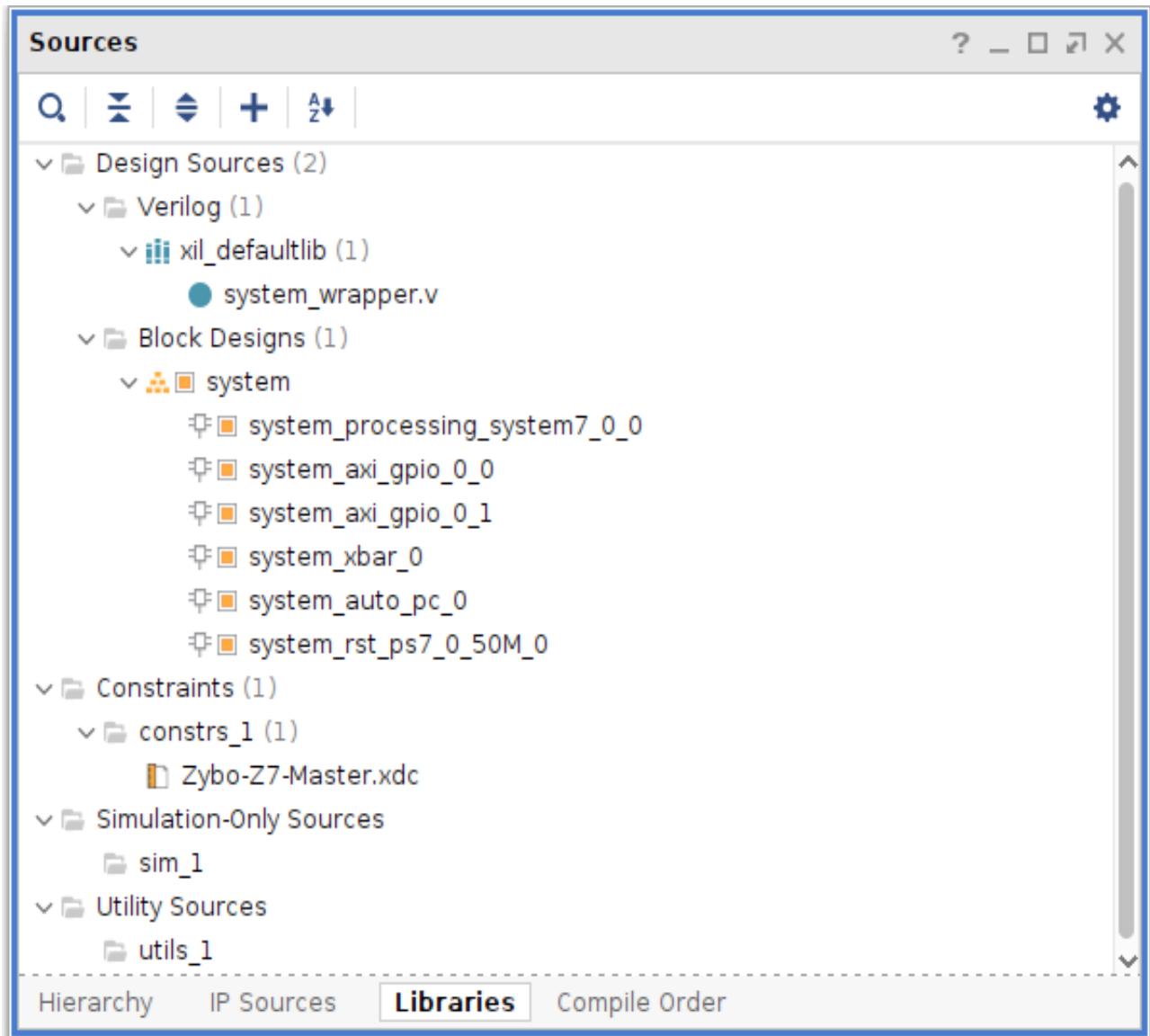
Итого имеем:

1. Block Design (\*.bd) вместе с оберткой (\*.sv, \*.v или \*.vhdl).
2. Файл ограничений (\*.xdc).
3. Не хватает тестов (Testbench может иметь расширение: \*.sv, \*.v или \*.vhdl).

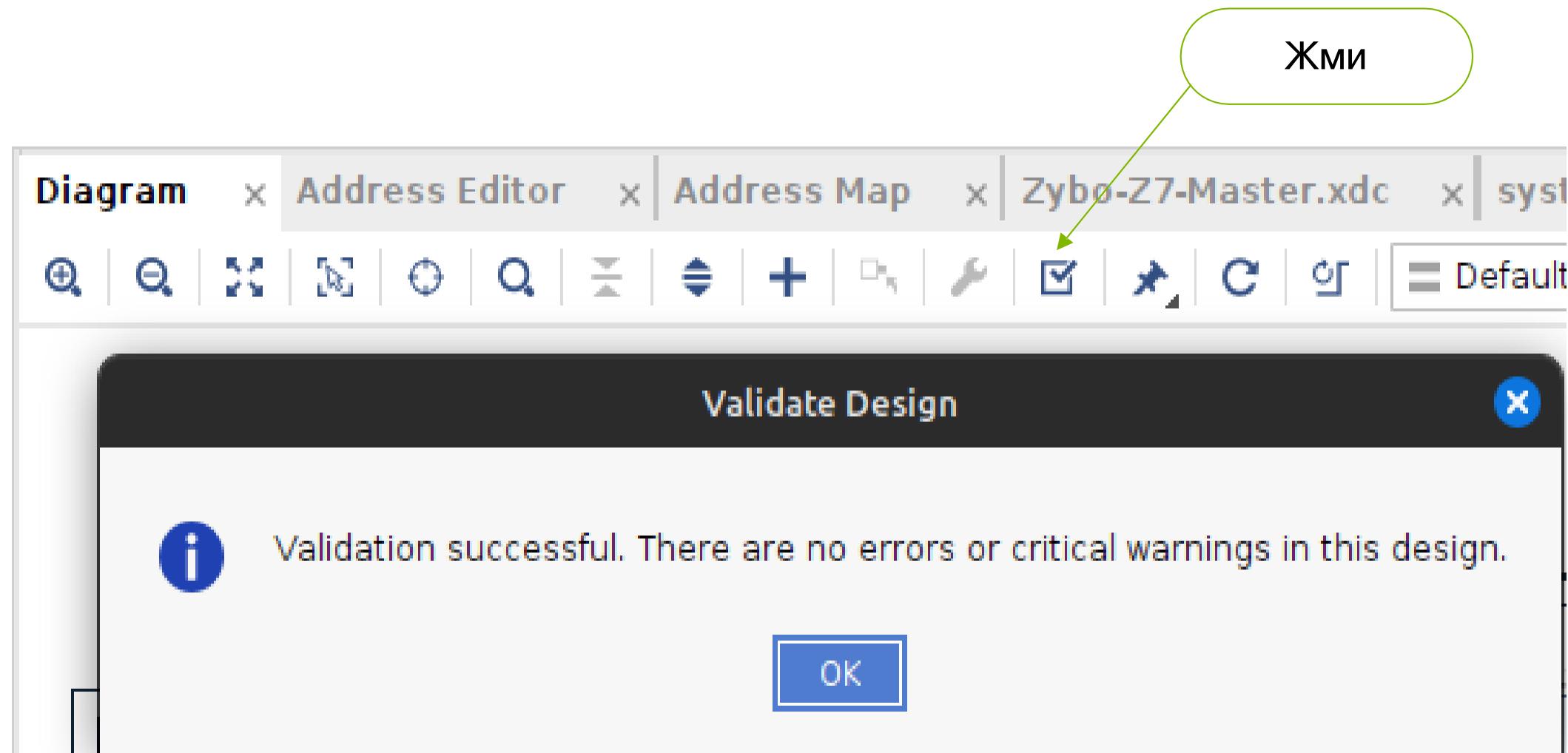
# Знакомство с САПР. Создаем свой проект



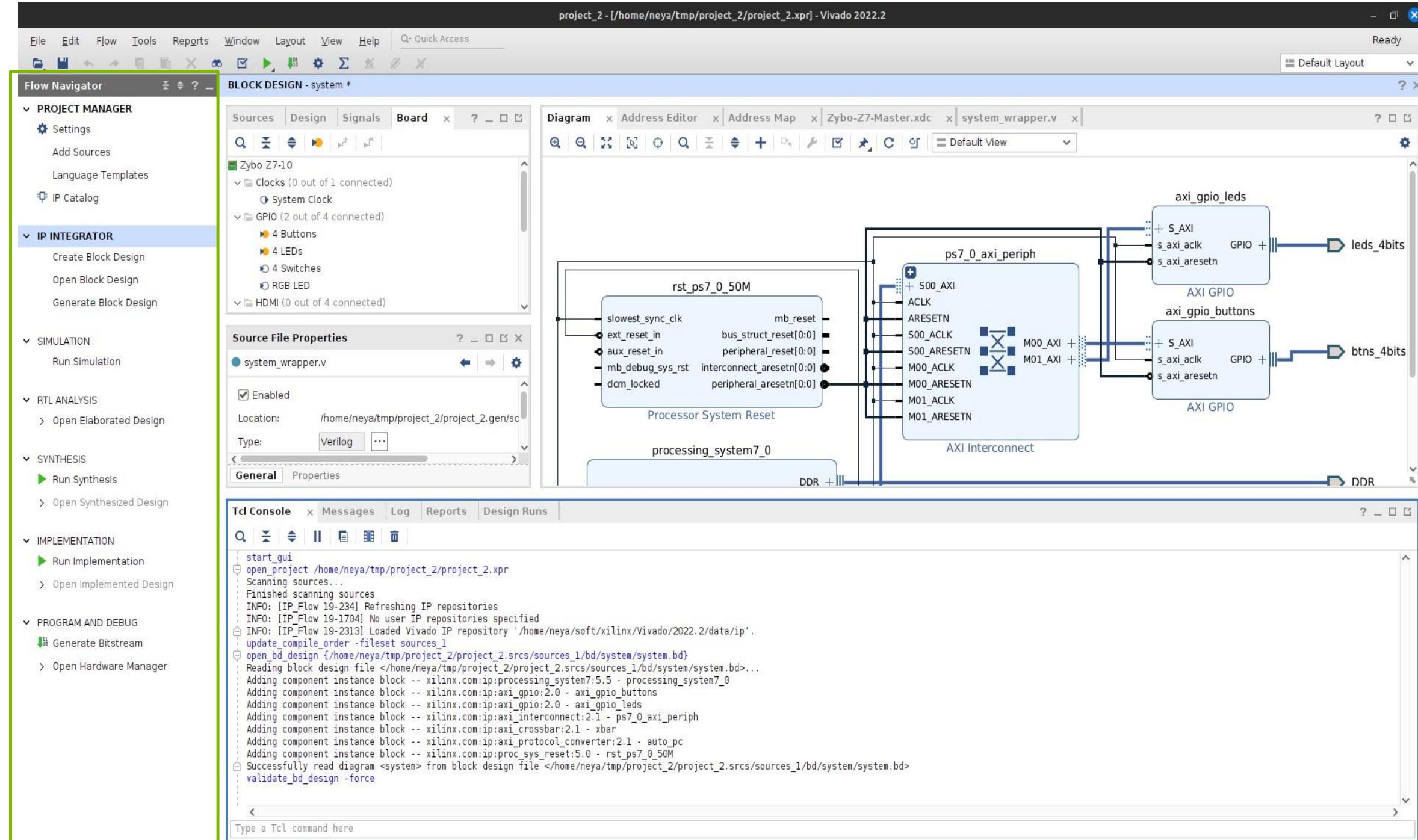
# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект

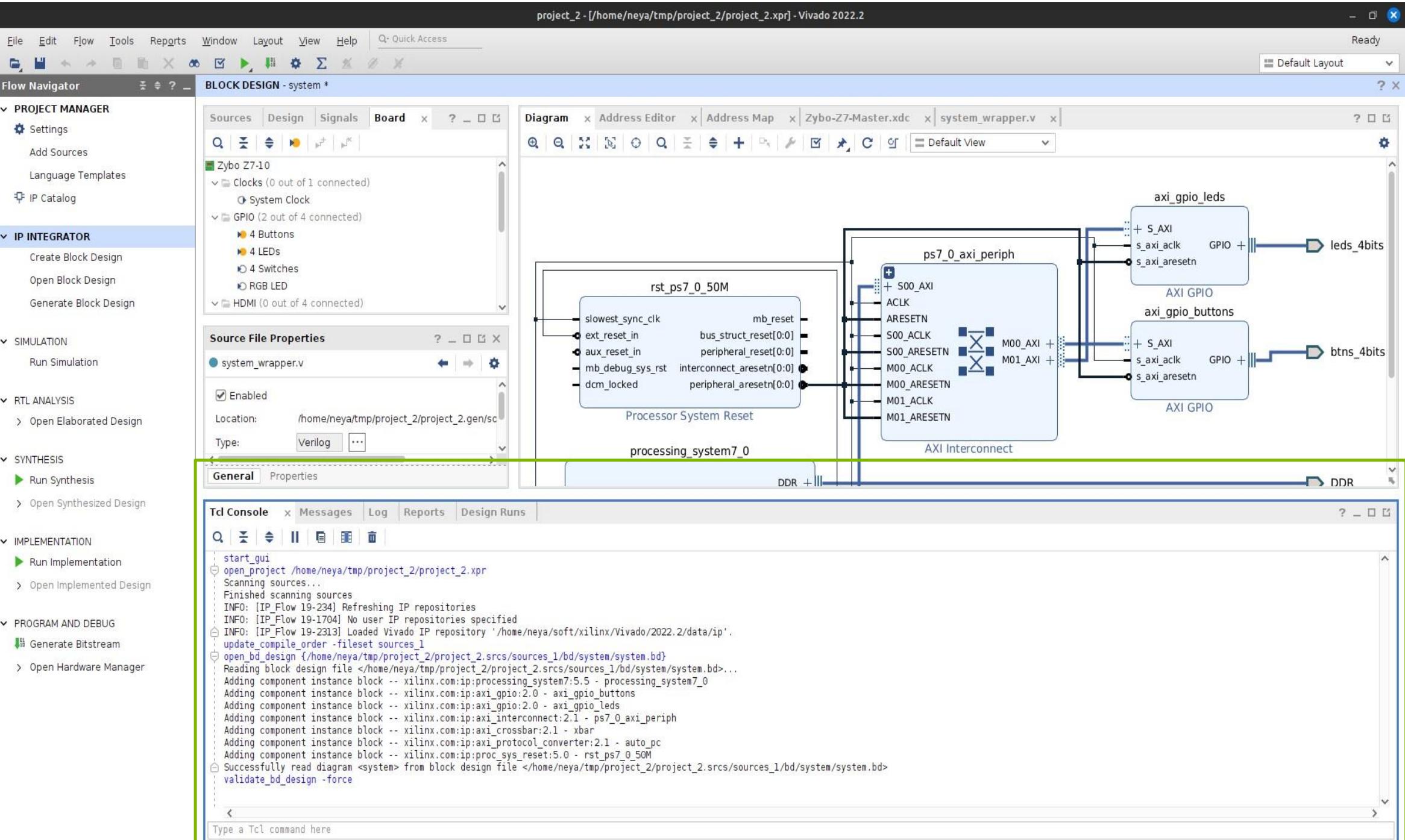


# Знакомство с САПР. Создаем свой проект



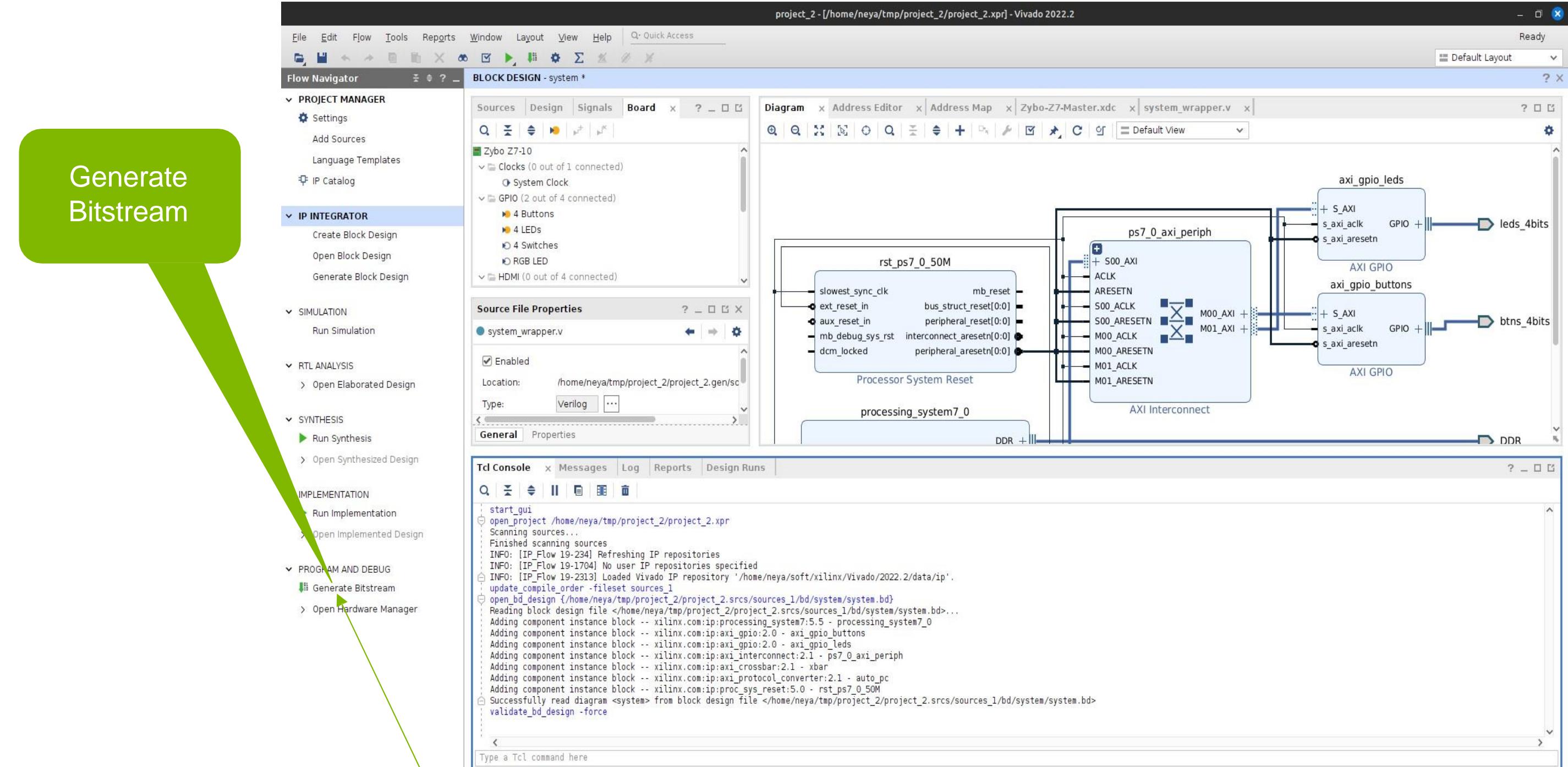
Порядок работы сверху вниз

# Знакомство с САПР. Создаем свой проект



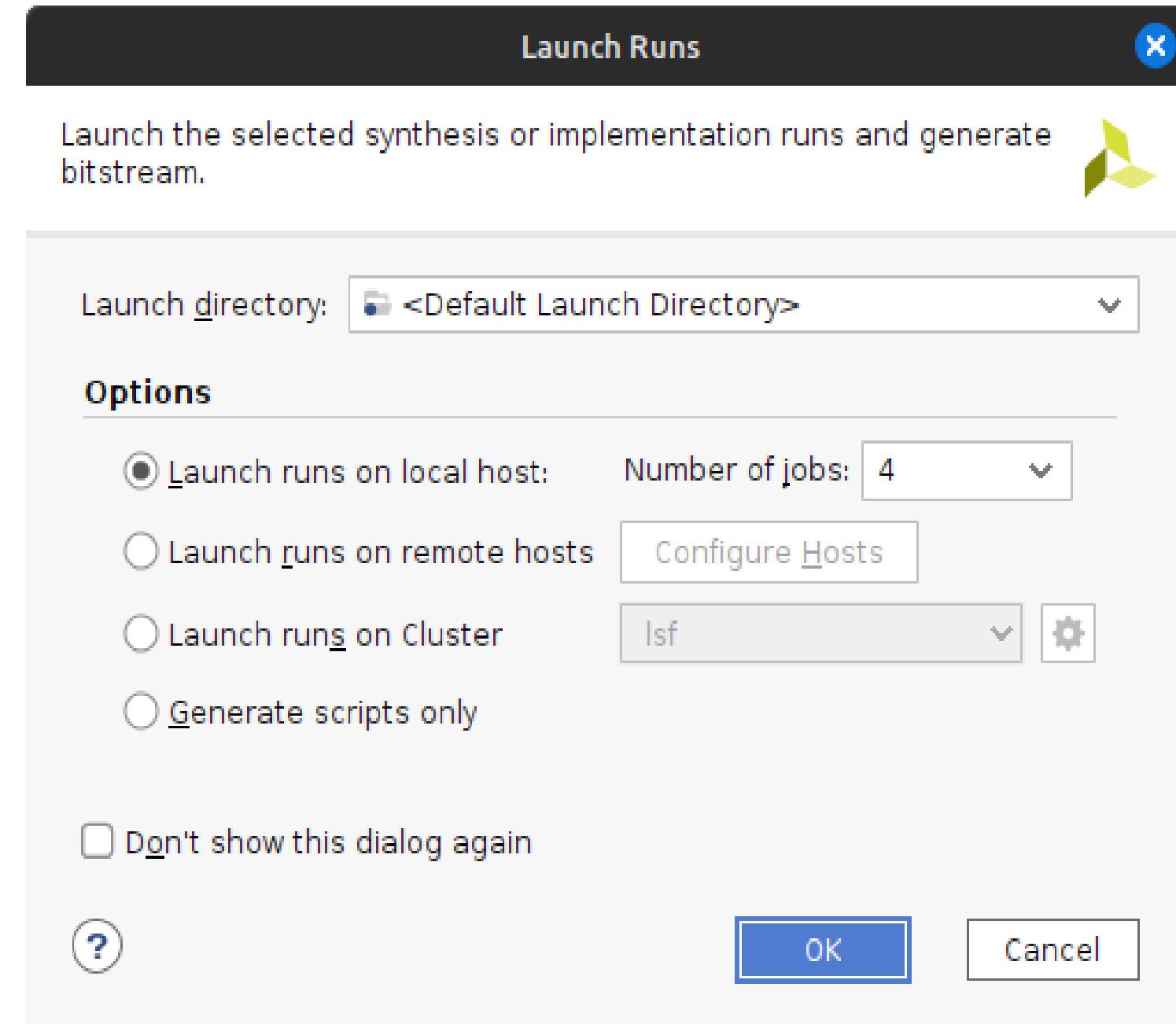
TCL - Мощный инструмент автоматизации

# Знакомство с САПР. Создаем свой проект



Можно сразу попросить прошивку

# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект



The screenshot shows a software interface titled 'Tcl Console'. The window has tabs at the top: 'Tcl Console' (selected), 'Messages', 'Log', 'Reports', and 'Design Runs'. Below the tabs are several icons: a magnifying glass, a double arrow, a triangle, a double vertical bar, a clipboard, a blue square, and a trash can. The main area displays a log of synthesis commands and their outputs:

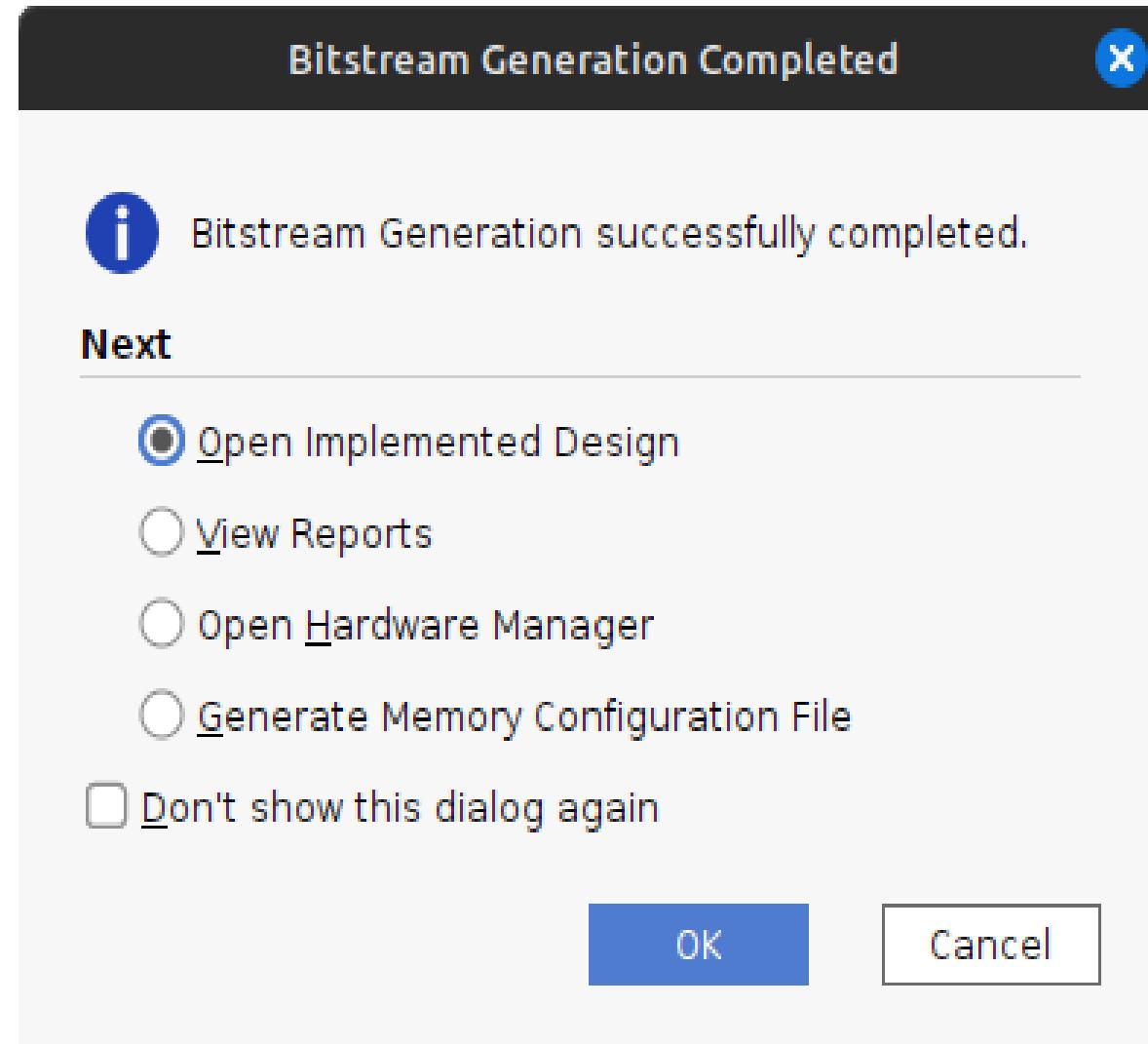
```
INFO: [IP_Flow 19-6924] IPCACHE: Running cache check for IP inst:  
[Mon Sep 16 15:08:24 2024] Launched system_processing_system7_0_0  
Run output will be captured here:  
system_processing_system7_0_0_synth_1: /home/neya/tmp/project_2/p  
system_axi_gpio_0_0_synth_1: /home/neya/tmp/project_2/project_2.r  
system_axi_gpio_0_1_synth_1: /home/neya/tmp/project_2/project_2.r  
system_xbar_0_synth_1: /home/neya/tmp/project_2/project_2.runs/sy  
system_rst_ps7_0_50M_0_synth_1: /home/neya/tmp/project_2/project_2.r  
system_auto_pc_0_synth_1: /home/neya/tmp/project_2/project_2.runs/sy  
nth_1: /home/neya/tmp/project_2/project_2.runs/synth_1/runme.l  
[Mon Sep 16 15:08:24 2024] Launched impl_1...  
Run output will be captured here: /home/neya/tmp/project_2/project_2.r  
launch_runs: Time (s): cpu = 00:00:30 ; elapsed = 00:00:25 . Memo
```

At the bottom of the console window, there is a command prompt:

```
launch_runs impl_1 -to_step write_bitstream -jobs 4
```

Или вот так,  
по-хакерски

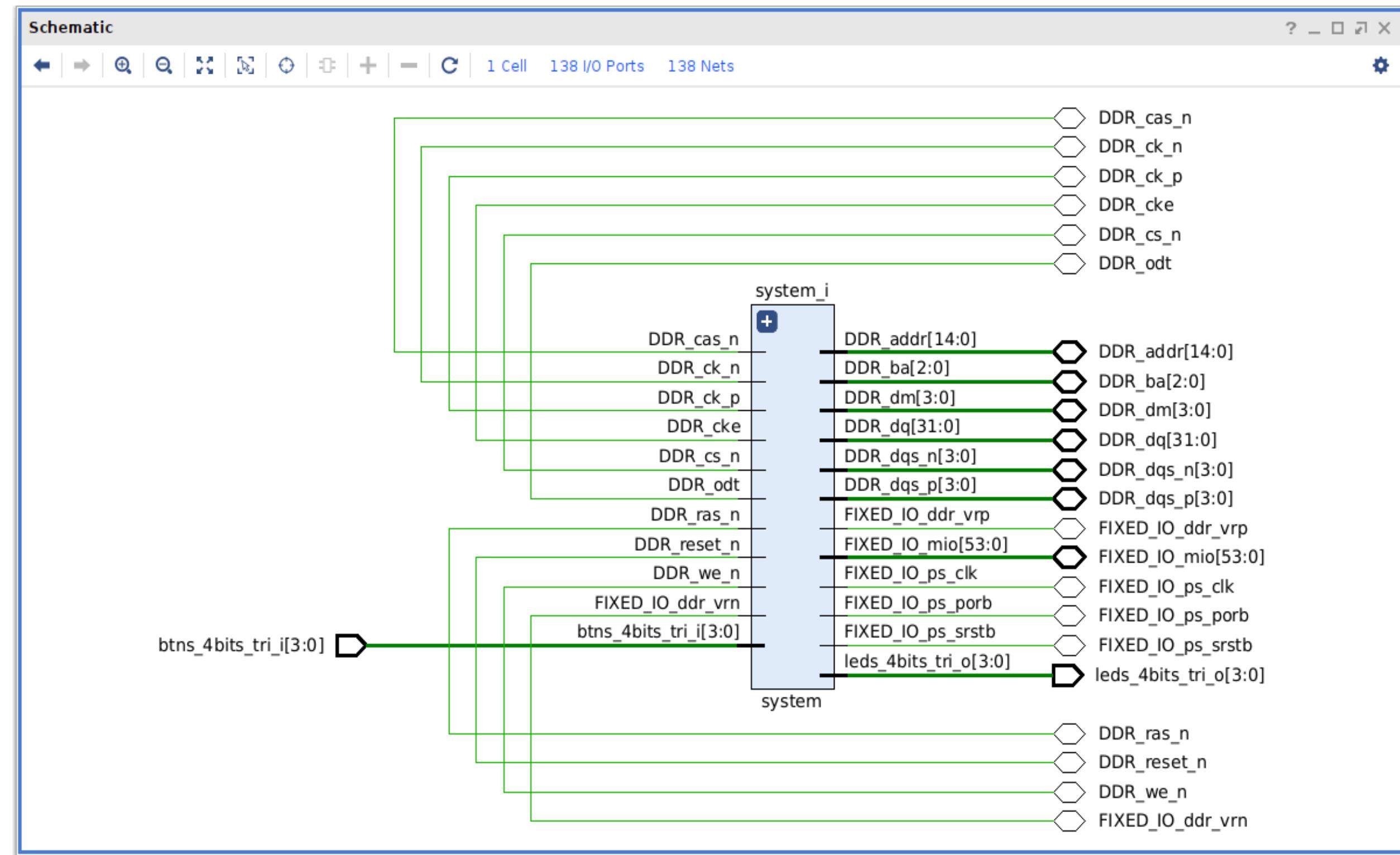
# Знакомство с САПР. Создаем свой проект



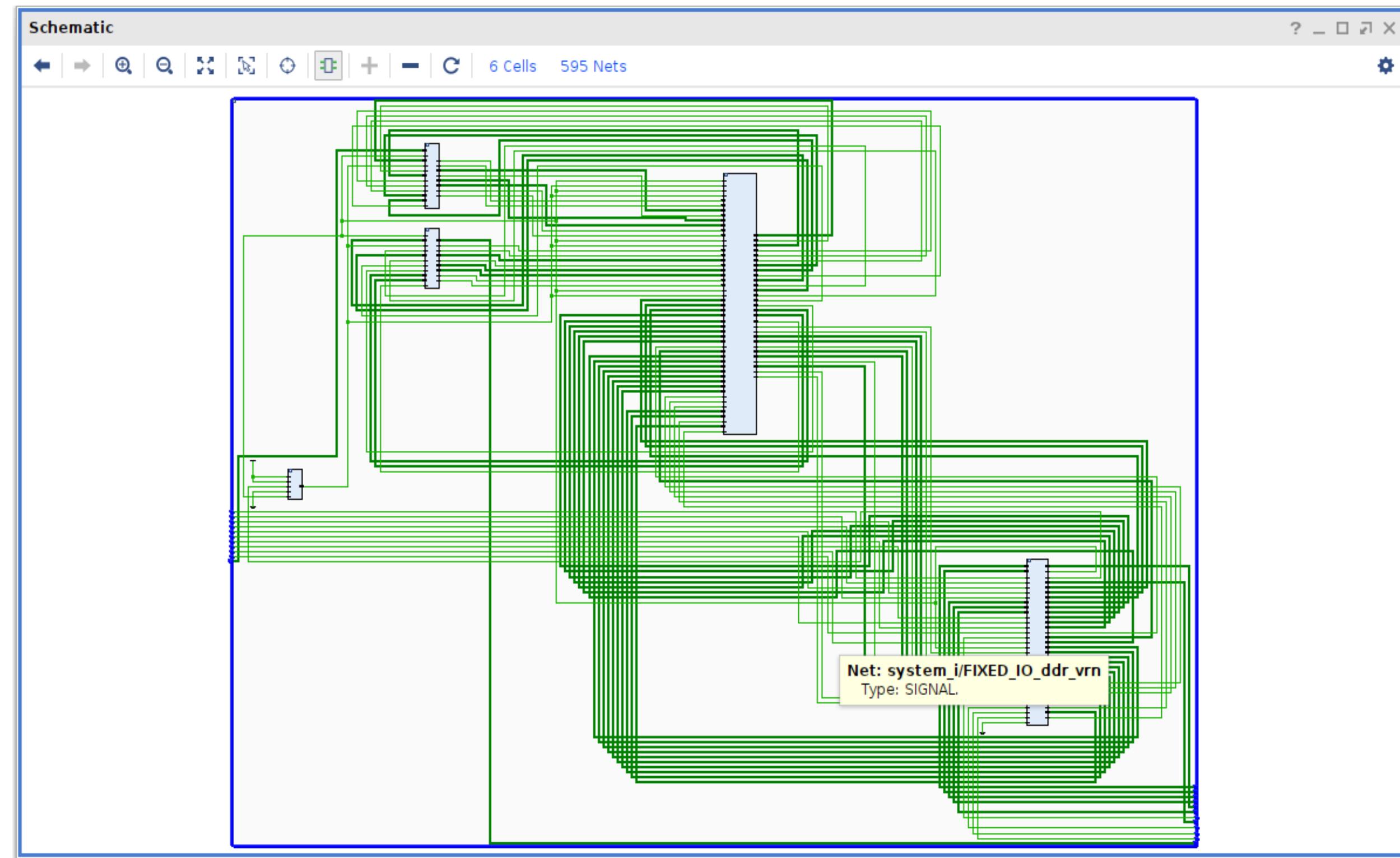
Результат один. Теперь можно посмотреть все промежуточные этапы:

1. Элаборация.
2. Синтез.
3. Имплементация.

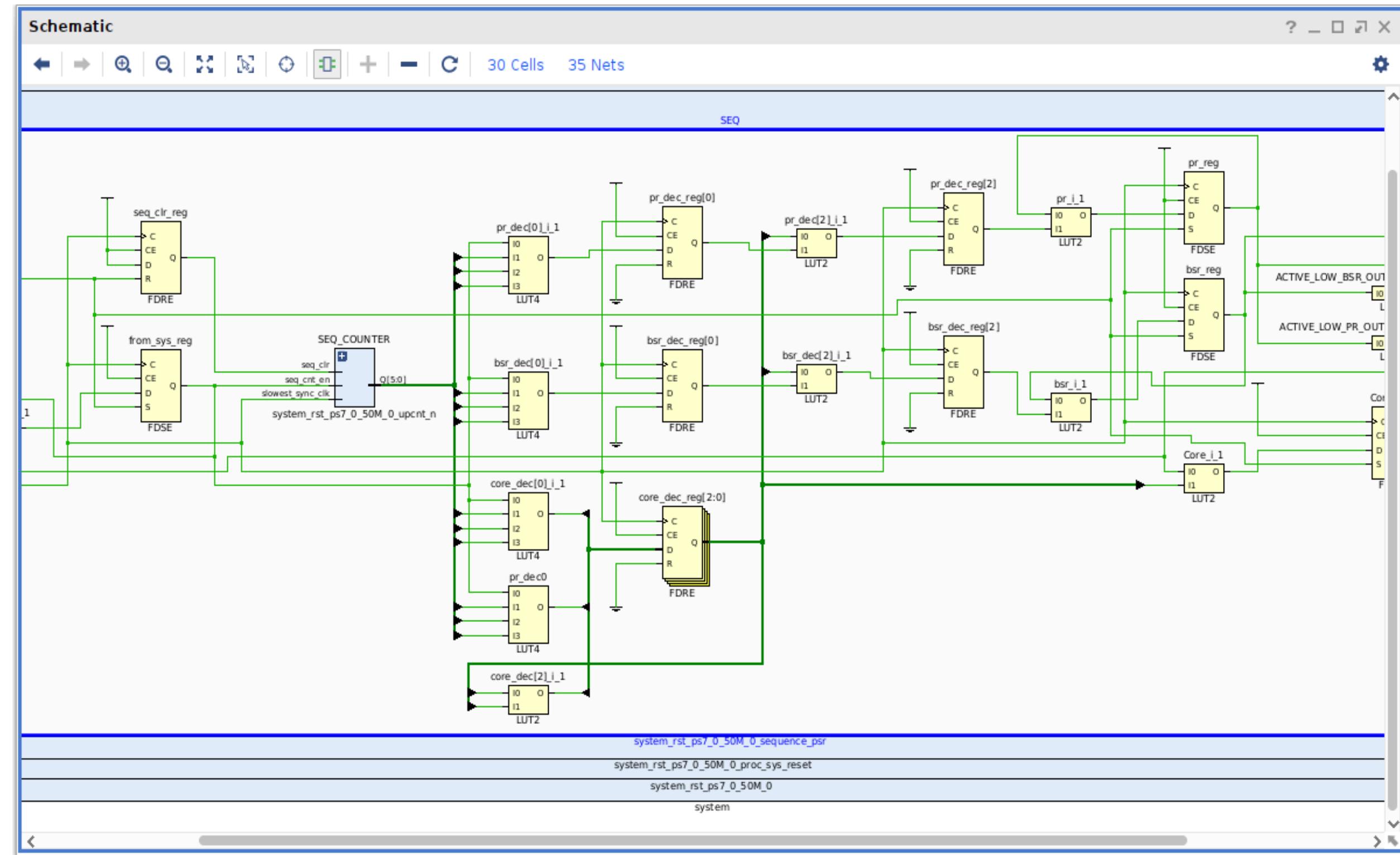
# Знакомство с САПР. Создаем свой проект



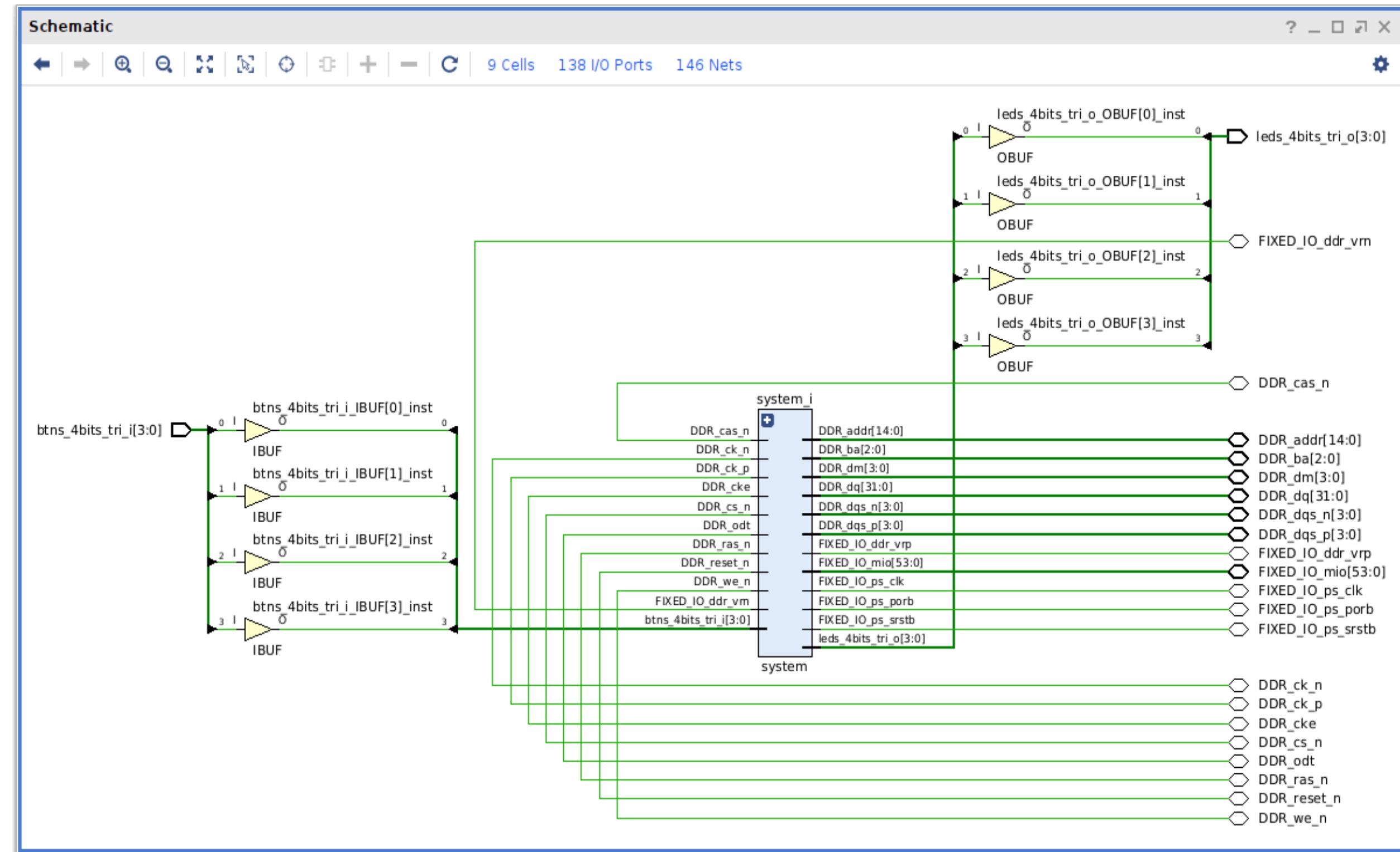
# Знакомство с САПР. Создаем свой проект



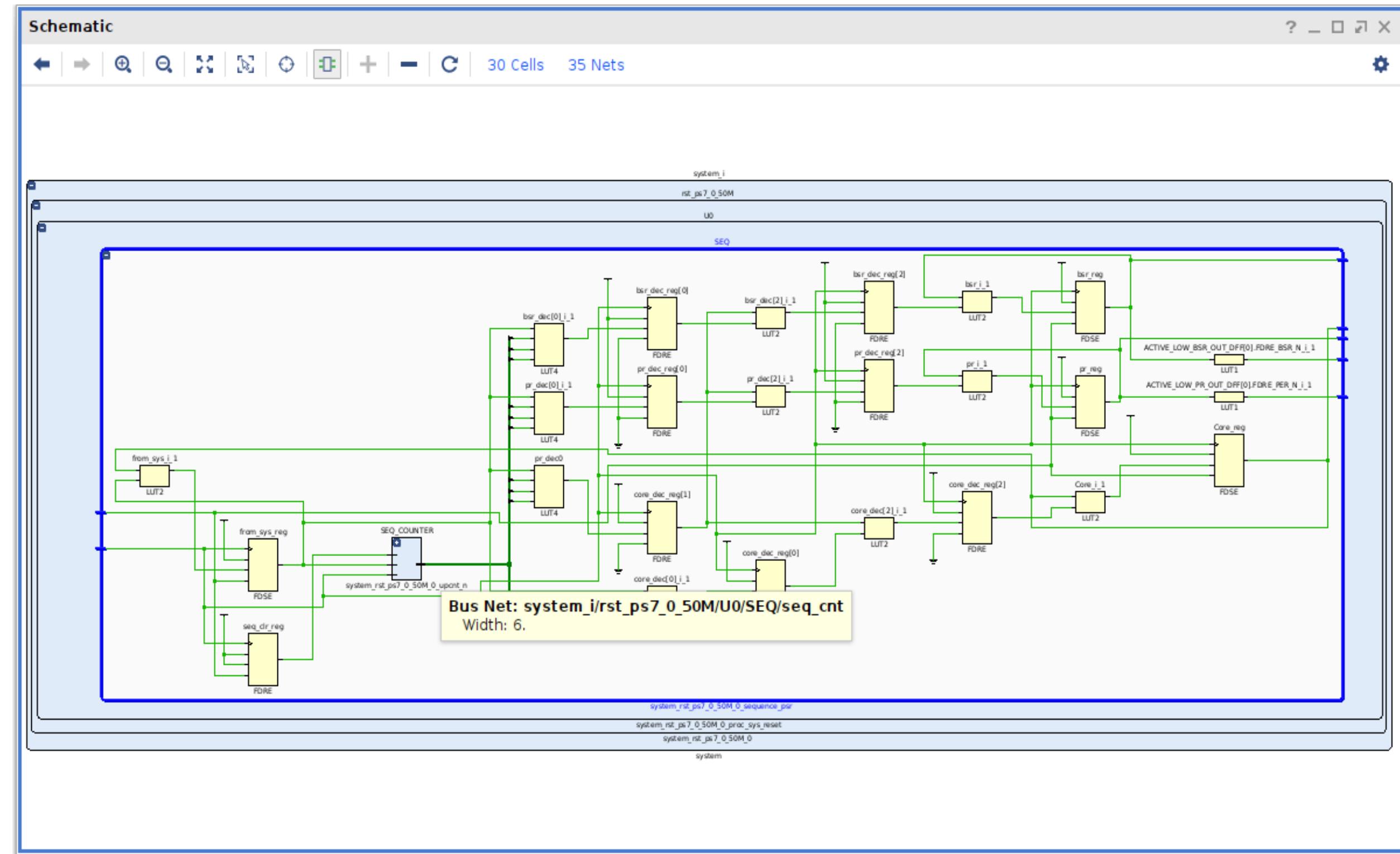
# Знакомство с САПР. Создаем свой проект



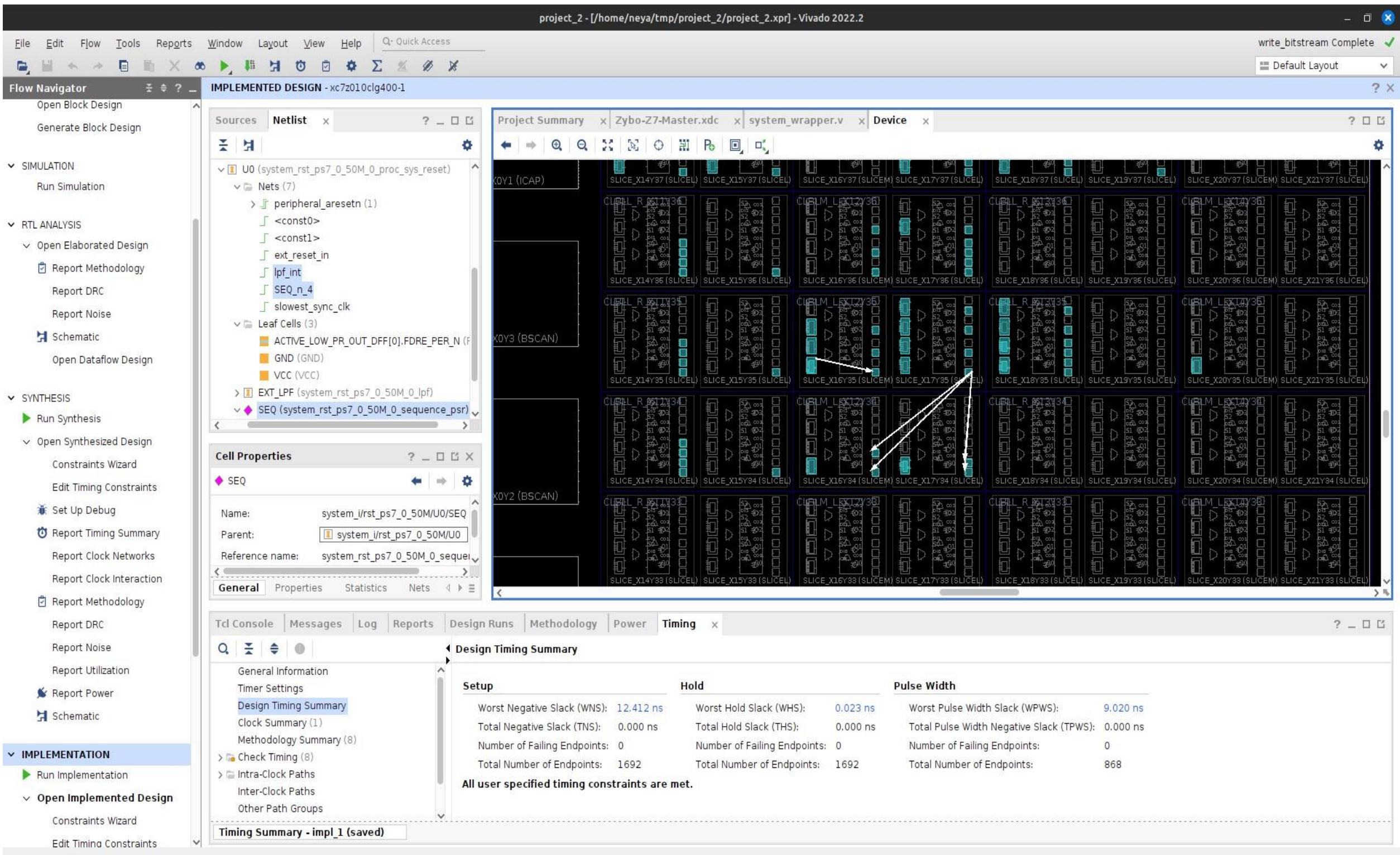
# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект

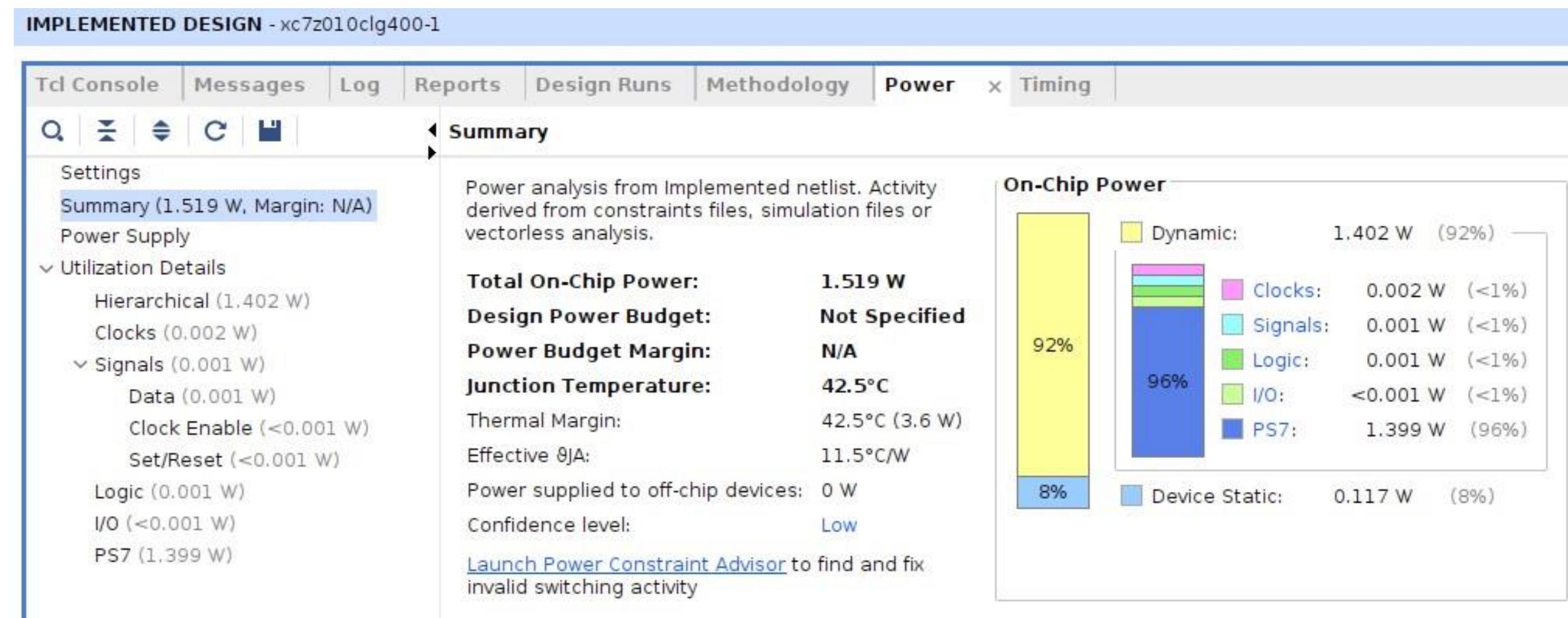


# Знакомство с САПР. Создаем свой проект

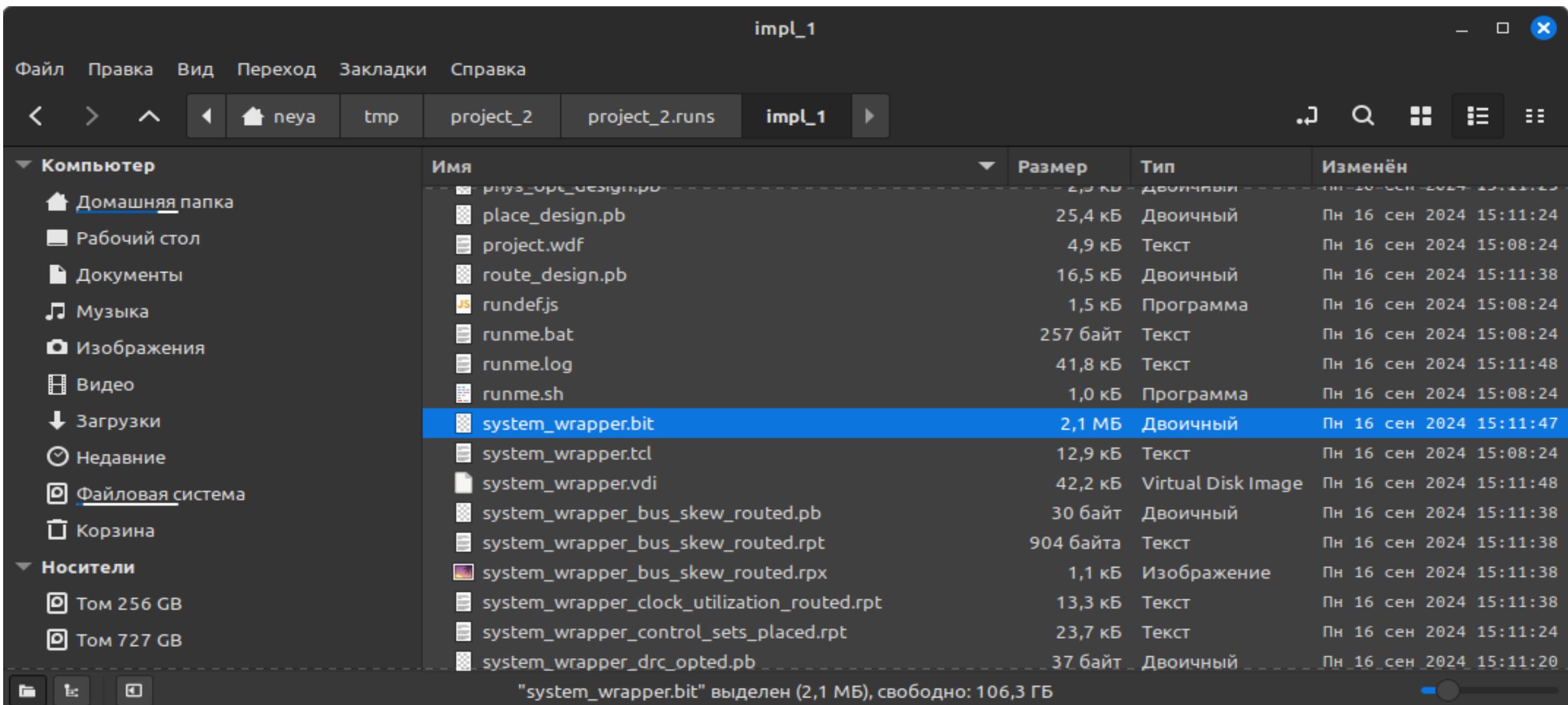


Name	Slice LUTs (17600)	Slice Registers (35200)	Slice (4400)	LUT as Logic (17600)	LUT as Memory (6000)	Bonded IOB (100)	Bonded IOPADs (130)	BUFGCTRL (32)
system_wrapper	585	801	266	523	62	8	130	1
system_i (system)	585	801	266	523	62	0	0	1
axi_gpio_buttons	41	68	22	41	0	0	0	0
U0 (system_ax)	41	68	22	41	0	0	0	0
AXI_LITE_IPIF	30	31	13	30	0	0	0	0
I_SLAVE_A	30	31	13	30	0	0	0	0
> I_DECO	13	6	7	13	0	0	0	0
gpio_core_1	11	30	13	11	0	0	0	0
Not_Dual,	0	16	4	0	0	0	0	0
U0 (system_ax)	37	52	18	37	0	0	0	0
AXI_LITE_IPIF	29	31	13	29	0	0	0	0
I_SLAVE_A	29	31	13	29	0	0	0	0
> I_DECO	12	6	7	12	0	0	0	0
MEM	1	0	1	1	0	0	0	0
MEM	1	0	1	1	0	0	0	0
gpio_core_1	8	14	5	8	0	0	0	0
processing_system7_0_axi_periph	491	648	221	430	61	0	0	0
s00_couplers (	390	534	181	329	61	0	0	0
auto_pc (sys	390	534	181	329	61	0	0	0
> inst (syst	390	534	181	329	61	0	0	0
> gen_ax	390	534	181	329	61	0	0	0
xbar (system_x	101	114	48	101	0	0	0	0
> inst (system	101	114	48	101	0	0	0	0
> gen_sasd	101	114	48	101	0	0	0	0
rst_ps7_0_50M (s	17	33	11	16	1	0	0	0
U0 (system_rst	17	33	11	16	1	0	0	0
EXT_LPF (sys	5	17	7	4	1	0	0	0
ACTIVE_LC	1	4	1	1	0	0	0	0
ACTIVE_LC	2	4	2	2	0	0	0	0
SEQ (system	12	15	5	12	0	0	0	0
SEQ_COUI	5	6	2	5	0	0	0	0

# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект



# Знакомство с САПР. Создаем свой проект



Пример создания проекта Vivado:

[https://digilent.com/reference/vivado/getting\\_started/2018.2](https://digilent.com/reference/vivado/getting_started/2018.2).



# Спасибо за внимание!