

Discover NX Design Suite 23.5

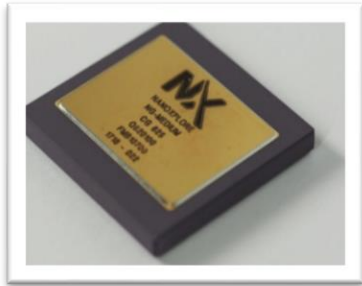


Agenda

- Introduction
- Graphical User Interface
- Design Flow
- IP & Debug Tools
- Documentation
- Conclusion

Introduction





NG medium

Low-End FPGA

- 35kLUTs/32kDFFs
- 3Mb RAM
- 112 DSP
- No HSSL
- No Hard IP Processor

- Companion chip

ESCC9000 qualified



ultra300

Mid-End FPGA

- 290kLUTs/273kDFFs
- 21Mb RAM
- 896 DSP
- 16x HSSL 12G
- ADC/DAC

- Payload
- Platform
- Sensor control
- Power control loop



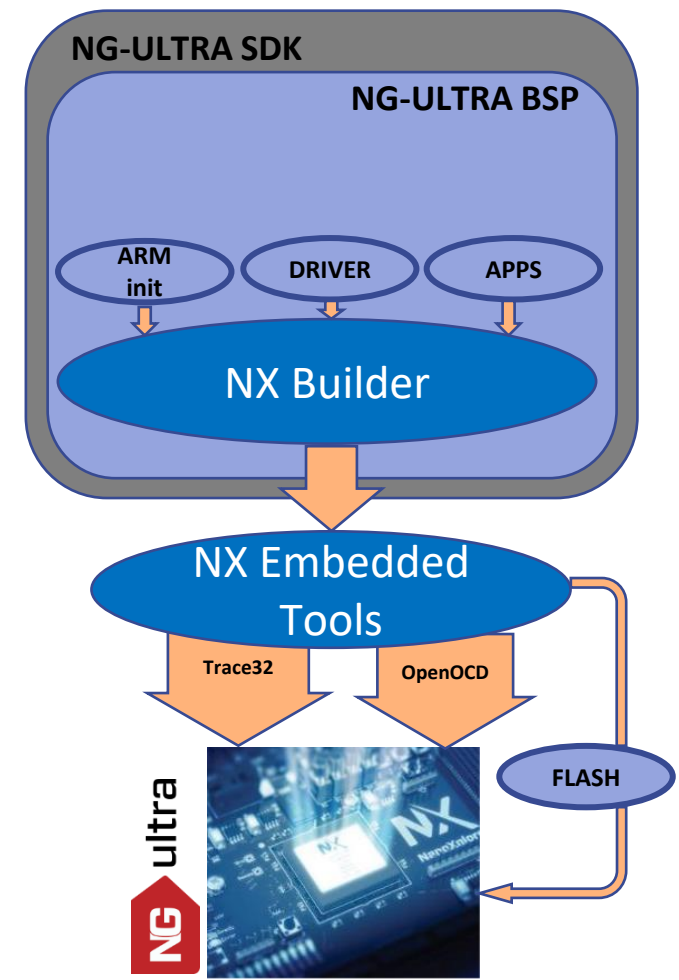
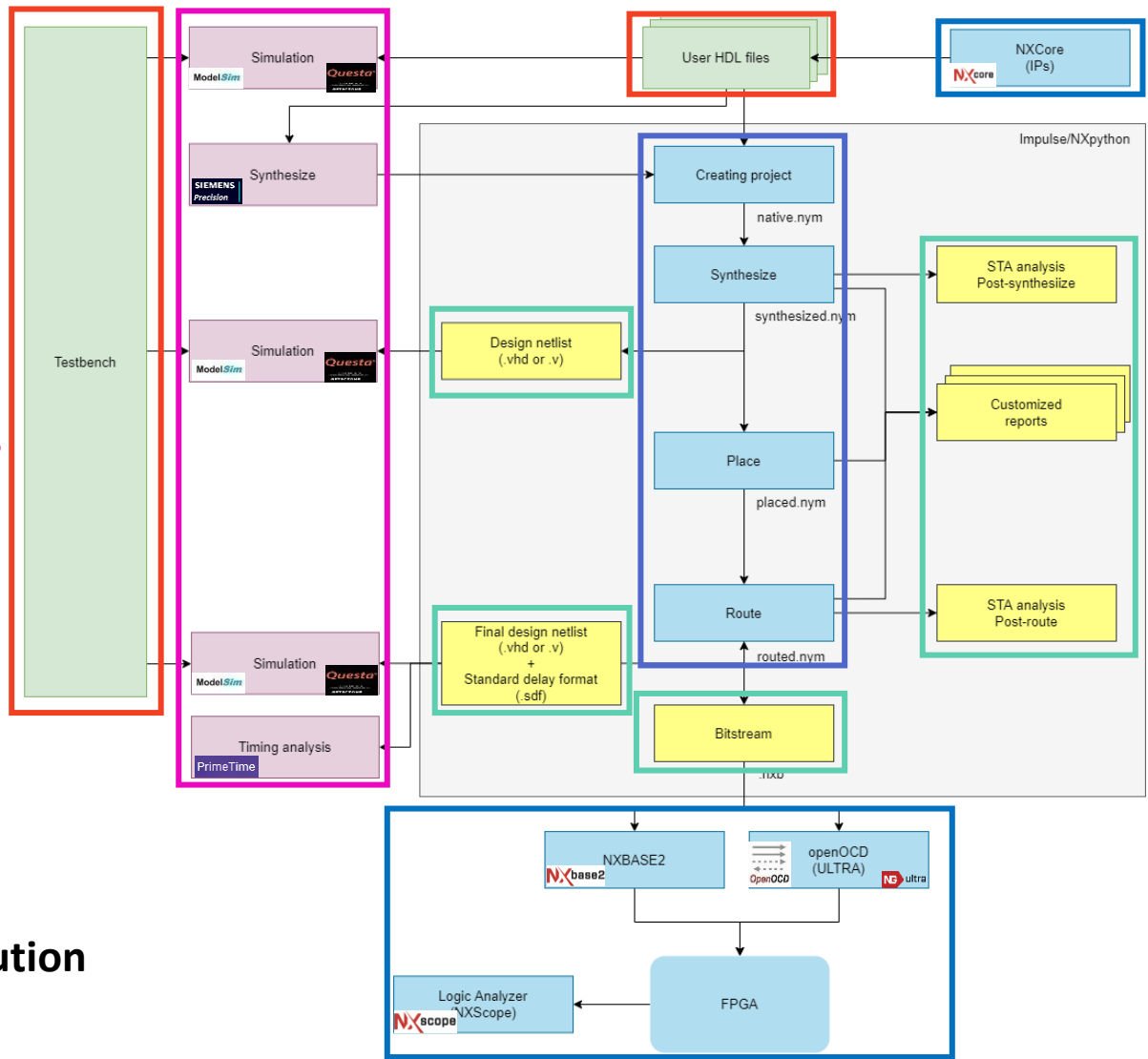
NG ultra

High-End FPGA

- 537kLUTs/505kDFFs
- 32Mb RAM
- 1344 DSP
- 32x HSSL 12G
- Quad-core ARM-R52 (SoC)

- Payload
- Platform

User Resources
 NX Tools
 NX Output Files
 Third-Party Tools



Complete solution

Graphical User Interface

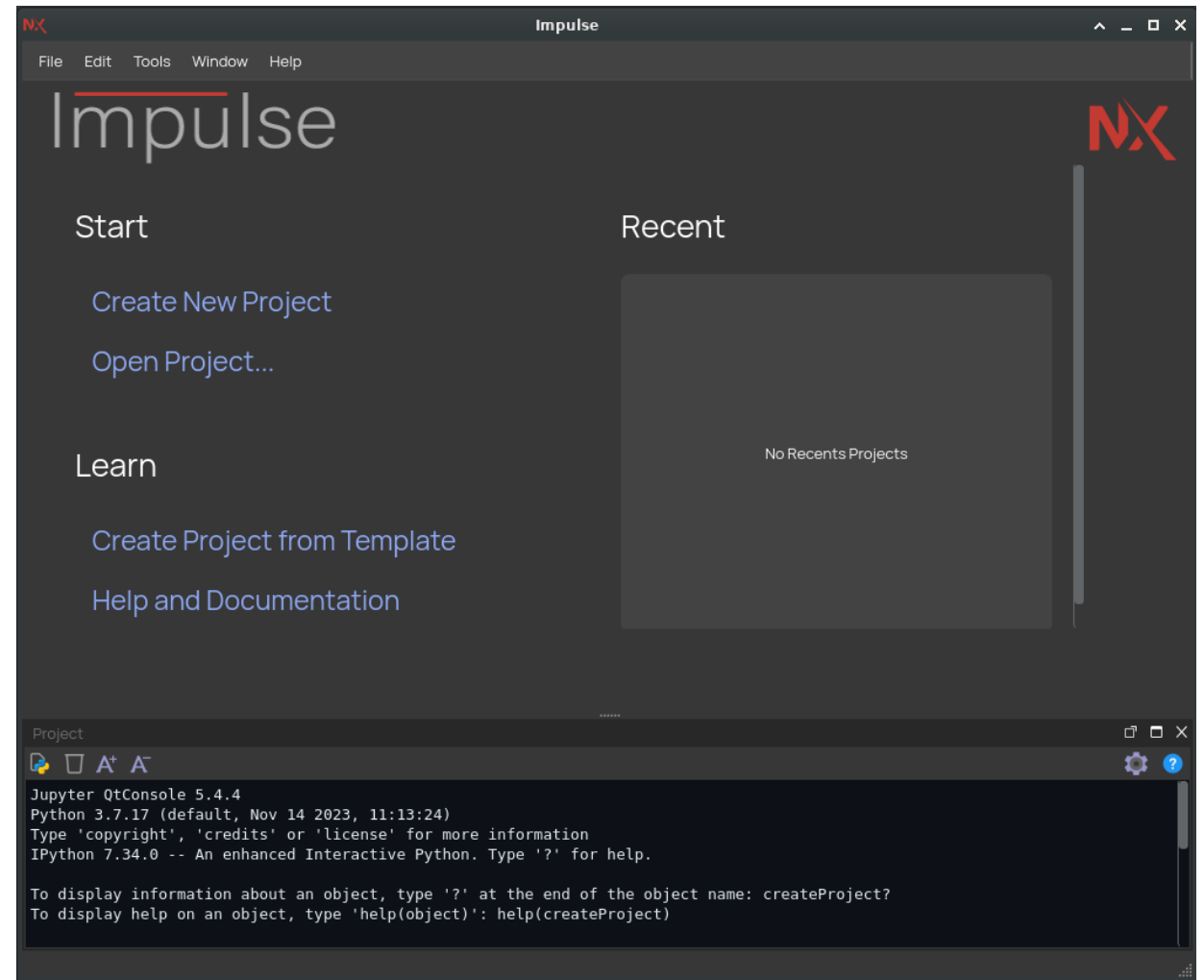


To launch the design flow, you can:

- Use GUI
- Use NXpython (Python based tool)

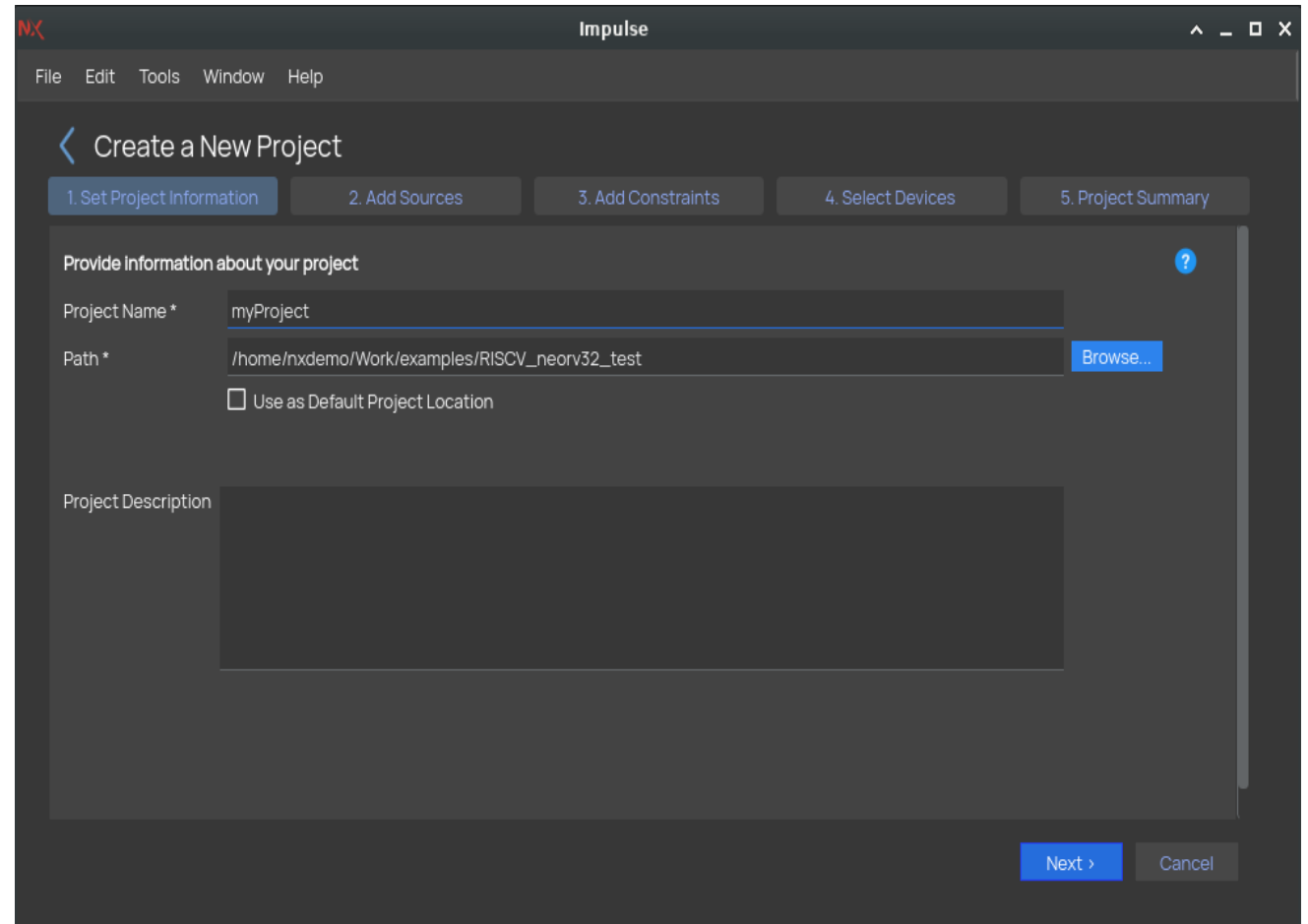
The steps before launching the design flow:

- Create and configure a new project
- Add source files
- Set tool options (Synthesis, Place, Route & Bitstream Generation)
- Add constraints (IOs, timings, placement)

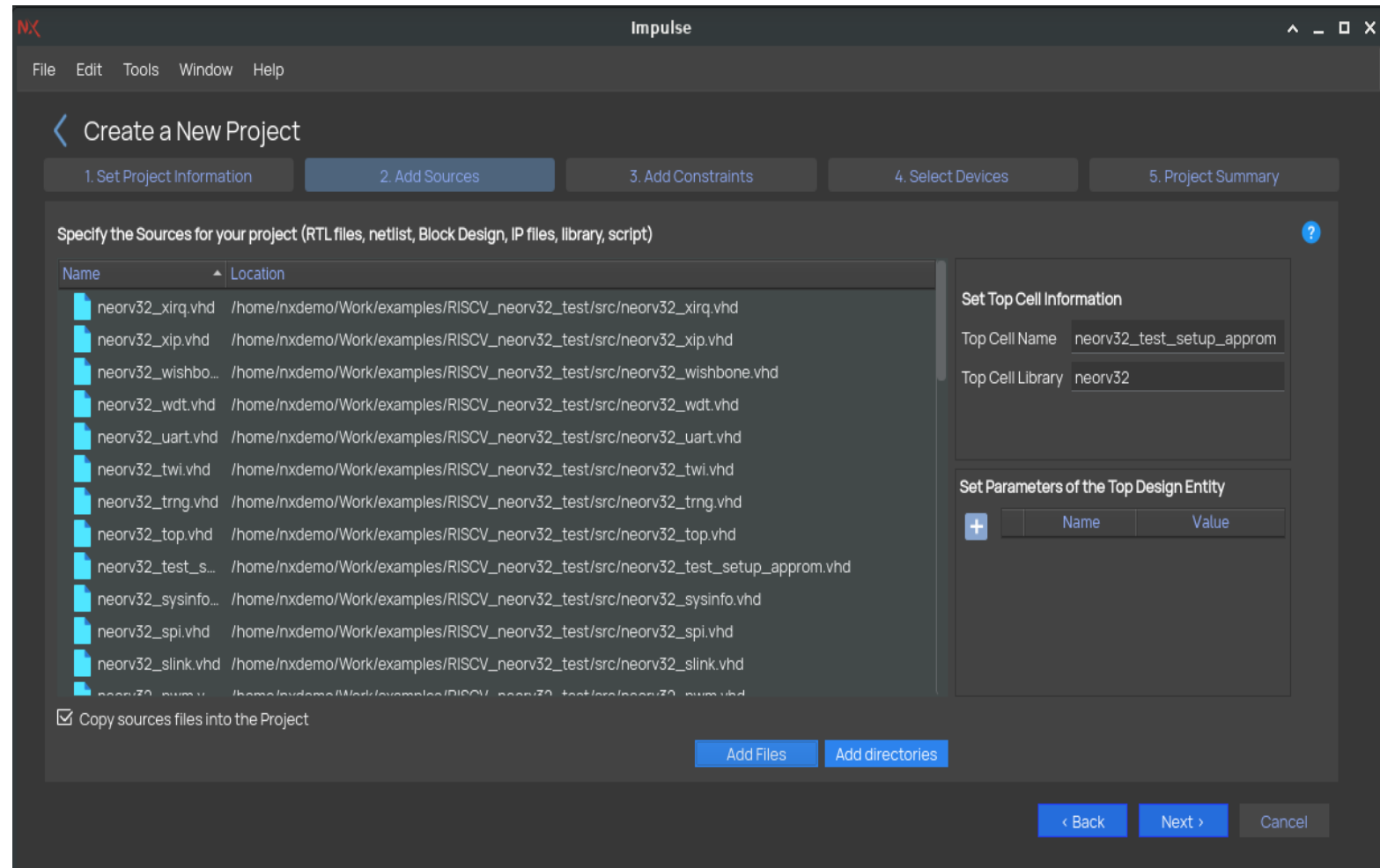


To create a project from scratch:

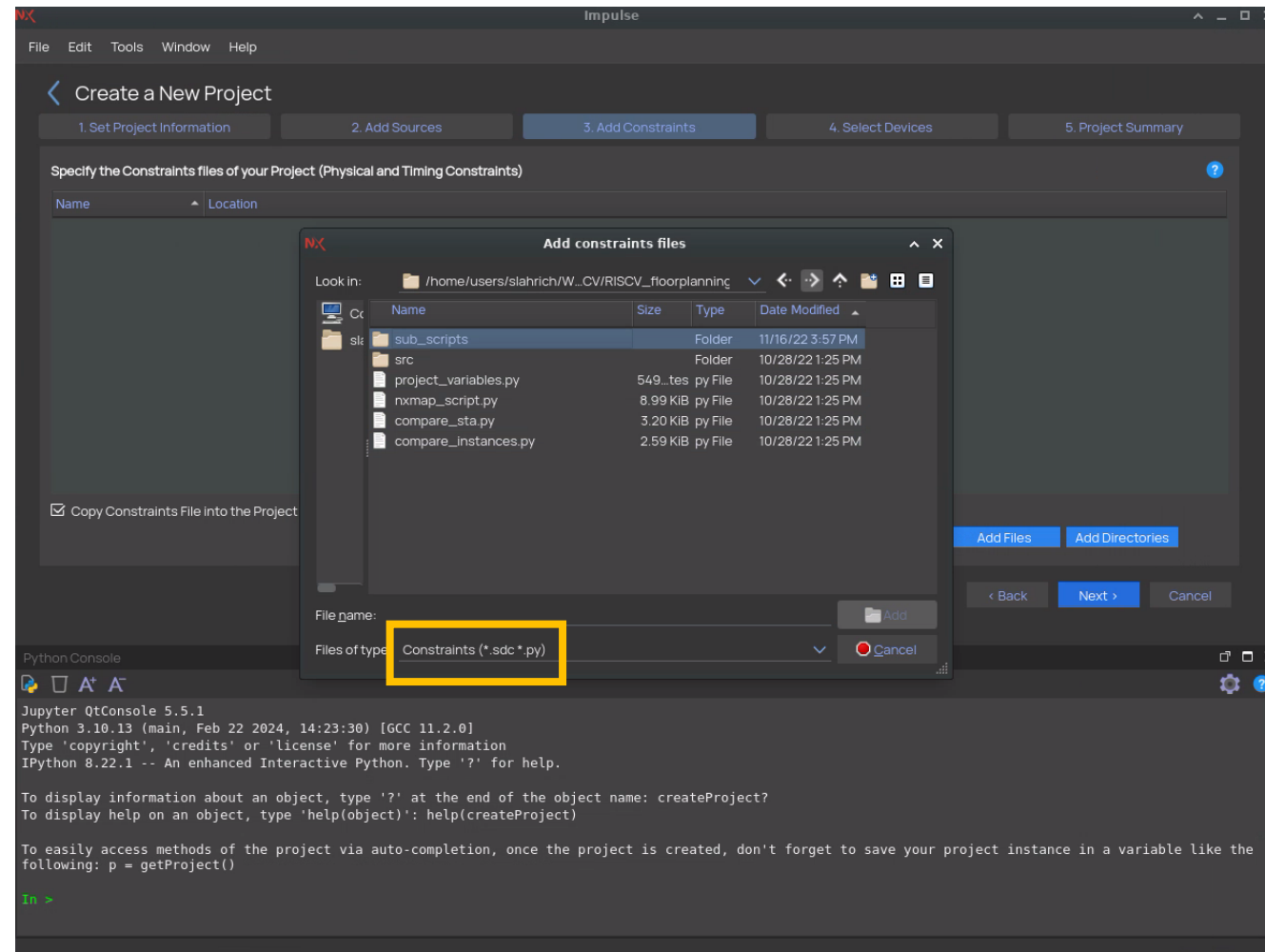
- Create new project
- Provide a project name
- Specify project path



- Add source files
- Specify top cell name
- Specify top cell library

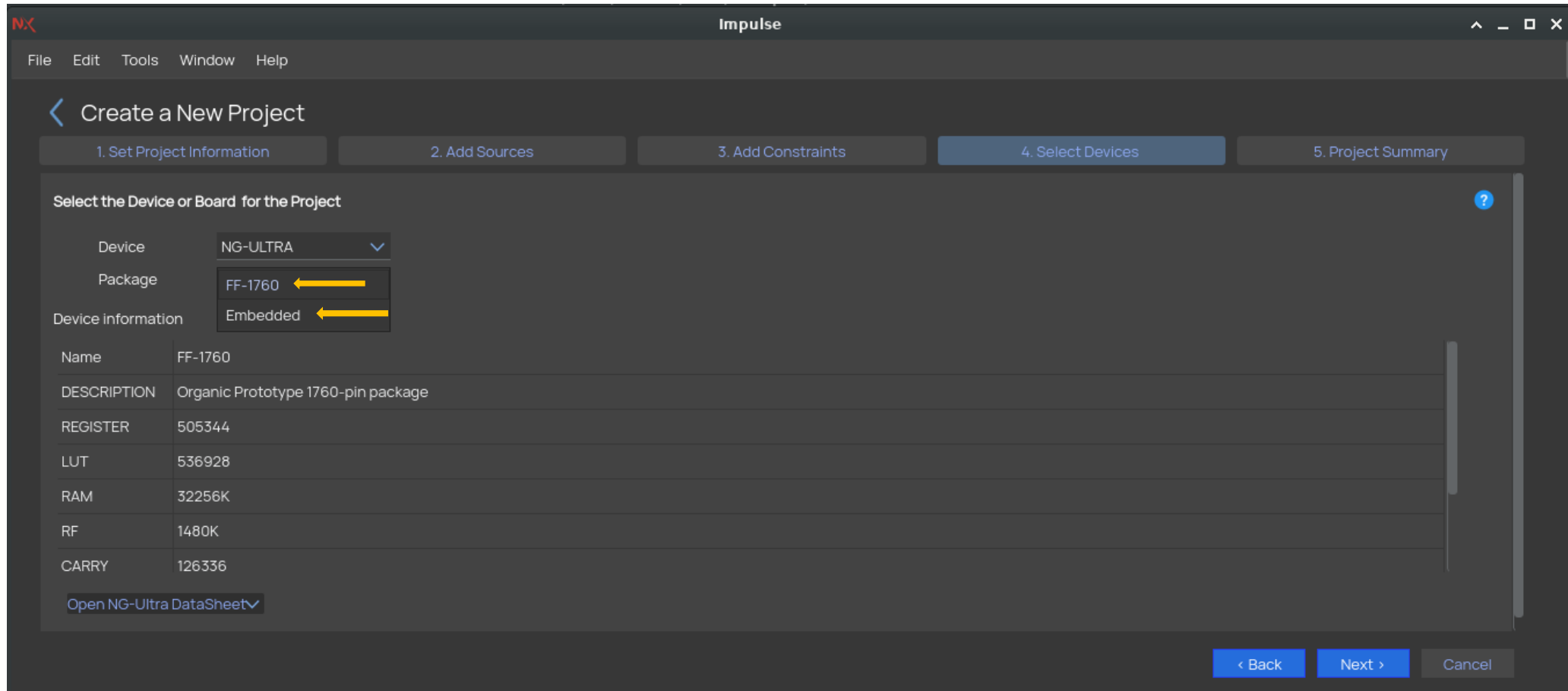


For timing constraints, SDC or Python file (NXDC) is used.

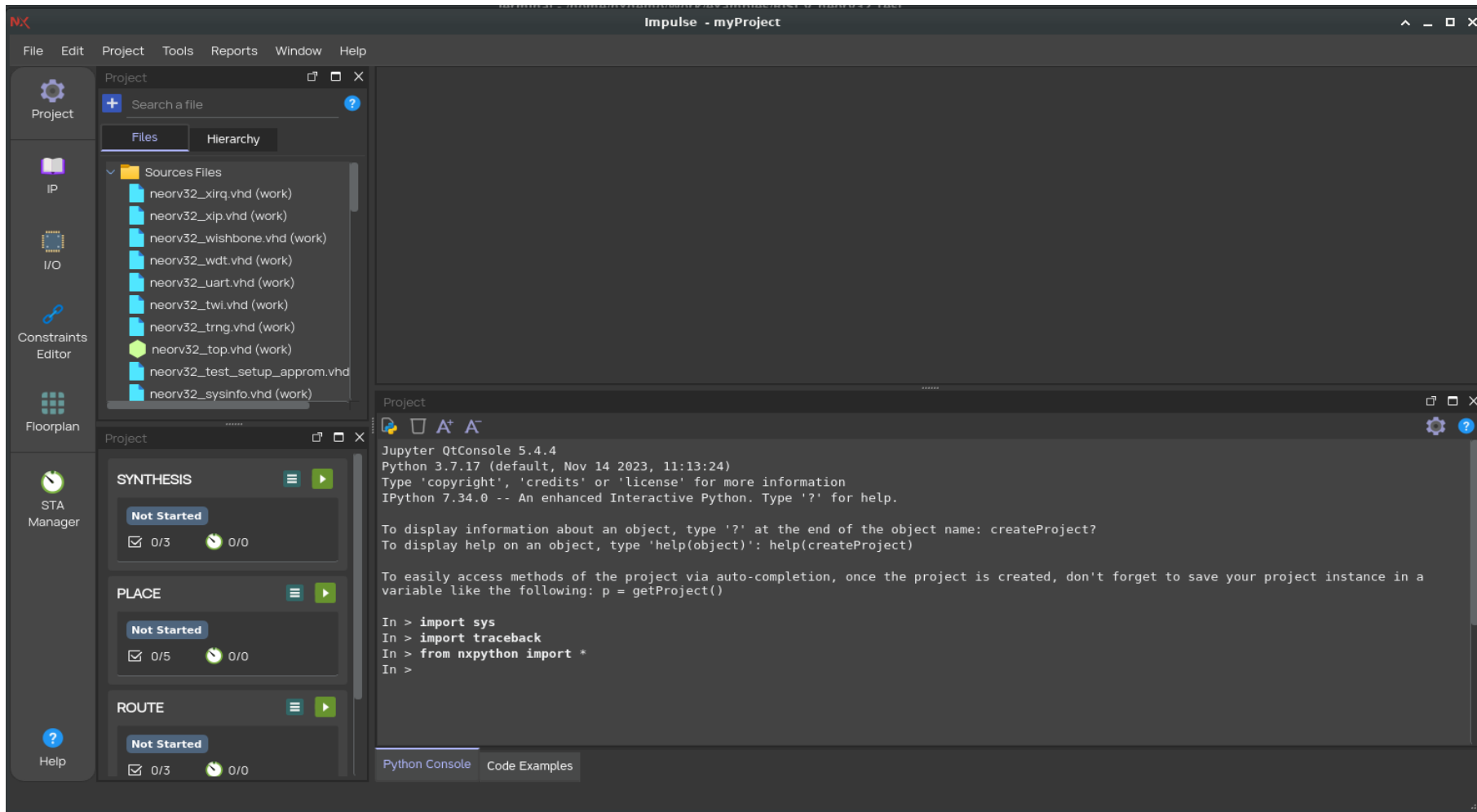


For more details => <https://nanoxplore-wiki.atlassian.net/wiki/spaces/NAN/pages/357467312/NxDesignSuite+23.5+STA>

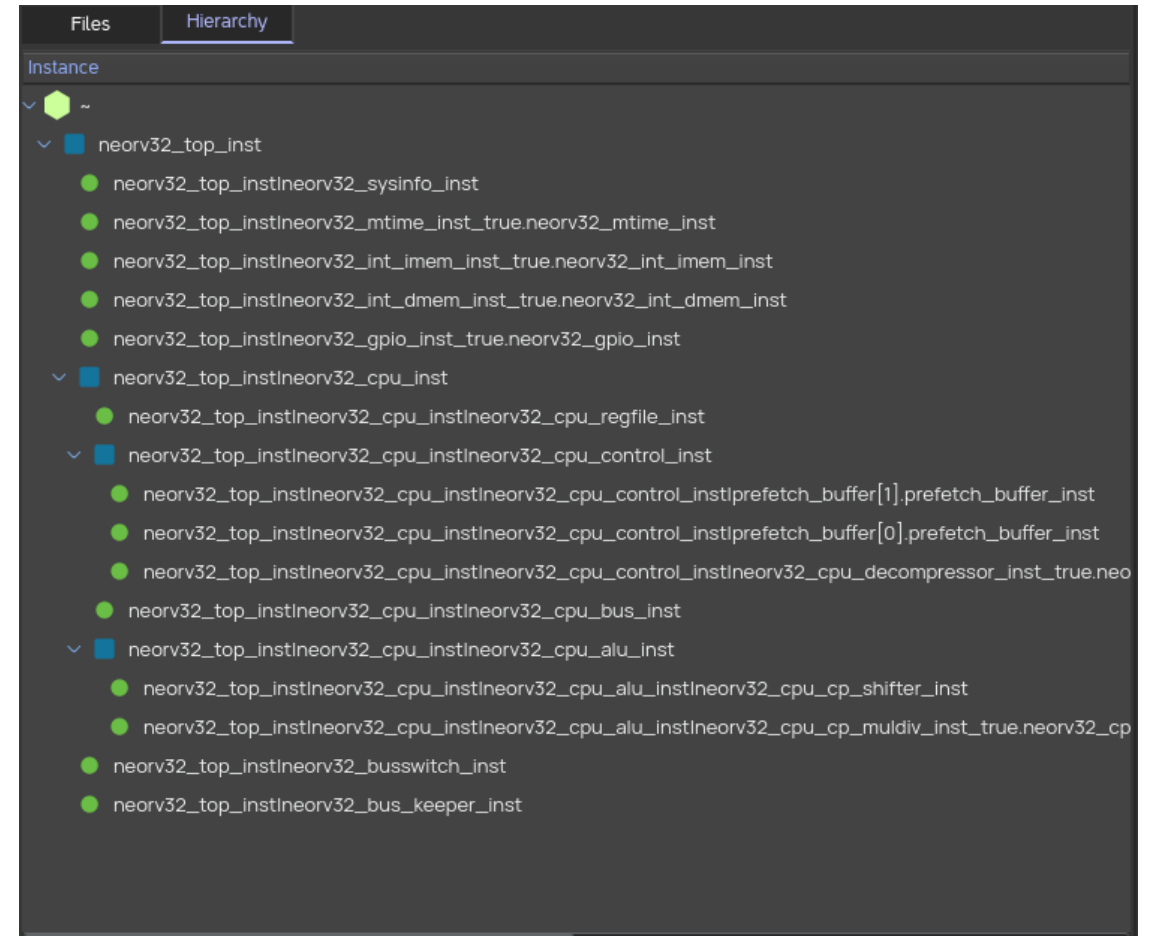
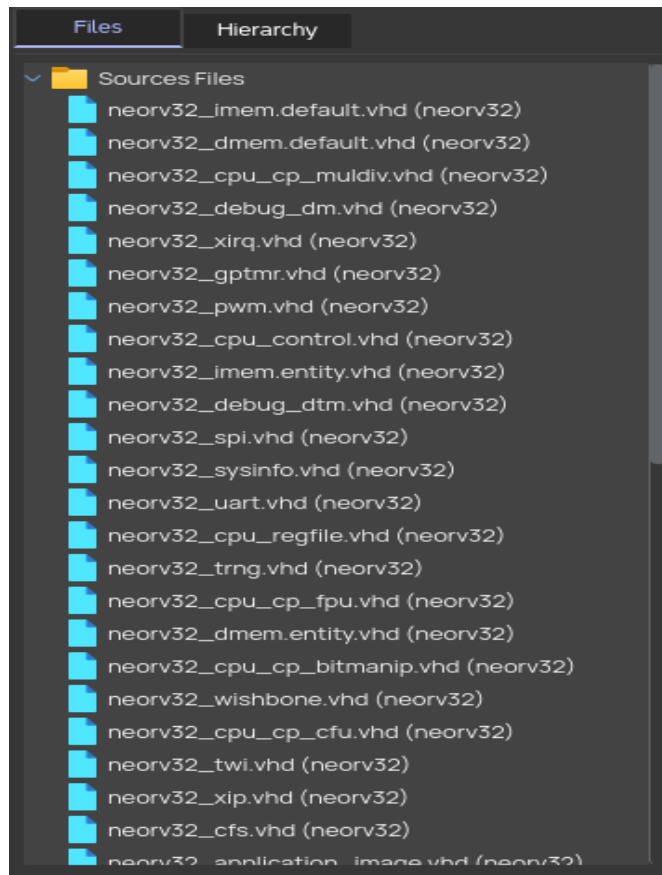
In addition to the device selection, the package can be specified.



This window appears when project settings have been done.
Then you can launch the full flow or a step of the flow.



Hierarchy Project information is generated once the synthesis has been done.



Flow Navigator

The Flow Navigator widget displays the progress of the design flow. It is divided into four main sections: SYNTHESIS, PLACE, ROUTE, and BITSTREAM. Each section shows a 'Complete' status with a green checkmark and a progress indicator (e.g., 3/3 for SYNTHESIS, 5/5 for PLACE, 3/3 for ROUTE). A context menu is open over the SYNTHESIS section, showing options like 'Edit options', 'Run a sub step', 'Run Synthesis', 'Generate Netlist', and 'Run a STA'. The 'Run Synthesis' option is expanded, showing sub-steps: 'Start Synthesis step 1/3', 'Start Synthesis step 2/3', and 'Start Synthesis step 3/3'.

Reports

The Reports widget displays a list of reports generated during the design process. The reports are organized into categories: Design, STA, and Messages. The table below shows the report names and their last modification times.

Report Name	Last Modification
Unconnected	7:57:51 pm
RegisterSummary	10:49:26 am
Registers	8:2:51 pm
Operators	8:2:51 pm
Memories	8:2:51 pm
Maplogic	8:2:44 pm
Lowskew	10:49:26 am
Latches	7:58:33 pm
Instances	10:49:27 am
HierarchyComplexity	
Hierarchy	8:3:1 pm
Hdlfiles	7:57:52 pm
Hdlanalysis	7:57:49 pm
Fsmachines	8:2:51 pm
Constraints	7:57:52 pm

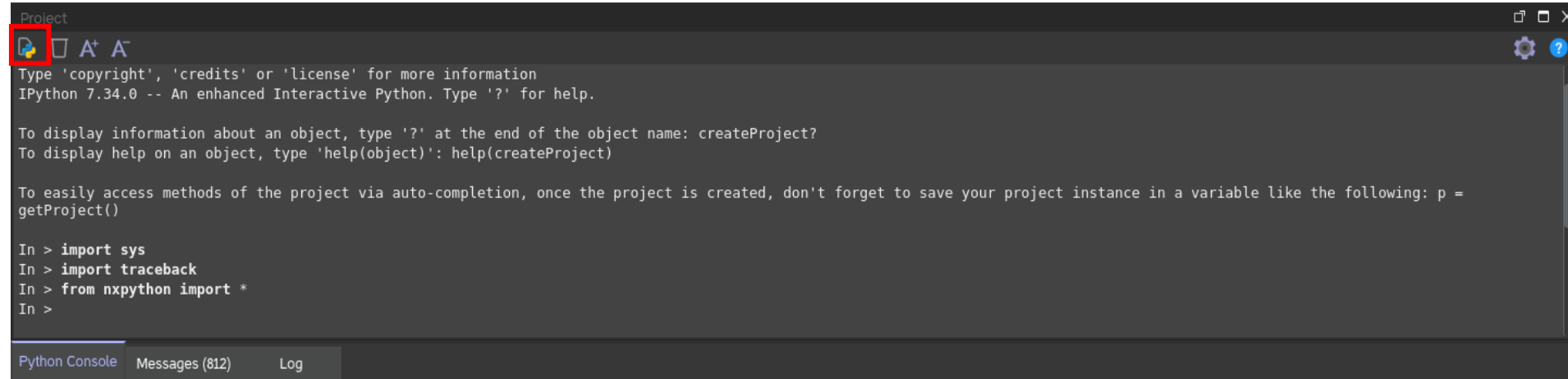
You can « **Edit files** » in Impulse

```

37 -- # The NEORV32 Processor - https://github.com/stnolting/neorv32          (c) Stephan Nolting #
38 -- #####
39
40 library ieee;
41 use ieee.std_logic_1164.all;
42 use ieee.numeric_std.all;
43
44 library neorv32;
45 use neorv32.neorv32_package.all;
46
47 entity neorv32_top is
48   generic (
49     -- General --
50     CLOCK_FREQUENCY      : natural;      -- clock frequency of clk_i in Hz
51     HW_THREAD_ID         : natural := 0;  -- hardware thread id (32-bit)
52     CUSTOM_ID            : std_ulogic_vector(31 downto 0) := x"00000000"; -- custom user-defined ID
53     INT_BOOTLOADER_EN   : boolean := false; -- boot configuration: true = boot explicit bootloader; false = boot from int/ext (I)MEM
54
55     -- On-Chip Debugger (OCD) --
56     ON_CHIP_DEBUGGER_EN : boolean := false; -- implement on-chip debugger
57
58     -- RISC-V CPU Extensions --
59     CPU_EXTENSION_RISCV_B : boolean := false; -- implement bit-manipulation extension?
60     CPU_EXTENSION_RISCV_C : boolean := false; -- implement compressed extension?
61     CPU_EXTENSION_RISCV_E : boolean := false; -- implement embedded RF extension?
62     CPU_EXTENSION_RISCV_M : boolean := false; -- implement mul/div extension?
63     CPU_EXTENSION_RISCV_U : boolean := false; -- implement user mode extension?
64     CPU_EXTENSION_RISCV_Zfinx : boolean := false; -- implement 32-bit floating-point extension (using INT regs!)
65     CPU_EXTENSION_RISCV_Zicsr : boolean := true; -- implement CSR system?
66     CPU_EXTENSION_RISCV_Zicntr : boolean := true; -- implement base counters?
67     CPU_EXTENSION_RISCV_Zihpm : boolean := false; -- implement hardware performance monitors?
68     CPU_EXTENSION_RISCV_Zifencei : boolean := false; -- implement instruction stream sync.?
69     CPU_EXTENSION_RISCV_Zmmul : boolean := false; -- implement multiply-only M sub-extension?
70     CPU_EXTENSION_RISCV_Zxcfu : boolean := false; -- implement custom (instr.) functions unit?
71
72     -- Tuning Options --
73     FAST_MUL_EN : boolean := false; -- use DSPs for M extension's multiplier
74     FAST_SHIFT_EN : boolean := false; -- use barrel shifter for shift operations
75     CPU_CNT_WIDTH : natural := 64; -- total width of CPU cycle and instret counters (0..64)
76     CPU_IPB_ENTRIES : natural := 2; -- entries in instruction prefetch buffer, has to be a power of 2, min 2
77
78     -- Physical Memory Protection (PMP) --
79     PMP_NUM_REGIONS : natural := 0; -- number of regions (0..16)
80     PMP_MIN_GRANULARITY : natural := 4; -- minimal region granularity in bytes, has to be a power of 2, min 4 bytes
81
82   );
83 end entity neorv32_top;
  
```

In **Python Console** you can:

- Launch NXpython commands
- Execute a NXpython script



```

Project
Type 'copyright', 'credits' or 'license' for more information
IPython 7.34.0 -- An enhanced Interactive Python. Type '?' for help.

To display information about an object, type '?' at the end of the object name: createProject?
To display help on an object, type 'help(object)': help(createProject)

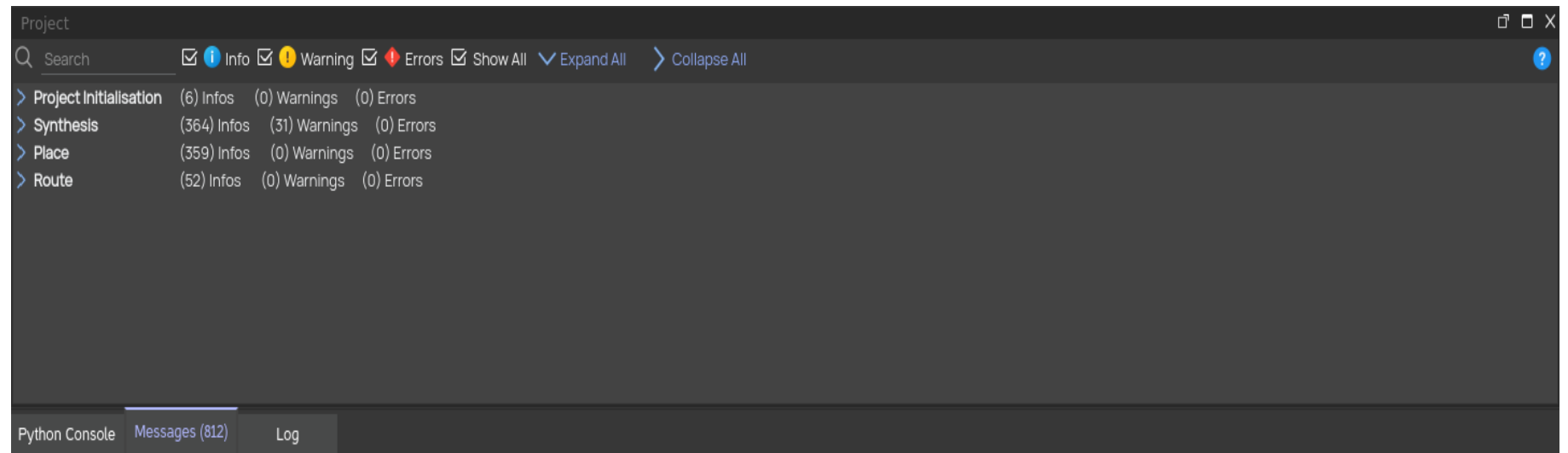
To easily access methods of the project via auto-completion, once the project is created, don't forget to save your project instance in a variable like the following: p = getProject()

In > import sys
In > import traceback
In > from nxpython import *
In >
  
```

Python Console Messages (812) Log

In **Messages** field you can check at each step generated:

- Information
- Warnings
- Errors



```

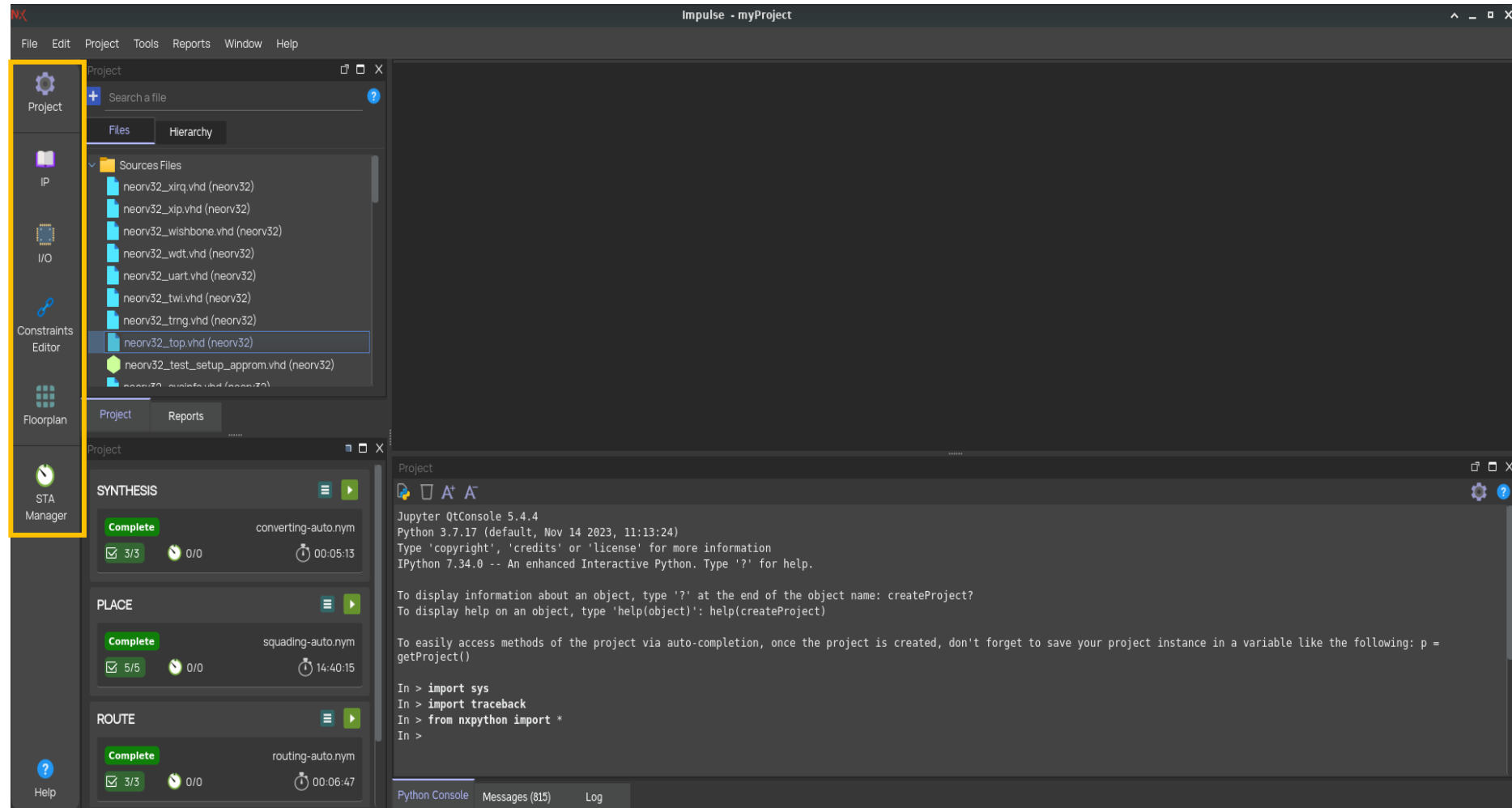
Project
Search [x] Info [x] Warning [x] Errors [x] Show All [v] Expand All [v] Collapse All [v]

> Project Initialisation (6) Infos (0) Warnings (0) Errors
> Synthesis (364) Infos (31) Warnings (0) Errors
> Place (359) Infos (0) Warnings (0) Errors
> Route (52) Infos (0) Warnings (0) Errors
  
```

Python Console Messages (812) Log

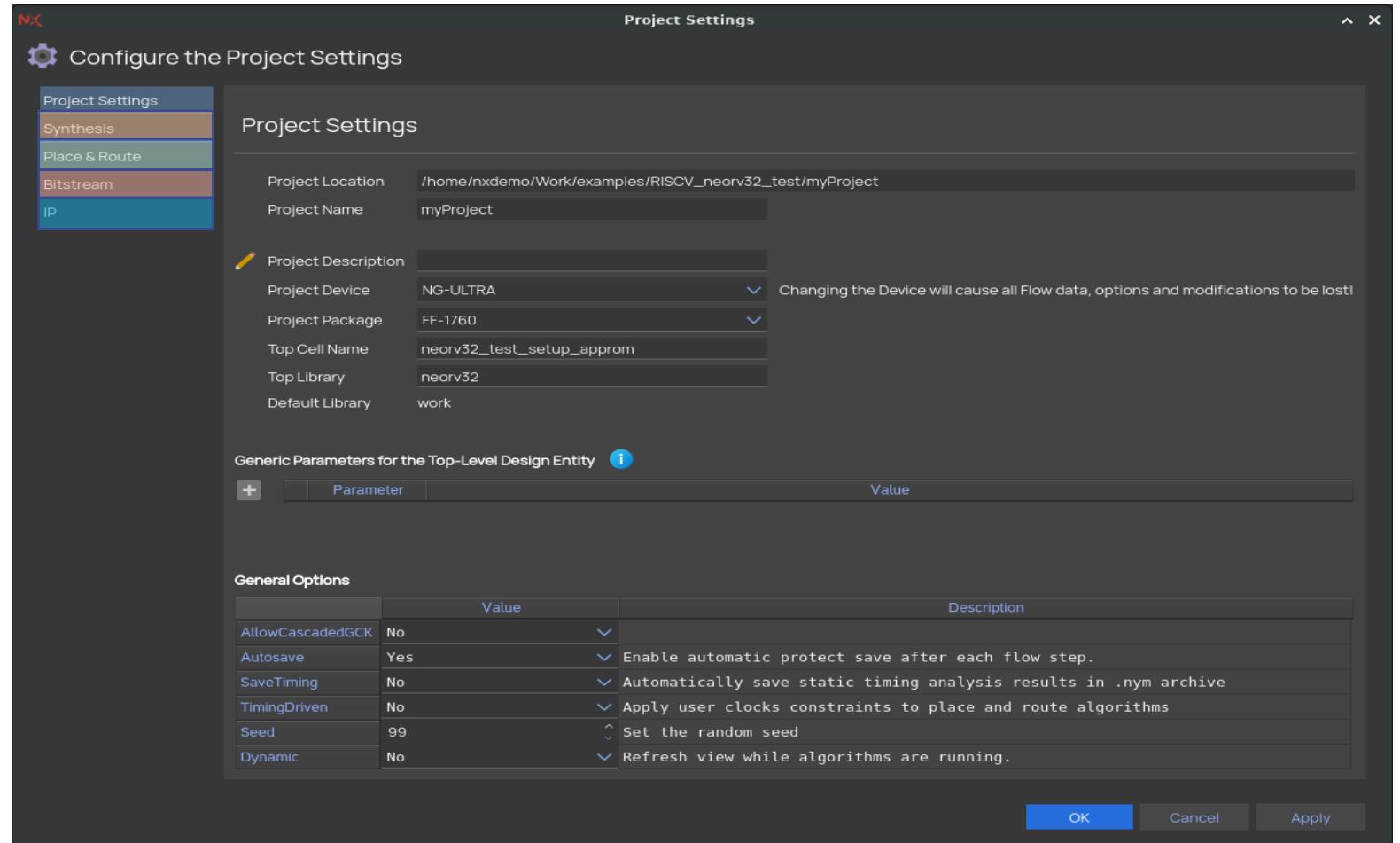
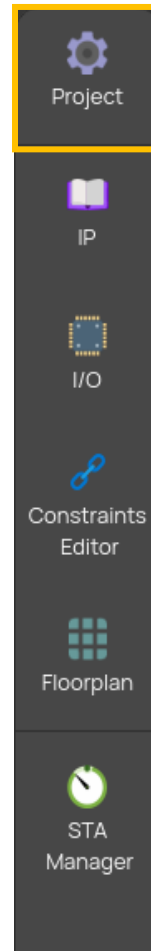
On Impulse interface, the widgets on the left gives you access to:

- Project Settings
- NXcore IP Catalog
- IOs Configuration
- STA Constraints Editor
- Floorplan Viewer
- STA Manager



Depending on the step in the flow, it is possible to modify the project settings

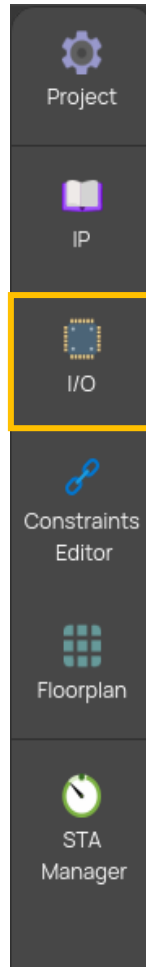
- Update options
- Add generic parameters



The IP Catalog shows the list of IPs available for a specific device. In this case the device concerned is NG-ULTRA.

Name	Variant	Provider	License
Interface			
Spw RX v1.0.0	NG-LARGE NG-MEDIUM NG-ULTRA	NanoXplore	Included
SpaceWire RMAP v2.1.0	NG-LARGE NG-MEDIUM NG-ULTRA	NanoXplore	Purchase
hssl_u_spacefiber_64	NG-LARGE NG-MEDIUM NG-ULTRA	NanoXplore	Purchase
hssl_u_spacefiber_simu_...	NG-ULTRA	NanoXplore	Purchase
wrapper_apb_i2c v0.0.1	NG-LARGE NG-MEDIUM NG-ULTRA	NanoXplore	Purchase
nx_spw_u	NG-ULTRA	NanoXplore	Purchase
Clock			
Clock	NG-ULTRA	NanoXplore	Included
Debug			
ScopeV2	NG-MEDIUM NG-LARGE NG-ULTRA	NanoXplore	Included

Depending on the step in the flow, you can update the ring configuration for:



I/O Config Banks_CKG Config

Filters Search...

HDL Name	Location	Standard	Drive	Weak Termination	SlewRate	Termination	Input Delay Line	Output Delay Line
clk_i	IOB5_D09P	LVC MOS	2mA	PullUp	Medium		0	0
gpio_o[0]	IOB7_D16	LVC MOS	2mA	PullUp	Medium		0	0
gpio_o[1]	IOB7_D17	LVC MOS	2mA	PullUp	Medium		0	0
gpio_o[2]	IOB7_D18	LVC MOS	2mA	PullUp	Medium		0	0
gpio_o[3]	IOB7_D19	LVC MOS	2mA	PullUp	Medium		0	0
gpio_o[4]	IOB7_D20	LVC MOS	2mA	PullUp	Medium		0	0
gpio_o[5]	IOB7_D21	LVC MOS	2mA	PullUp	Medium		0	0
gpio_o[6]	IOB7_D22	LVC MOS	2mA	PullUp	Medium		0	0
gpio_o[7]	IOB7_D23	LVC MOS	2mA	PullUp	Medium		0	0
rstn_i	IOB7_D01	LVC MOS	2mA	PullUp	Medium		0	0

IOs

I/O Editor Banks_CKG Config

Bank Name	Voltage	Type	HDL Name	Location
1 IOB0	3.3V	Direct	1 wfg_B_clk_i	CKG4-WFG-WFG_C1
2 IOB1	3.3V	Direct		
3 IOB10	1.8V	Complex		
4 IOB11	1.8V	Complex		
5 IOB12	1.8V	Complex		
6 IOB13	1.8V	Complex		
7 IOB2	1.8V	Complex		
8 IOB3	1.8V	Complex		
9 IOB4	1.8V	Complex		
10 IOB5	1.8V	Complex		
11 IOB6	3.3V	Direct		
12 IOB7	3.3V	Direct		
13 IOB8	1.8V	Complex		

Banks

The constraints editor allows you to add or remove timing constraints

The screenshot shows the Constraints Editor interface. On the left is a sidebar with navigation options: Project, IP, I/O, Constraints Editor (highlighted), Floorplan, and STA Manager. The main workspace is titled 'Constraints Editor' and contains a 'Create Clock' dialog box. The dialog has a table with the following data:

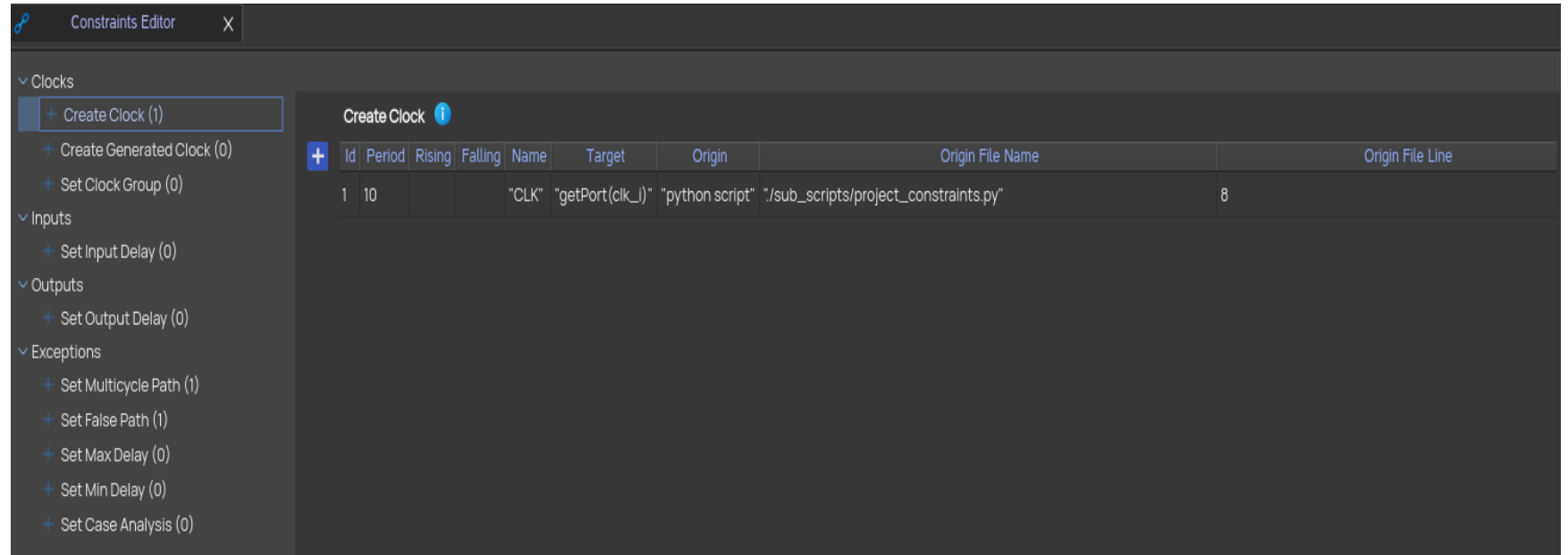
+	Id	Period	Rising	Falling	Name	Target	Origin	Origin File Name	Origin File Line
	1	10			"CLK"	"getPort(clk_i)"	"python script"	./sub_scripts/project_constraints.py	8

Below the dialog is a section titled 'All Project Constraints' with a 'Remove all Constraints' button and a 'Help' dropdown. It contains a table of constraints:

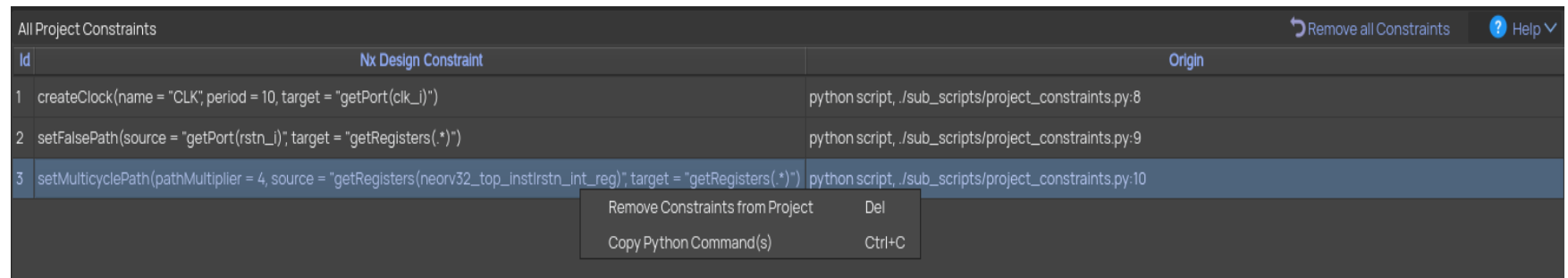
Id	Nx Design Constraint	Origin
1	createClock(name = "CLK", period = 10, target = "getPort(clk_i)")	python script, ./sub_scripts/project_constraints.py:8
2	setFalsePath(source = "getPort(rstn_i)", target = "getRegisters(*)")	python script, ./sub_scripts/project_constraints.py:9
3	setMulticyclePath(pathMultiplier = 4, source = "getRegisters(neorv32_top_inst1rstn_int_reg)", target = "getRegisters(*)")	python script, ./sub_scripts/project_constraints.py:10

You can modify timing constraints before STA launch.

- Create a clock
- Remove a constraint



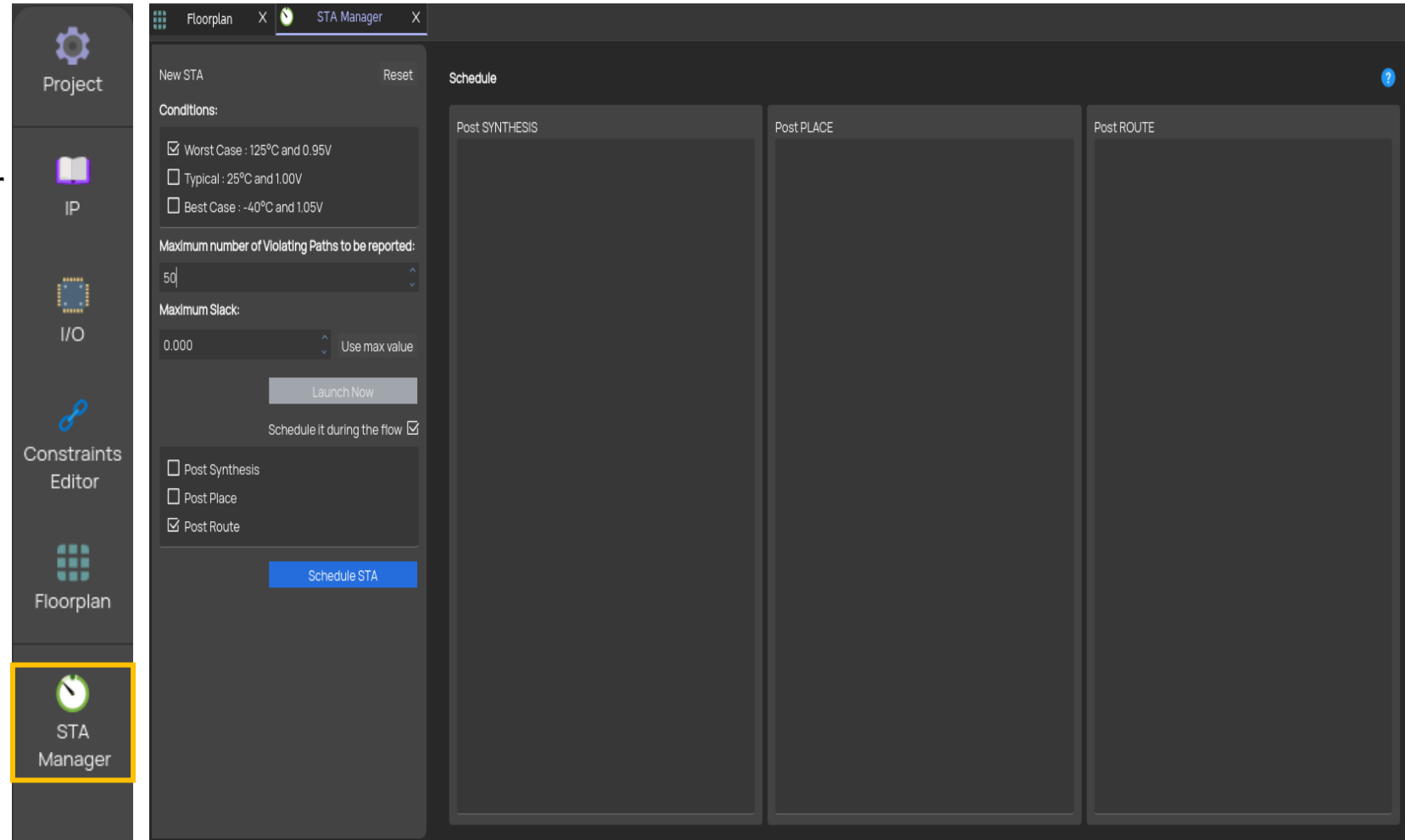
The constraint in python is available and you can copy it in a NXpython script.



In STA Manager you can:

- Schedule STA after each step.
 - Therefore, STA can be launched after
 - Synthesis
 - Place
 - Route

- Specify the STA conditions
 - Worst Case
 - Typical
 - Best Case



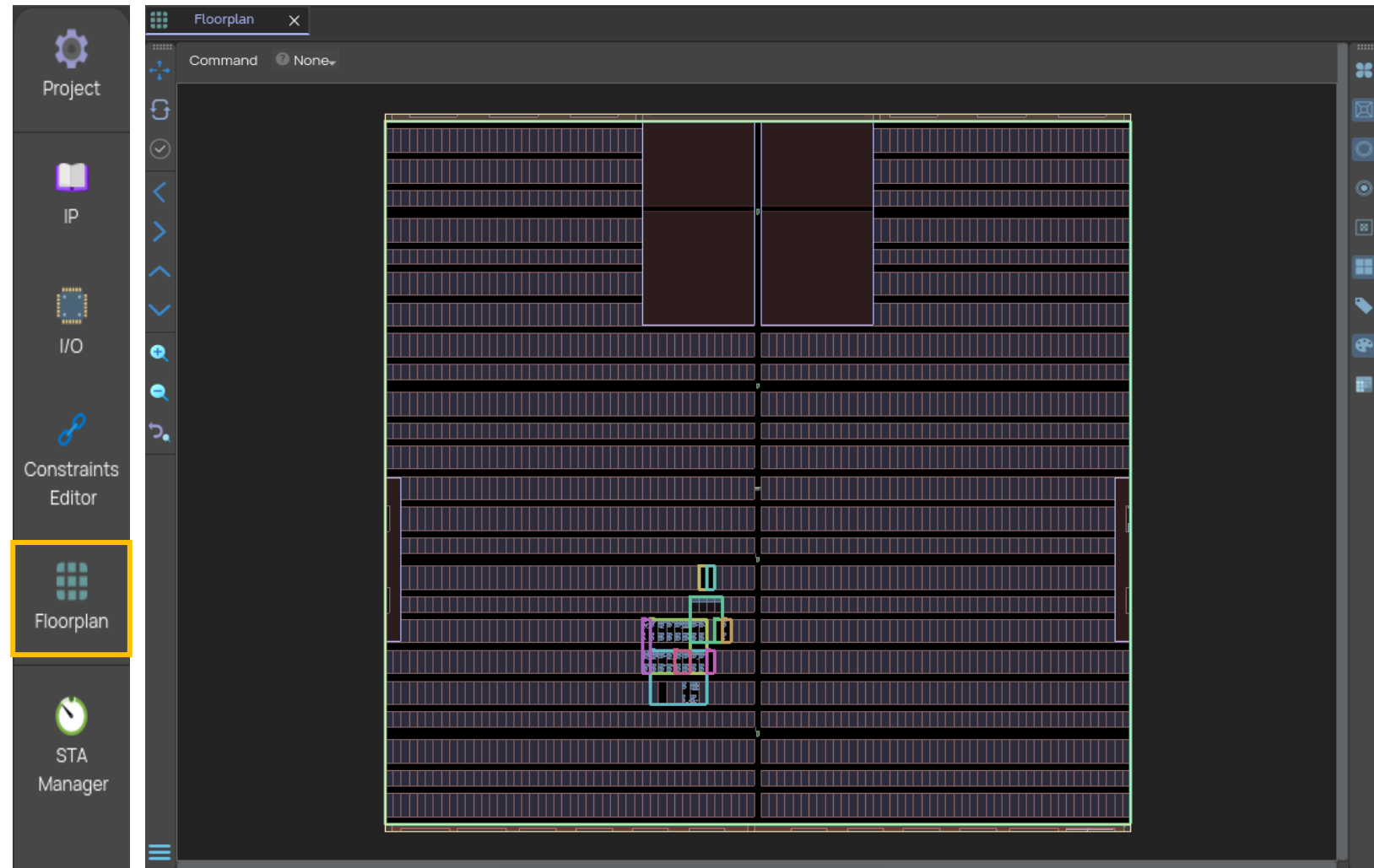
You can schedule as many STAs as you want through the GUI.

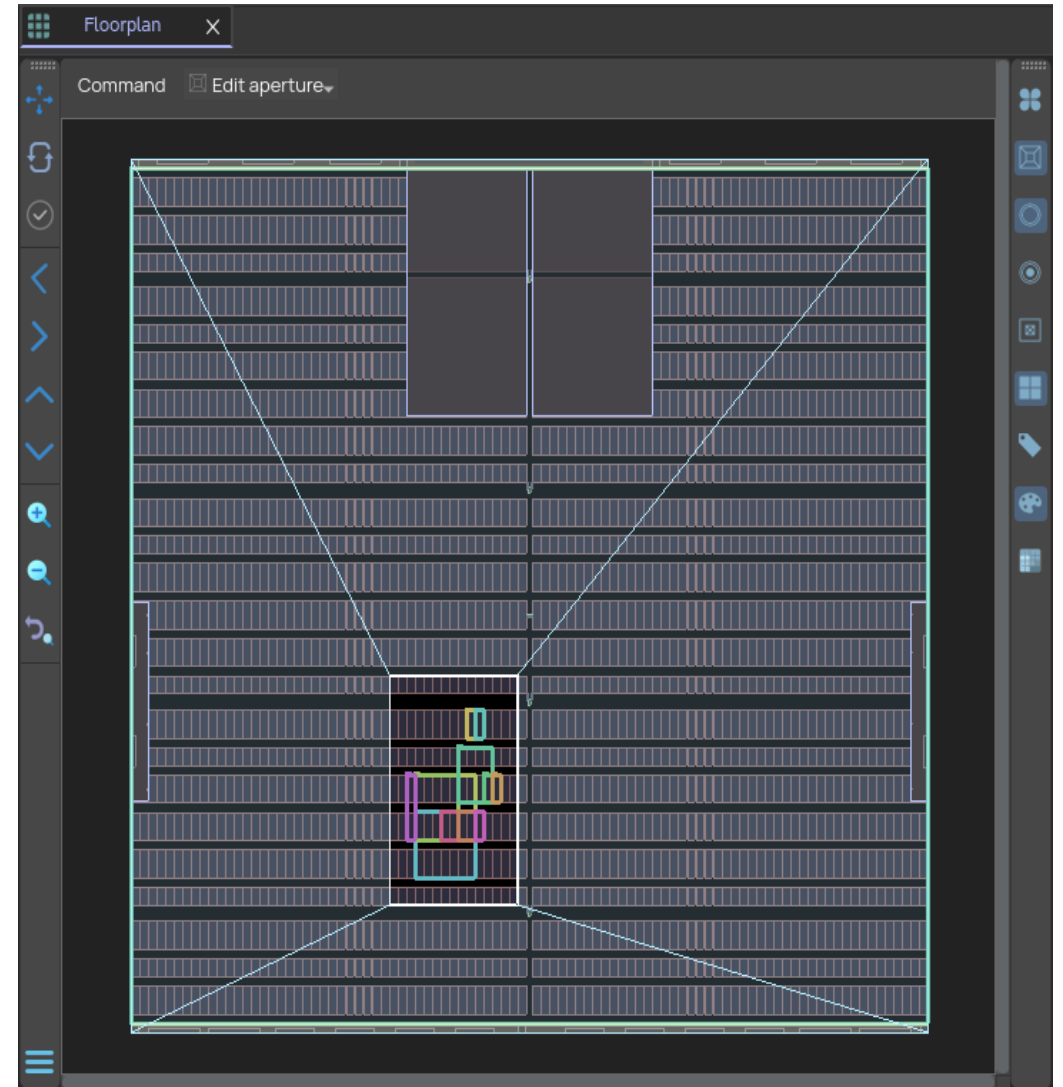
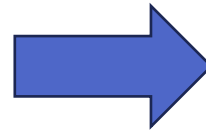
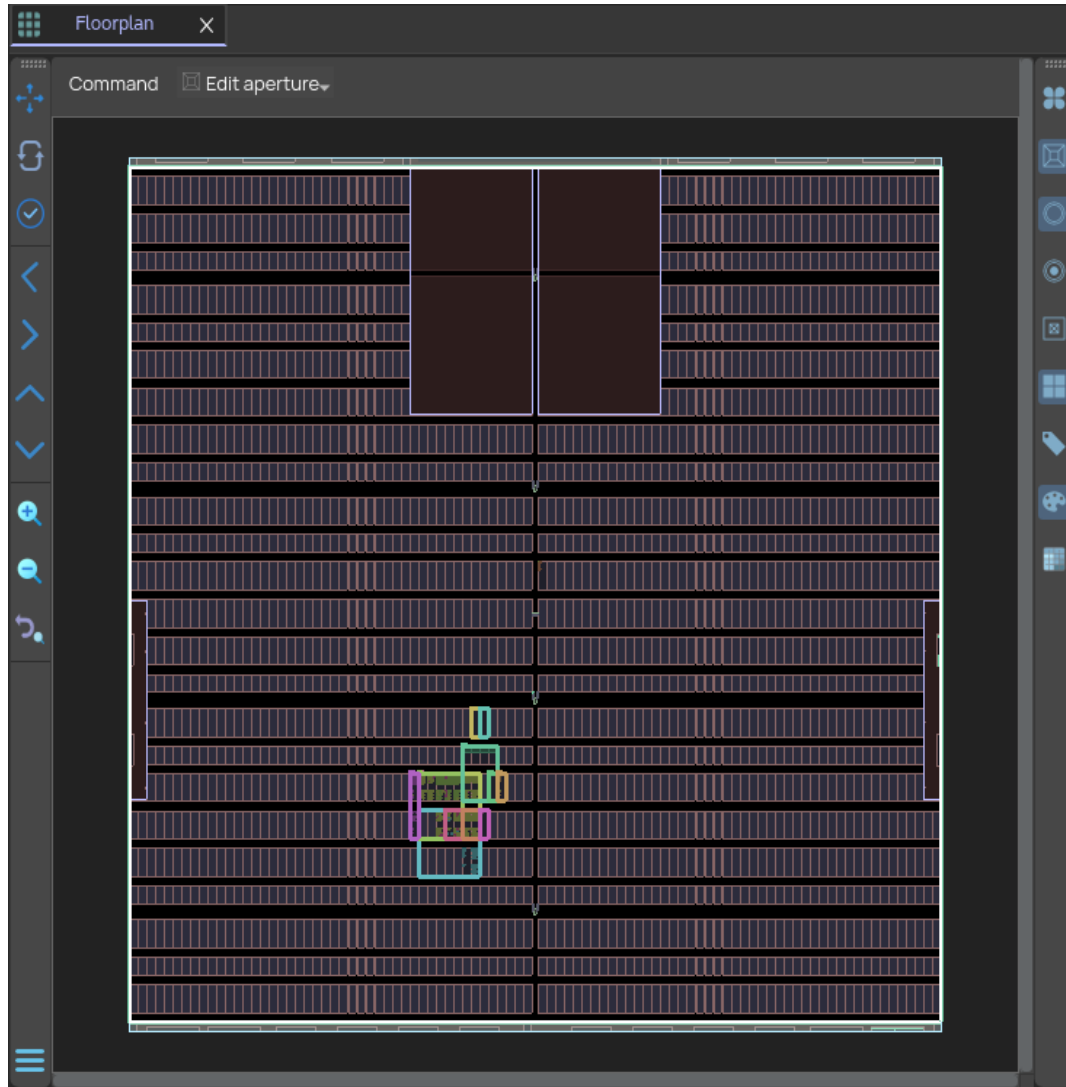
The screenshot displays the STA Manager interface with the following configuration and scheduling details:

- New STA Configuration:**
 - Conditions: Worst Case : 125°C and 0.95V, Typical : 25°C and 1.00V, Best Case : -40°C and 1.05V
 - Maximum number of Violating Paths to be reported: 30
 - Maximum Slack: 100.000 (Use max value)
 - Launch Now button
 - Schedule it during the flow
 - Post Synthesis, Post Place, Post Route
 - Schedule STA button
- Schedule Panel:**
 - Post SYNTHESIS:**
 - S Conditions: *worstcase*
 - T Maximum number of Violating Paths: 10
 - A Maximum Slack: 100
 - Post PLACE:**
 - S Conditions: *typical, worstcase*
 - T Maximum number of Violating Paths: 20
 - A Maximum Slack: 100
 - Post ROUTE:**
 - S Conditions: *worstcase*
 - T Maximum number of Violating Paths: 30
 - A Maximum Slack: 100

Through the floorplan viewer you can:

- Set global Aperture
- Create/Edit Regions
- View Nets
- View Instances
- View Paths
- View interconnexions between regions
- Display the routing Congestion Map





The screenshot displays the NanoXplore Floorplan tool interface. On the left, a grid-based floorplan is shown with several regions highlighted in different colors (yellow, green, purple, cyan). The right-hand pane, titled 'Tools', shows a 'Region' view with a table of statistics for various modules and clusters.

Region/Obstruction	Module/Cluster	Name	Tone	Aperture	Focus	TILEs	CGBs	LUTs	DFFs	CYs	RF
~	~	~	11	1x1 w:92 h:49	47x24	1316	672	4571(0.90%)	1382(0.27%)	142(0.45%)	0(0.00%)
~	~	~						8	5	-	-
~	~	~						4563	1377	142	-
~	neorv32_cpu_cp_muldiv_R	neorv32_cpu_cp_muldiv_R	15	34x36 w:5 h:3	38x36	10	-	651(16.95%)	173(4.51%)	29(12.08%)	0(0.00%)
~	neorv32_cpu_cp_muldiv_M	neorv32_cpu_cp_muldiv_M						651	173	29	-
~	neorv32_cpu_alu_R	neorv32_cpu_alu_R	2	37x38 w:2 h:1	38x38	2	-	111(14.45%)	39(5.08%)	2(4.17%)	0(0.00%)
~	neorv32_cpu_alu_M	neorv32_cpu_alu_M						111	39	2	-
~	neorv32_cpu_regfile_R	neorv32_cpu_regfile_R	9	34x38 w:7 h:3	40x38	14	-	511(9.51%)	64(1.19%)	26(7.74%)	0(0.00%)
~	neorv32_cpu_regfile_M	neorv32_cpu_regfile_M						511	64	26	-
~	neorv32_cpu_control_R	neorv32_cpu_control_R	16	34x36 w:7 h:3	40x36	14	-	2484(46.21%)	665(12.37%)	50(14.88%)	0(0.00%)
~	neorv32_cpu_control_M	neorv32_cpu_control_M						2484	665	50	-
~	neorv32_cpu_R	neorv32_cpu_R	3	34x36 w:7 h:5	40x36	21	-	154(1.91%)	103(1.28%)	0(0.00%)	0(0.00%)
~	neorv32_cpu_M	neorv32_cpu_M						154	103	-	-
~	neorv32_busswitch_R	neorv32_busswitch_R	10	41x32 w:1 h:1	41x32	1	-	18(4.69%)	8(2.08%)	0(0.00%)	0(0.00%)
~	neorv32_busswitch_M	neorv32_busswitch_M						18	8	-	-
~	neorv32_bus_keeper_R	neorv32_bus_keeper_R	17	40x32 w:1 h:1	40x32	1	-	21(5.47%)	12(3.13%)	0(0.00%)	0(0.00%)
~	neorv32_bus_keeper_M	neorv32_bus_keeper_M						21	12	-	-
~	neorv32_imem_R	neorv32_imem_R	4	41x38 w:1 h:1	41x38	1	-	36(9.38%)	32(8.33%)	0(0.00%)	0(0.00%)
~	neorv32_imem_M	neorv32_imem_M						36	32	-	-

Selection
Logic Cone Manager

The screenshot shows the NanoXplore Floorplan tool interface. The 'Select nets' menu is open, showing options for selection and edition. The main floorplan view shows a complex circuit layout with various components and connections. The right-hand pane displays a list of selected nets, including their IDs and names, along with a detailed view of the selected net (ID: 1770).

Command: Select nets

Selection: Select nets, Select instances, Select pipes, Select paths

Edition: Edit aperture, Edit focus, View obstructions, Edit regions

Tools: Region, Selection

Combine: Replace, Select Single, Count: 6071, Selected: 1

ID: *

Name: *

ID	Name
1770	neorv32_top_inst neorv32_cpu_inst rs1[26]
1771	neorv32_top_inst neorv32_c...rv32_cpu_control_inst n783
1772	neorv32_top_inst neorv32_...e.neorv32_mtime_inst n525
1773	neorv32_top_inst neorv32_c...32_cpu_cp_muldiv_inst n104
1774	LOGIC n4471
1775	neorv32_top_inst neorv32_c... neorv32_cpu_alu_inst n186
1776	LOGIC n5218
1777	LOGIC n3904
1778	LOGIC n6737
1779	neorv32_top_inst neorv32_g...rue.neorv32_gpio_inst n238
1780	LOGIC n4341

ID: 1770
Name: neorv32_top_inst|neorv32_cpu_inst|rs1[26]
Fanout: 11

2 instances of module neorv32_cpu_cp_muldiv_M in region neorv32_cpu_cp_muldiv_R
 1 instance of module neorv32_cpu_alu_M in region neorv32_cpu_alu_R
 6 instances of module neorv32_cpu_regfile_M in region neorv32_cpu_regfile_R
 3 instances of module neorv32_cpu_control_M in region neorv32_cpu_control_R

Logic Cone Manager

The screenshot displays the NanoXplore Floorplan interface. The 'Command' menu is open, showing the 'Select Instances' option. The main floorplan area shows a grid of tiles with various components highlighted in different colors. The 'Tools' panel on the right shows the 'Selection' tab with a list of selected instances.

Command Menu:

- Default
- None
- Selection
 - Select nets
 - Select instances
- Edition
 - Select pipes
 - Select paths
 - Edit aperture
 - Edit focus
 - View obstructions
 - Edit regions

Tools Panel - Selection:

Combine Replace Select Single Count: 9404 Selected: 1

ID	Type	Location	Name
9403	LUT		neorv32_top_inst neorv32_cpu_inst neorv32_cpu_control_inst trap_ctr1[env_start]_reg~csc~lut
9401	LUT		neorv32_top_inst neorv32_cpu_inst neorv32_cpu_control_inst ctrl_reg[35]~csc~lut
9399	LUT		neorv32_top_inst neorv32_cpu_inst neorv32_cp...t execute_engine_reg[state].execute~csc~lut
9397	LUT		neorv32_top_inst neorv32_cpu_inst neorv32_cp...ntrol_inst fetch_engine[restart]_reg~csc~lut
9395	LUT		neorv32_top_inst neorv32_cpu_inst neorv32_cp...p_multdiv_inst ctrl_reg[state].s_idle~csc~lut
9393	LUT		LOGIC lut_2668[D]~csc~lut
9391	LUT		neorv32_top_inst neorv32_cpu_inst neorv32_cp...er[1].prefetch_buffer_inst lut_42[D]~csc~lut
9389	LUT		LOGIC lut_3566[D]~csc~lut
9387	LUT		neorv32_top_inst neorv32_cpu_inst neorv32_cp...er[0].prefetch_buffer_inst lut_41[D]~csc~lut
9385	LUT		neorv32_top_inst neorv32_cpu_inst neorv32_cpu_control_inst lut_29[D]~csc~lut

Instance Details:

Name: neorv32_top_inst|neorv32_cpu_inst|neorv32_cpu_control_inst|trap_ctr1[env_start]_reg~csc~lut
 Module: neorv32_cpu_control1_M
 Region: neorv32_cpu_control1_R
 Location: TILE[36x38]:S3.LUT65

Equation: $0 \leq I1$

I1: neorv32_top_inst|neorv32_cpu_inst|neorv32_cpu_control_inst|trap_ctr1[env_start]
 I2: '0'
 I3: '0'
 I4: '0'
 0: neorv32_top_inst|neorv32_cpu_inst|neorv32_cpu_control_inst|trap_ctr1[env_start]

Logic Cone Manager

The screenshot displays the NanoXplore Floorplan interface. The 'Command' menu is open, showing the 'Select paths' option. The main floorplan area shows a circuit diagram with several paths highlighted in different colors (yellow, green, purple, cyan). The 'Logic Cone Manager' table on the right lists the source and target nodes for the selected paths.

Command Menu:

- Default
- None
- Selection
 - Select nets
 - Select instances
 - Select pipes
 - Select paths
- Edition
 - Edit aperture
 - Edit focus
 - View obstructions
 - Edit regions

Logic Cone Manager Table:

Source	Target
neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK	neorv32_top_inst neorv32_mtime_inst_true.neorv32_mtime_in
neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK	neorv32_top_inst neorv32_mtime_inst_true.neorv32_mtime_in
neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK	neorv32_top_inst neorv32_mtime_inst_true.neorv32_mtime_in
neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK	neorv32_top_inst neorv32_mtime_inst_true.neorv32_mtime_in
neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK	neorv32_top_inst neorv32_mtime_inst_true.neorv32_mtime_in
neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK	neorv32_top_inst neorv32_mtime_inst_true.neorv32_mtime_in
neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK	neorv32_top_inst neorv32_mtime_inst_true.neorv32_mtime_in
neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK	neorv32_top_inst neorv32_mtime_inst_true.neorv32_mtime_in
neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK	neorv32_top_inst neorv32_mtime_inst_true.neorv32_mtime_in
neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK	neorv32_top_inst neorv32_mtime_inst_true.neorv32_mtime_in

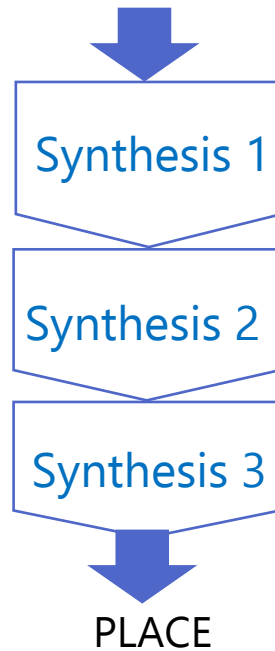
The screenshot displays the NanoXplore Floorplan interface. A context menu is open over the floorplan, with the 'Select pipes' option highlighted. The menu includes sections for Default, Selection, and Edition. The 'Logic Cone Manager' table on the right provides a detailed view of the selected pipes.

Rank	Nets	Source	Target
76	130	REG1_R	neorv32_mtime_R
75	43	REG1_R	neorv32_gpio_R
74	44	REG1_R	neorv32_dmem_R
73	92	REG1_R	neorv32_imem_R
72	8	REG1_R	neorv32_busswitch_R
71	2	REG1_R	neorv32_cpu_R
70	1353	REG1_R	neorv32_cpu_control_R
69	2	neorv32_sysinfo_R	neorv32_bus_keeper_R
68	5	neorv32_sysinfo_R	neorv32_cpu_R
67	11	neorv32_sysinfo_R	neorv32_cpu_control_R
66	4	neorv32_mtime_R	neorv32_gpio_R
65	1	neorv32_mtime_R	neorv32_imem_R
64	2	neorv32_mtime_R	neorv32_bus_keeper_R
63	1	neorv32_mtime_R	neorv32_busswitch_R
62	11	neorv32_mtime_R	neorv32_cpu_R
61	121	neorv32_mtime_R	neorv32_cpu_control_R
60	1	neorv32_gpio_R	neorv32_sysinfo_R
59	1	neorv32_gpio_R	neorv32_mtime_R
58	2	neorv32_gpio_R	neorv32_bus_keeper_R
57	1	neorv32_gpio_R	neorv32_busswitch_R
56	11	neorv32_gpio_R	neorv32_cpu_R
55	39	neorv32_gpio_R	neorv32_cpu_control_R
54	8	neorv32_gpio_R	~

Synthesis

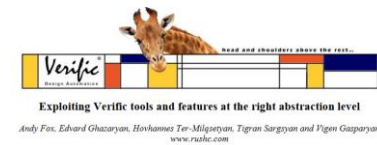


FPGA synthesis is the process of converting a design into Netlist. From HDL code (VHDL, Verilog, etc.) to dedicated resources into FPGA.



- Analysis & Elaboration

- Verilog IEEE 1164
- VHDL IEEE-1076 standard (2008, 1993, 1987)



- Mapping into NX resources (NX_LUT, NX_DFF, NX_RAM, NX_DSP...)

- Replacing BlackBoxes by Preplaced IPs

Post-Synthesis STA

- Logs
 - `general.log` : contains all messages printed during the Synthesis step
- Reports
 - `hierarchy.rpt` : Reports the hierarchy of the design
 - `constraints.rpt` : Reports the unused constraints and the constraints that were not applied
 - `memories.rpt` : Reports the information about memories used in the design
 - `operators.rpt` : Reports all operators (adder, subtractor, multiplier, comparator...) used in the design
- You can generate at synthesis stage:
 - Netlist in VHDL or Verilog

All synthesis steps are saved in different `.nym` files which could be loaded by the GUI and Nxpypython method.

The design resources is available after Synthesis.

You can find design resources information in `general.log`.

```
Reporting instances at state 'Synthesized':
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 4-LUT | DFF | XLUT | 1 - bit | Register | Cross | Clock | Clock | Digital | Memory | WFG | PLL | GCK |
|       |     |     | Carry  | file     | domain | Buffer | switch | signal | block  |     |     |     |
|       |     |     |        | block   | clock  |       |       | processor |       |       |     |     |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 3588/505344 (1%) | 1377/505344 (1%) | 0/31584 (0%) | 531/126336 (1%) | 4/2632 (1%) | 0/2632 (0%) | 0 | 0/5264 (0%) | 0/1344 (0%) | 4/672 (1%) | 0/70 (0%) | 0/7 (0%) | 0/160 (0%) |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

The design hierarchy is available after Synthesis.

```

-----
HDL Modules Hierarchy Detail
-----

~          neorv32_test_setup_approm [ ~ ]
Resources:
  NX_LUT : 3587
  NX_DFF : 1377
  NX_CY  : 142
  NX_RFB_U : 2
  NX_RAM : 4
  NX_IOB : 10
  NX_BFR : 19
  NX_WFG_U : 1

~          |-> neorv32#neorv32_top(X8381227E) [ neorv32_top_inst ]
          |-> Resources:
          |->  NX_LUT : 2546
          |->  NX_DFF : 1377
          |->  NX_CY  : 142
          |->  NX_RFB_U : 2
          |->  NX_RAM : 4
  
```

Files
Hierarchy

Instance

- ~
- neorv32_top_inst
 - neorv32_top_instneorv32_sysinfo_inst
 - neorv32_top_instneorv32_mtime_inst_true.neorv32_mtime_inst
 - neorv32_top_instneorv32_int_imem_inst_true.neorv32_int_imem_inst
 - neorv32_top_instneorv32_int_dmem_inst_true.neorv32_int_dmem_inst
 - neorv32_top_instneorv32_gpio_inst_true.neorv32_gpio_inst
 - neorv32_top_instneorv32_cpu_inst
 - neorv32_top_instneorv32_cpu_instneorv32_cpu_regfile_inst
 - neorv32_top_instneorv32_cpu_instneorv32_cpu_control_inst
 - neorv32_top_instneorv32_cpu_instneorv32_cpu_bus_inst
 - neorv32_top_instneorv32_cpu_instneorv32_cpu_alu_inst

Detailed info of each **operator**.

Operators (8) | Operators Mapping (21) | Notifications (0)

Search...

Name

- LessThan_32u_32u
 - Instances (2)

Name	File Origin	Line Origin
LessThan_cmp_lo_ge	../src/neorv32_mtime.vhd	196
LessThan_cmp_hi_gt	../src/neorv32_mtime.vhd	198
- LessThan_33u_33u
 - Instances (1)

Name	File Origin	Line Origin
LessThan_cmp_o	../src/neorv32_cpu_alu.vhd	104
- add_2u_2u
 - Instances (3)

Name	File Origin	Line Origin
add_L113	../src/neorv32_fifo.vhd	113
add_L119	../src/neorv32_fifo.vhd	119
add_level_diff	../src/neorv32_fifo.vhd	132
- add_4u_4u
 - Instances (1)

Name	File Origin	Line Origin
add_L179	../src/neorv32_bus_keeper.vhd	179

```

-----
- Operator Cells analysis -
-----

Operator 'LessThan_32u_32u'
  : LessThan_cmp_lo_ge      (line 196 in ../src/neorv32_mtime.vhd)
  : LessThan_cmp_hi_gt     (line 198 in ../src/neorv32_mtime.vhd)

Operator 'LessThan_33u_33u'
  : LessThan_cmp_o        (line 104 in ../src/neorv32_cpu_alu.vhd)

Operator 'add_2u_2u'
  : add_L113              (line 113 in ../src/neorv32_fifo.vhd)
  : add_L119              (line 119 in ../src/neorv32_fifo.vhd)
  : add_level_diff        (line 132 in ../src/neorv32_fifo.vhd)

Operator 'add_4u_4u'
  : add_L179              (line 179 in ../src/neorv32_bus_keeper.vhd)
  
```

Detailed info of each **memory**.

Mem (17) Mem Mapped (5) Notifications (0)							
Search...							
Name	Type	Depth	Width	Instance Name	Read Type	Async	Generated
✓ neorv32#neorv32_cpu_regfile(X5BDBBAD2)_reg_file	Ram	32	32	reg_file	ReadAfterWrite	false	true
▼ Ports (2)							
Addr Size	Din Size	Dout Size	Dout Type	Id	Wr		
5	32	32	RS	0	true		
5	0	32	RS	1	false		

```

-----
- Memory Cells analysis
-----

HDL 'neorv32#neorv32_cpu_regfile(X5BDBBAD2)_reg_file' Raw description seems to be 'Read after Write'
Physical implementation in RF/RAM may not follow it exactly without additional logic

Ram 'neorv32#neorv32_cpu_regfile(X5BDBBAD2)_reg_file' Analysis:
Port 0 :
    Slc Size: 0
    Addr Size: 5
    Din Size: 32 (W)
    Dout Size: 32 (RS)

Port 1 :
    Slc Size: 0
    Addr Size: 5
    Din Size: 0
    Dout Size: 32 (RS)

Array Depth 32
Array Width 32
  
```

You can **force mapping** in specific resources

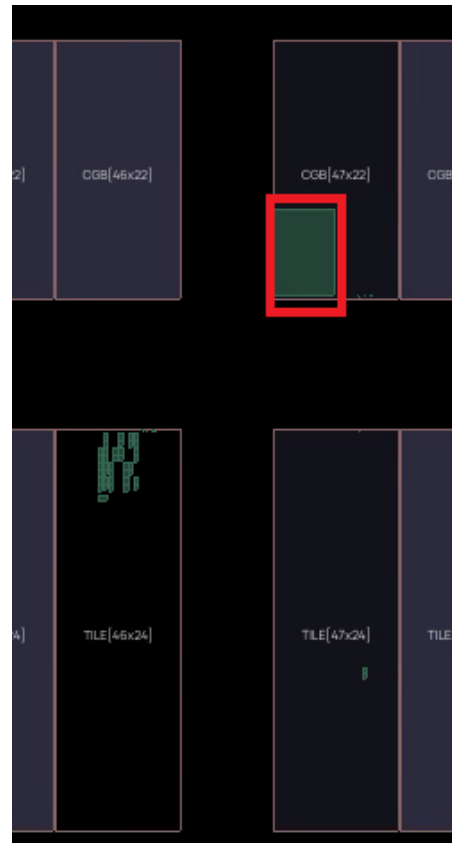
	Type	Mapping targets
Operators	ADD	LUT, CY, DSP
	LTN	LUT, CY, DSP
	MUL	CY,DSP
Memories	RAM	RF, RAM, RAM_ECC, DFF
	ROM	LUT, RF, RAM

The floorplan view of same **operator cell** mapped into different resources.

Adder mapped into CY



Adder mapped into DSP



Adder mapped into LUT



The floorplan view of **memory cell** mapped into different resources.

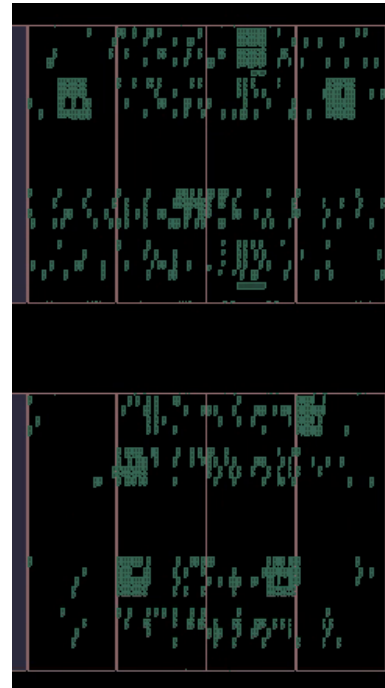
RAM mapped into RF



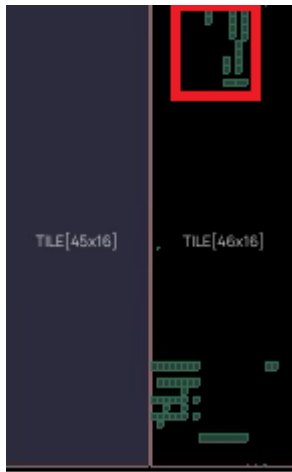
RAM mapped into RAM block



RAM mapped into DFF



ROM mapped into LUT



Post-synthesis STA is done at the end of synthesis

This static timing analysis helps to know if the frequency requested can be reached.

Static Timing Analysis Conditions: typical
 Voltage: 1.20V
 Temperature: 25°C

Summary of DOMAIN_clk25m_to_clk25m

Domain		Frequency		Hold/Removal Summary			Setup/Recovery Summary		
Source	Target	Required	Actual	Slack	Minimum Data Arrival Time	Minimum Required Relationship	Slack	Maximum Data Arrival Time	Maximum Required Relationship
clk25m (Falling)	clk25m (Rising)	-	-	15.165ns	2.246ns	-12.919ns	9.323ns	3.596ns	12.919ns
clk25m (Rising)	clk25m (Falling)	-	38.703 MHz	14.310ns	1.391ns	-12.919ns	0ps	12.919ns	12.919ns
clk25m (Rising)	clk25m (Rising)	-	-	787ps	787ps	0ps	10.945ns	14.893ns	25.838ns
Total									3

Reporting longest paths for DOMAIN_clk25m_to_clk25m

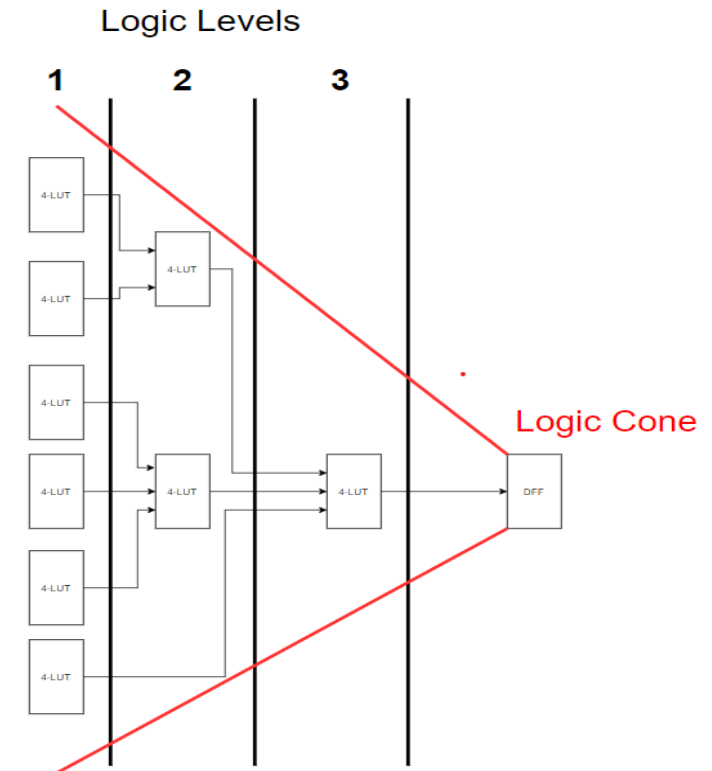
No.	Slack	Source	Target	Data Delay	Clock Skew	Setup/Recovery	Depth	Note
1	0ps	Pin: inst_AHBLite AHBLite_0 matrix4x16 masterstage_0 SDATASELInt_reg[4].CK	Pin: inst_CoreMemCtrl ISRAMWEN_reg.I	12.682ns	-10ps	227ps	14	(RF)
2	14ps	Pin: inst_AHBLite AHBLite_0 matrix4x16 masterstage_0 SDATASELInt_reg[5].CK	Pin: inst_CoreMemCtrl ISRAMWEN_reg.I	12.668ns	-10ps	227ps	14	(RF)
3	80ps	Pin: inst_AHBLite AHBLite_0 matrix4x16 masterstage_0 SDATASELInt_reg[6].CK	Pin: inst_CoreMemCtrl ISRAMWEN_reg.I	12.602ns	-10ps	227ps	14	(RF)
4	97ps	Pin: inst_AHBLite AHBLite_0 matrix4x16 masterstage_0 SDATASELInt_reg[7].CK	Pin: inst_CoreMemCtrl ISRAMWEN_reg.I	12.585ns	-10ps	227ps	14	(RF)
5	1.041ns	Pin: inst_AHBLite AHBLite_0 matrix4x16 masterstage_0 SDATASELInt_reg[1].CK	Pin: inst_CoreMemCtrl ISRAMWEN_reg.I	11.641ns	-10ps	227ps	13	(RF)
6	1.041ns	Pin: inst_AHBLite AHBLite_0 matrix4x16 masterstage_0 SDATASELInt_reg[1].CK	Pin: inst_CoreMemCtrl ISRAMWEN_reg.I	11.641ns	-10ps	227ps	13	(RF)
7	1.055ns	Pin: inst_AHBLite AHBLite_0 matrix4x16 masterstage_0 SDATASELInt_reg[2].CK	Pin: inst_CoreMemCtrl ISRAMWEN_reg.I	11.627ns	-10ps	227ps	13	(RF)
8	1.107ns	Pin: inst_AHBLite AHBLite_0 matrix4x16 masterstage_0 SDATASELInt_reg[2].CK	Pin: inst_CoreMemCtrl ISRAMWEN_reg.I	11.575ns	-10ps	227ps	13	(RF)
9	1.107ns	Pin: inst_AHBLite AHBLite_0 matrix4x16 masterstage_0 SDATASELInt_reg[4].CK	Pin: inst_CoreMemCtrl ISRAMWEN_reg.I	11.575ns	-10ps	227ps	13	(RF)
10	1.121ns	Pin: inst_AHBLite AHBLite_0 matrix4x16 masterstage_0 SDATASELInt_reg[5].CK	Pin: inst_CoreMemCtrl ISRAMWEN_reg.I	11.561ns	-10ps	227ps	13	(RF)
Total							10	

Logic levels depend on the capacity of elements to perform an operation.

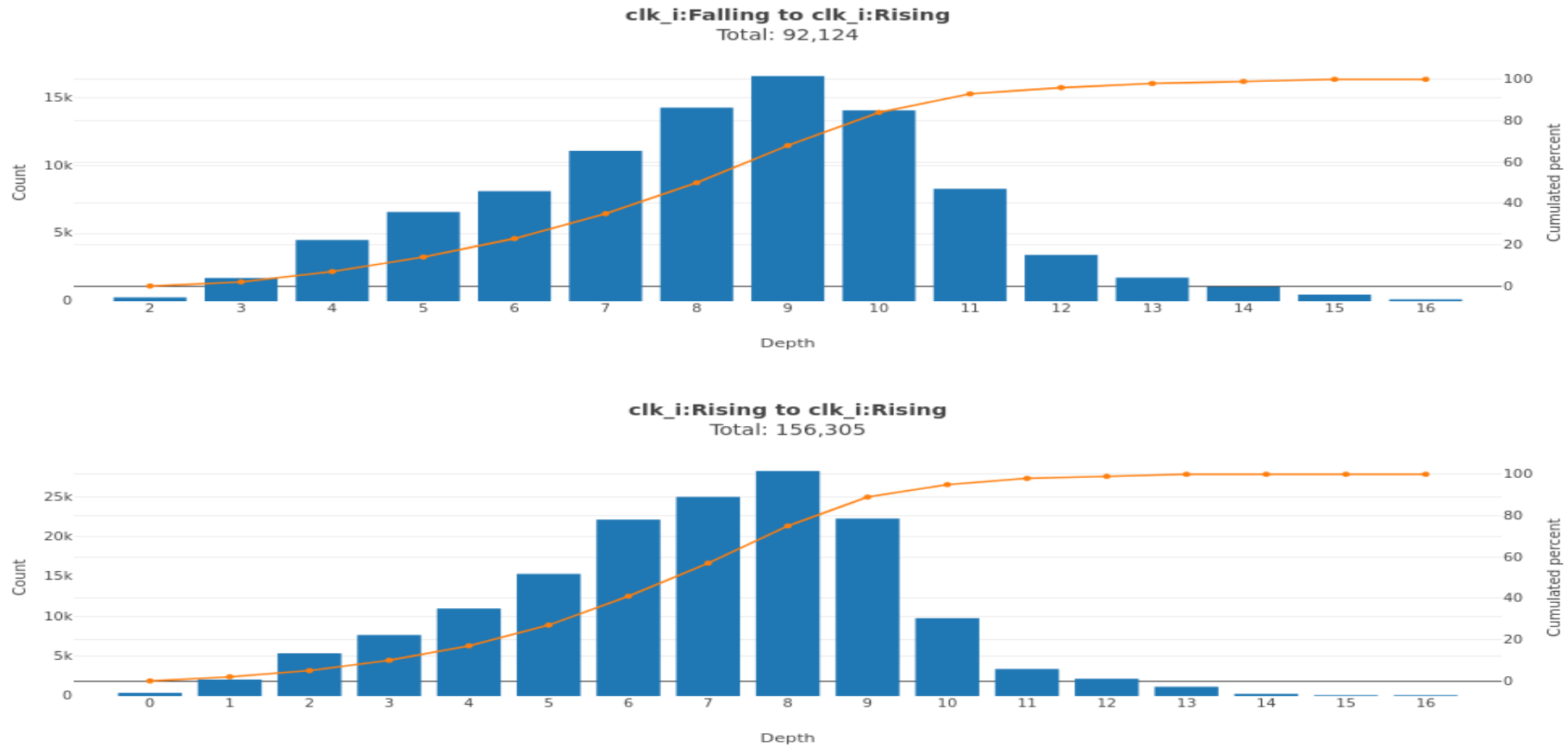
The lower the operational efficiency of a resource is, the higher its logic level will be high.

This creates **Logical depth**.

All the elements interconnected to generate one bit create one **Logic Cone**.



Design complexity charts illustrates repartition of paths based on logic depth by clock domain.



Place



SYNTHESIS



Place 1

- Apply placement constraints

Place 2

- Place Lowskew Network

Place 3

- Place cells
- Place regions
- Clustering

Place 4

- Post-Place
- Optimize the placement by area

Place 5

- Pre-route

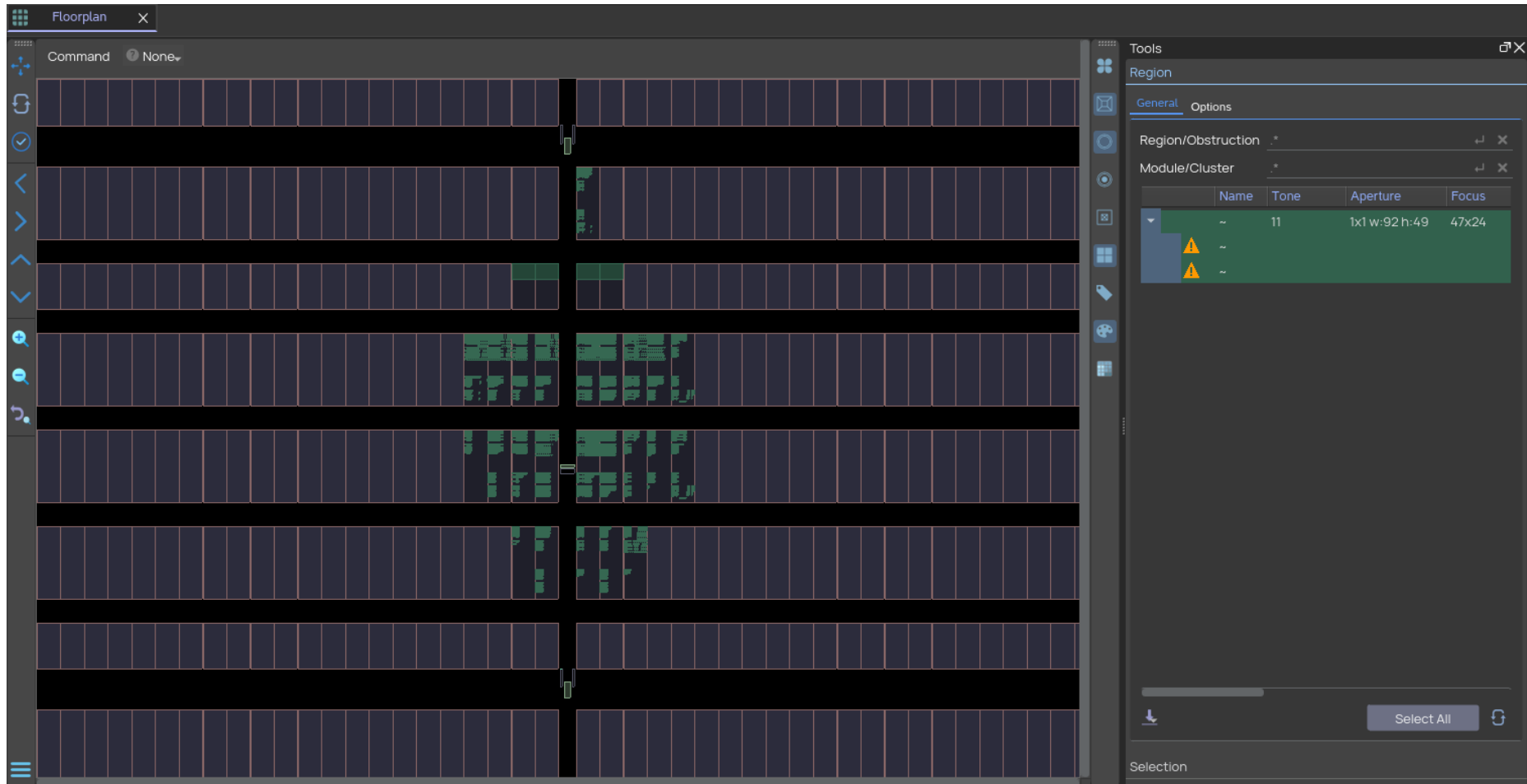


ROUTE

- Logs
 - `general.log` : contains all messages printed during the Place step.
- Reports
 - `ios.rpt` : Reports the IOs of the design
 - `lowskew.rpt` : Reports all the lowskew network statistics at each step of the flow
 - `regions.rpt` : Reports the regions created of the design
- You can generate at place stage:
 - Netlist in VHDL or Verilog

All place steps are saved in different `.nym` files which could be loaded by the GUI and NXpython command.

Automatic placement without floorplanning



To optimize performances, the design can be splitted into several regions.

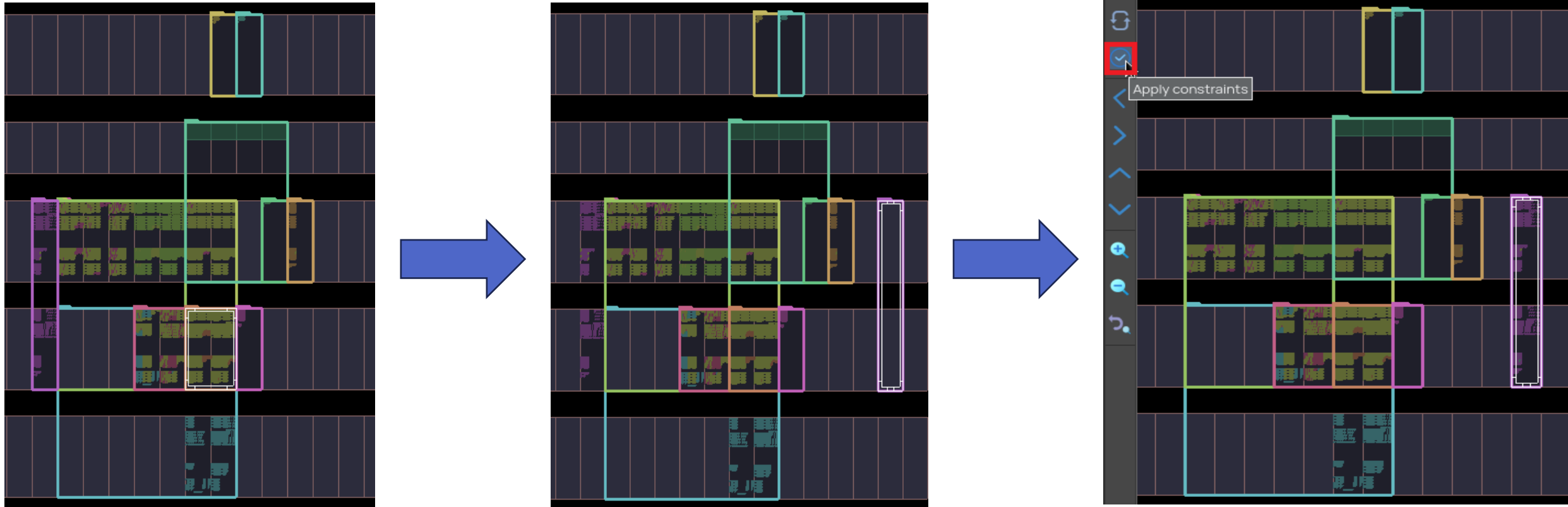
Regions can be easily edited through the GUI or by NXpython method.

Regions are based on `hierarchy.rpt` file.

The screenshot shows the NX NanoXplore floorplanning interface. The main window displays a 49x40 tile grid with various colored regions (yellow, green, purple, orange) overlaid on a dark background. A 'Tools' panel on the right displays a table of region definitions.

Name	colMin	rowMin	width	height
REGION[REG1_R]	39	38	2	1
REGION[neorv32_bus_keeper_R]	40	32	1	1
REGION[neorv32_busswitch_R]	41	32	1	1
REGION[neorv32_cpu_R]	34	36	7	5
REGION[neorv32_cpu_alu_R]	37	38	2	1
REGION[neorv32_cpu_control_R]	34	36	7	3
REGION[neorv32_cpu_cp_muldiv_R]	34	36	5	3
REGION[neorv32_cpu_regfile_R]	34	38	7	3
REGION[neorv32_dmem_R]	39	34	4	3
REGION[neorv32_gpio_R]	43	36	1	1
REGION[neorv32_imem_R]	41	38	1	1
REGION[neorv32_mtime_R]	33	36	1	3
REGION[neorv32_sysinfo_R]	42	36	1	1
REGION[~]	1	1	92	49

To update the region modification, you must click on "Apply constraints" button before continuing the flow.



Once regions are created, you can see:

- The regions in floorplan.
- The resources in each region
- If there is any overflow

The screenshot displays the NanoXplore interface. On the left, a floorplan view shows a grid of tiles with several colored rectangular regions overlaid. On the right, a 'Region' panel provides a detailed resource breakdown for the selected region. The table below summarizes the data shown in the panel.

Name	Tone	Aperture	Focus	TILES	CGBs	LUTs	DFFs	CYs	RF
-	11	1x1 w:92 h:49	47x24	1316	672	4571 (0.90%)	1382 (0.27%)	142 (0.45%)	0(0.00%)
-	-	-	-	-	-	8	5	-	-
-	-	-	-	-	-	4563	1377	142	-
neorv32_cpu_cp_muldiv_R	15	34x36 w:5 h:3	38x36	10	-	651 (16.95%)	173 (4.51%)	29 (12.08%)	0(0.00%)
neorv32_cpu_cp_muldiv_M	-	-	-	-	-	651	173	29	-
neorv32_cpu_alu_R	2	37x38 w:2 h:1	38x38	2	-	111 (14.45%)	39 (5.08%)	2 (4.17%)	0(0.00%)
neorv32_cpu_alu_M	-	-	-	-	-	111	39	2	-
neorv32_cpu_regfile_R	9	34x38 w:7 h:3	40x38	14	-	511 (9.51%)	64 (1.19%)	26 (7.74%)	0(0.00%)
neorv32_cpu_regfile_M	-	-	-	-	-	511	64	26	-
neorv32_cpu_control_R	16	34x36 w:7 h:3	40x36	14	-	2484 (46.21%)	665 (12.37%)	50 (14.88%)	0(0.00%)
neorv32_cpu_control_M	-	-	-	-	-	2484	665	50	-
neorv32_cpu_R	3	34x36 w:7 h:5	40x36	21	-	154 (1.91%)	103 (1.28%)	0 (0.00%)	0(0.00%)
neorv32_cpu_M	-	-	-	-	-	154	103	-	-
neorv32_busswitch_R	10	41x32 w:1 h:1	41x32	1	-	18 (4.69%)	8 (2.08%)	0 (0.00%)	0(0.00%)
neorv32_busswitch_M	-	-	-	-	-	18	8	-	-
neorv32_bus_keeper_R	17	40x32 w:1 h:1	40x32	1	-	21 (5.47%)	12 (3.13%)	0 (0.00%)	0(0.00%)
neorv32_bus_keeper_M	-	-	-	-	-	21	12	-	-
neorv32_mem_R	4	41x38 w:1 h:1	41x38	1	-	36 (9.38%)	32 (8.33%)	0 (0.00%)	0(0.00%)
neorv32_mem_M	-	-	-	-	-	36	32	-	-

How many connections there are between two regions?

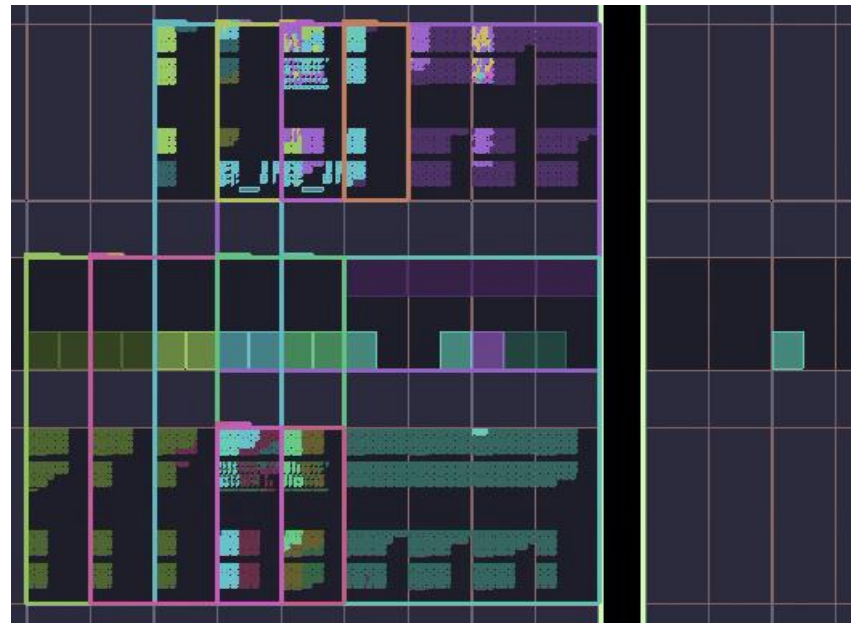
The screenshot shows the NanoXplore interface with a circuit board layout on the left and a 'Tools' panel on the right. The 'Tools' panel is set to 'Region' and 'Selection' mode, showing a list of connections between two regions. The table below is a transcription of the data shown in the 'Tools' panel.

Rank	Nets	Source	Target
76	130	REG1_R	neorv...ime_R
75	43	REG1_R	neorv...pio_R
74	44	REG1_R	neorv...em_R
73	92	REG1_R	neorv...em_R
72	8	REG1_R	neorv...tch_R
71	2	REG1_R	neorv...cpu_R
70	1353	REG1_R	neorv...rol_R
69	2	neorv32_sysinfo_R	neorv...per_R
68	5	neorv32_sysinfo_R	neorv...cpu_R
67	11	neorv32_sysinfo_R	neorv...rol_R
66	4	neorv32_mtime_R	neorv...pio_R
65	1	neorv32_mtime_R	neorv...em_R
64	2	neorv32_mtime_R	neorv...per_R
63	1	neorv32_mtime_R	neorv...tch_R
62	11	neorv32_mtime_R	neorv...cpu_R
61	121	neorv32_mtime_R	neorv...rol_R
60	1	neorv32_gpio_R	neorv...nfo_R
59	1	neorv32_gpio_R	neorv...ime_R
58	2	neorv32_gpio_R	neorv...per_R
57	1	neorv32_gpio_R	neorv...tch_R
56	11	neorv32_gpio_R	neorv...cpu_R
55	39	neorv32_gpio_R	neorv...rol_R
54	8	neorv32_gpio_R	~
53	1	neorv32_dmem_R	neorv...per_R
52	22	neorv32_dmem_R	neorv...cpu_R
51	42	neorv32_dmem_R	neorv...rol_R

This feature allows you to:

- Select one or several elements to spot in a specific location.
- Spot one or several elements override the region constraints

SetSite can be used through NXpython method and GUI.



Select Path to check critical paths in the GUI

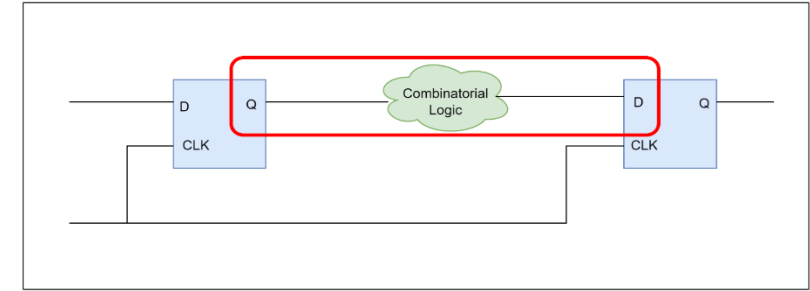
The floorplan shows the critical paths which can be selected by domain and by type (setup and hold).

The screenshot shows the NanoXplore GUI with the 'Select paths' button highlighted in the top toolbar. The floorplan displays several critical paths in green. The right-hand panel shows a table of path details:

Rank	Slack	Source	Target	Domain
1	-101ps	neorv32_top_instlrstn_int_reg.CK	neorv3...[1].CK	CLK_to_CLK
2	1199ns	neorv32_top_instlrstn_int_reg.CK	neorv3...[1].CK	CLK_to_CLK
3	1.712ns	neorv32_top_instneorv32...tro_instictrl_reg[34].CK	neorv3...25].CK	CLK_to_CLK
4	1.712ns	neorv32_top_instneorv32...tro_instictrl_reg[34].CK	neorv3...10].CK	CLK_to_CLK
5	1.712ns	neorv32_top_instneorv32...tro_instictrl_reg[34].CK	neorv3...23].CK	CLK_to_CLK
6	1.712ns	neorv32_top_instneorv32...tro_instictrl_reg[34].CK	neorv3...[3].CK	CLK_to_CLK
7	1.712ns	neorv32_top_instneorv32...tro_instictrl_reg[34].CK	neorv3...[9].CK	CLK_to_CLK
8	1.712ns	neorv32_top_instneorv32...tro_instictrl_reg[34].CK	neorv3...[2].CK	CLK_to_CLK
9	1.712ns	neorv32_top_instneorv32...tro_instictrl_reg[34].CK	neorv3...17].CK	CLK_to_CLK

There is a feature to manage specific critical path between two registers.

To reduce the propagation timing, you can constrain the path in a region.



Rank	Slack	Source
10	-4.948ns	neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK
9	-4.948ns	neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK
8	-4.948ns	neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK
7	-4.948ns	neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK
6	-4.948ns	neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK
5	-4.948ns	neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK
4	-4.948ns	neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK
3	-4.948ns	neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK
2	-4.948ns	neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK
1	-4.948ns	neorv32_top_inst neorv32_busswitch_inst arbiter_reg[state].idle.CK

```

Pin: LOGIC|lut_49.0
Pin: LOGIC|lut_48.I1
Pin: LOGIC|lut_48.0
Pin: LOGIC|lut_47.I1
Pin: LOGIC|lut_47.0
Pin: LOGIC|lut_46.I1
Pin: LOGIC|lut_46.0
Pin: LOGIC|lut_91.I1
Pin: LOGIC|lut_91.0
Pin: LOGIC|lut_90.I1
Pin: LOGIC|lut_90.0
Pin: neorv32_top_inst|neorv32_mtime_inst_true.neorv32_mtime_inst|mtimecmp_hi_reg
Total
  
```

ConstrainPath method allows you to catch all resources between 2 registers and create a specific region to constrain these resources.

Both registers could be in different regions.

The screenshot shows the NanoXplore Floorplan tool interface. The main area displays a grid of tiles labeled from TILE[33x38] to TILE[46x38]. A specific region is highlighted with an orange border, containing several smaller components. The 'Tools' panel on the right is open to the 'Region' tab, showing a table of resources. The table has columns for Name, Tone, Aperture, and Focus. The resource REG1_M is highlighted in brown, indicating it is selected.

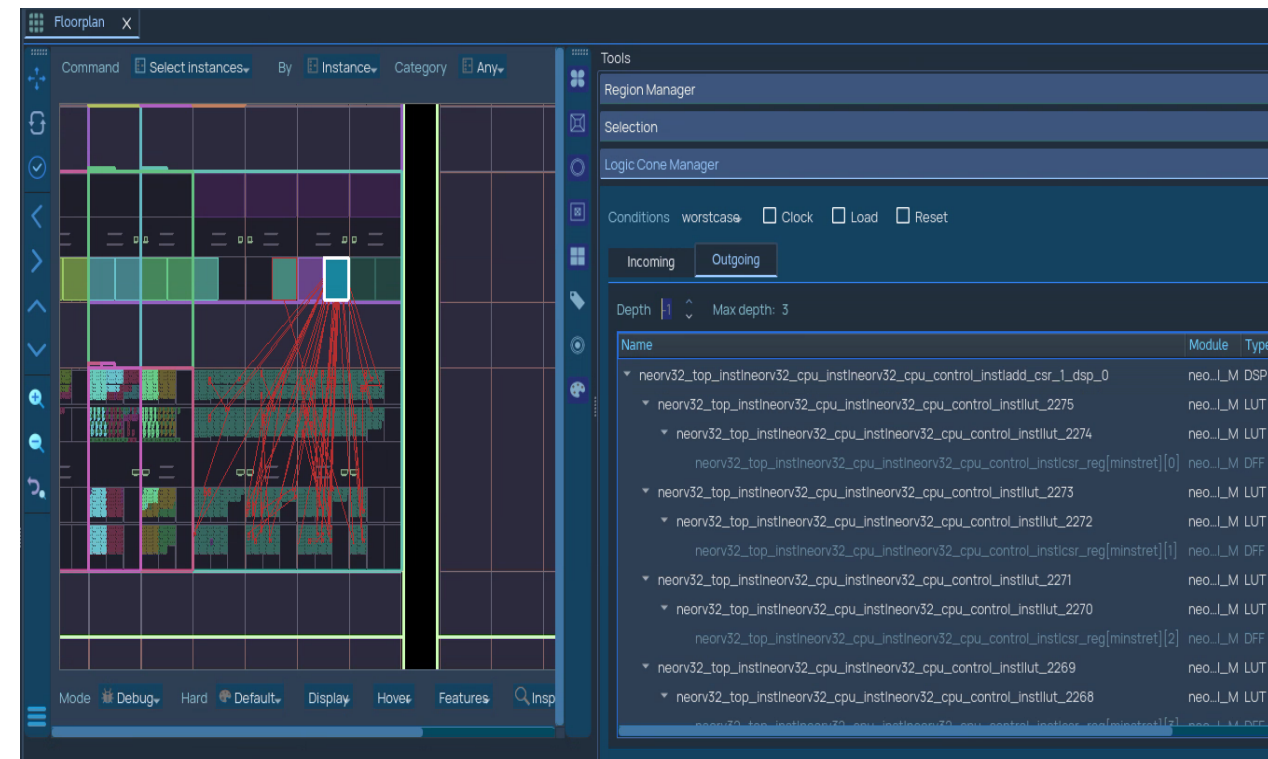
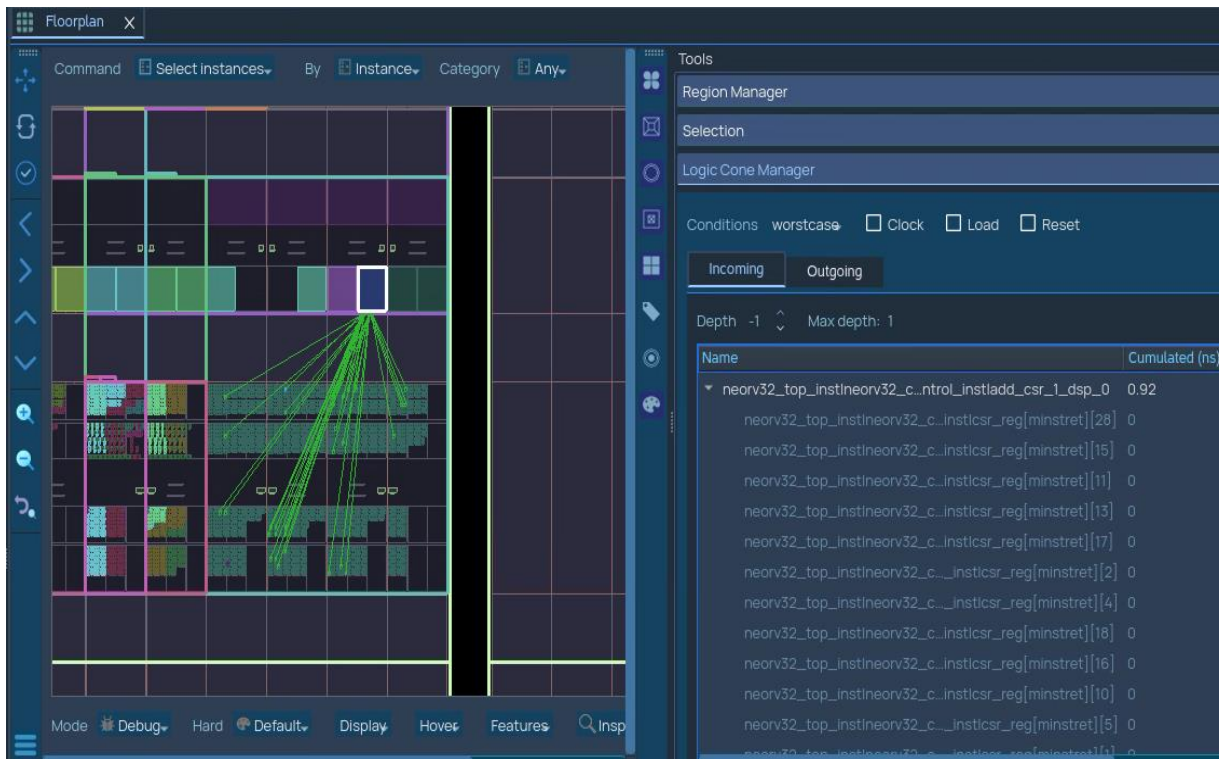
Name	Tone	Aperture	Focus
neorv32_imem_M			
neorv32_dmem_R	11	39x34 w:4 h:3	42x34
neorv32_dmem_M			
neorv32_gpio_R	18	43x36 w:1 h:1	43x36
neorv32_gpio_M			
neorv32_mtime_R	5	33x36 w:1 h:3	33x36
neorv32_mtime_M			
neorv32_sysinfo_R	12	42x36 w:1 h:1	42x36
neorv32_sysinfo_M			
REG1_R	19	39x38 w:2 h:1	40x38
REG1_M			

The Logic Cone Interface is only available in GUI.

It must be only used for **test** or when the **code is fixed**.

The Logic Cone Interface applies SetSite constraint on each element.

What is Logic Cone Interface?

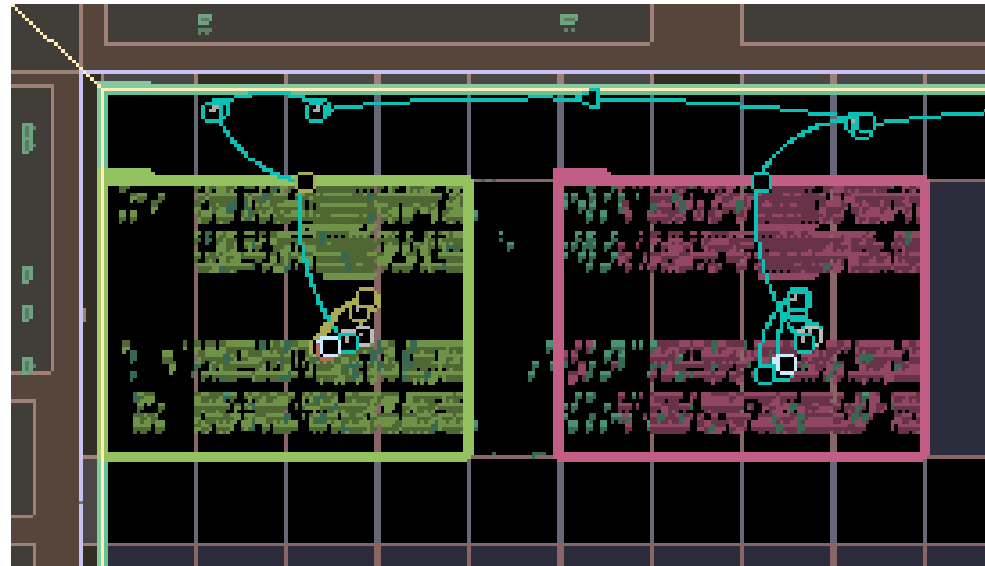


PreplaceIP gives the possibility to save a design as an IP and reuse it.

Only instances position are saved.

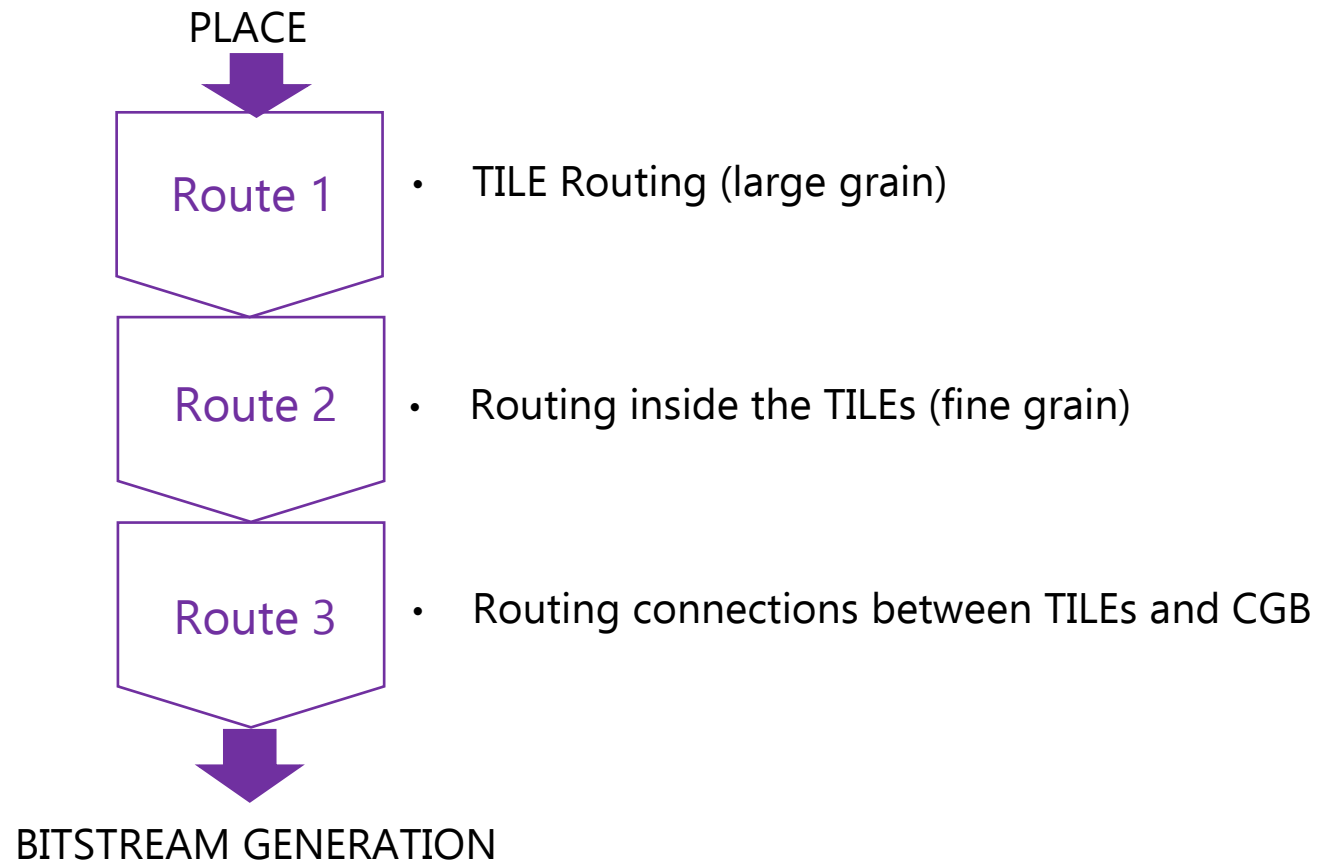
The spot (fix position) will be different.

The PreplaceIP regions are not exclusive



Route





- Logs
 - general.log : contains all messages printed during the Route step.
- You can generate at last route stage:
 - Netlist in VHDL or Verilog
 - SDF (Backannotated)

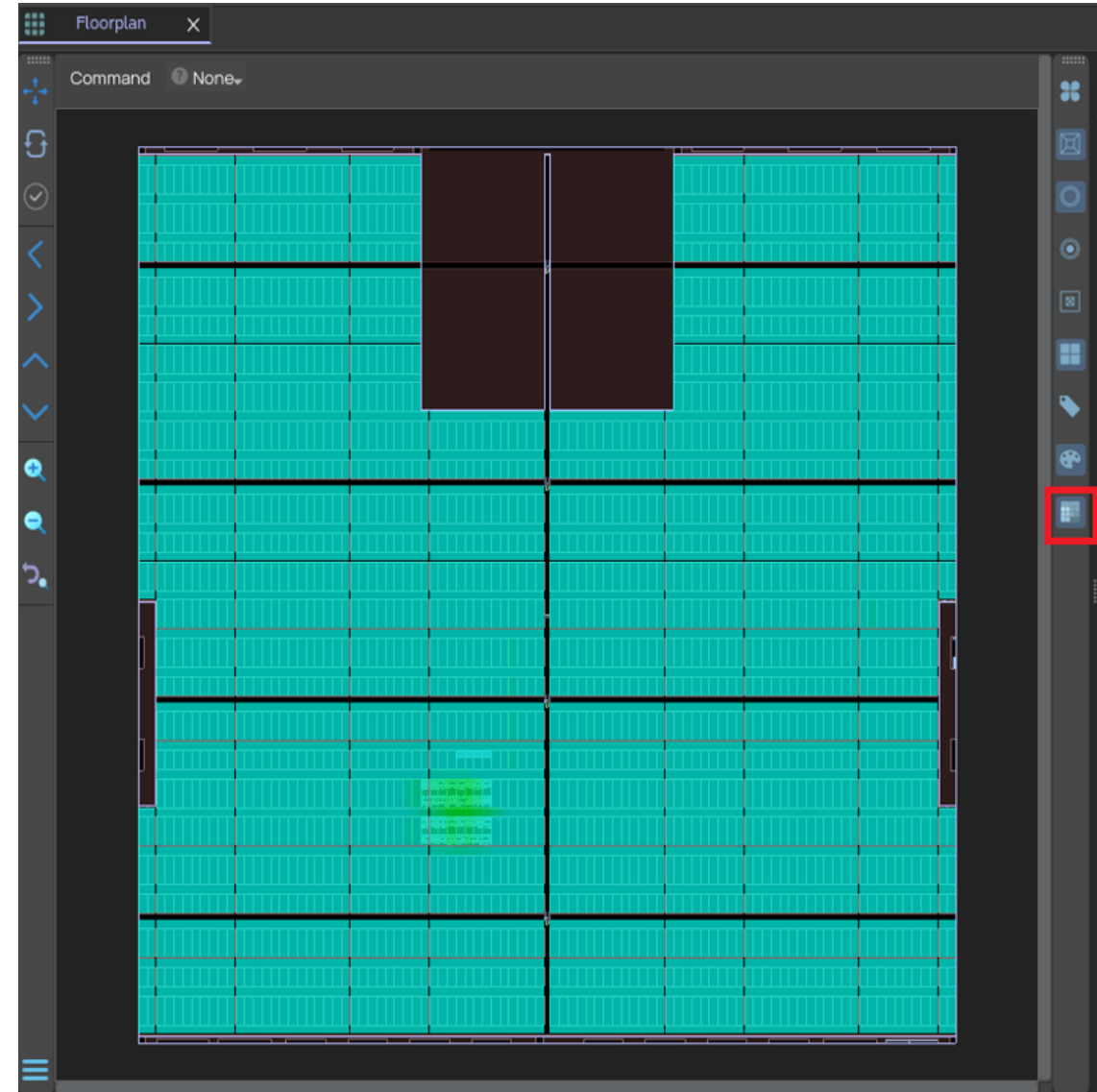
All route steps are saved in different .nym files which could be loaded by the GUI and NXpython command.

You can change several IO parameters as driver value.

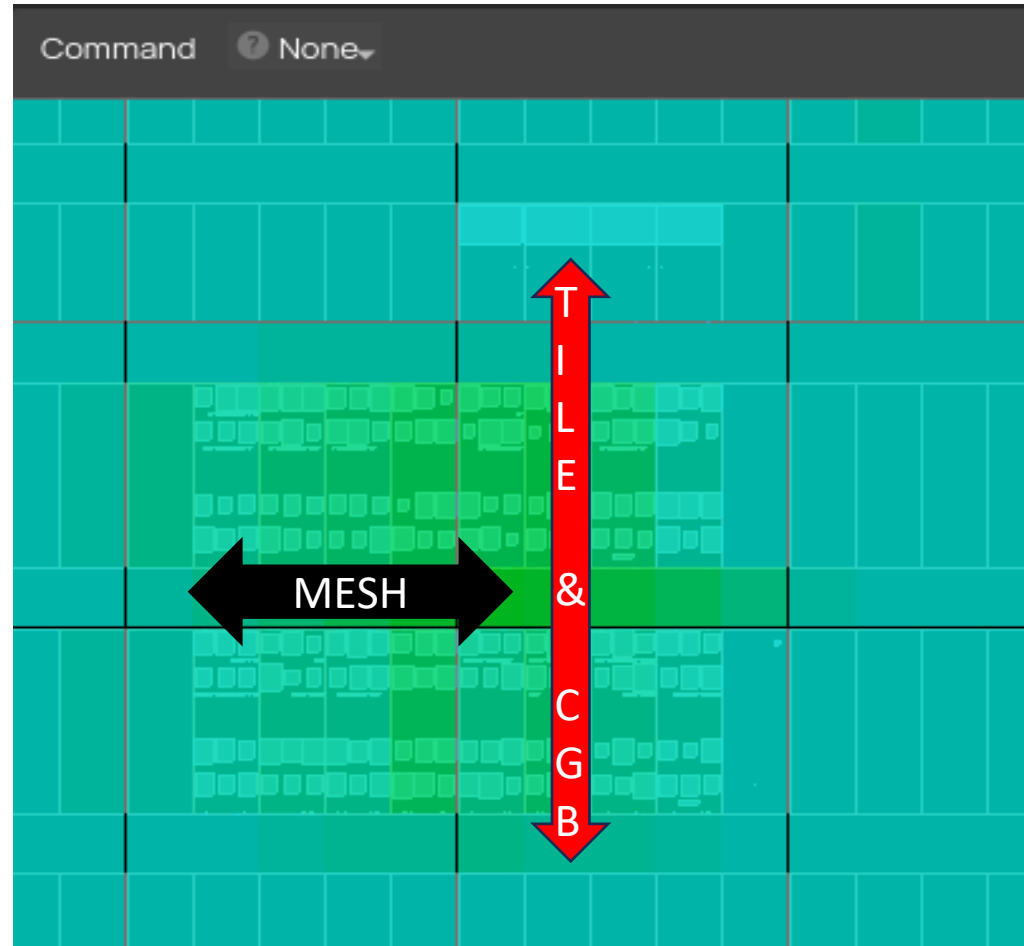
- Generate Bitstream can be done by GUI or NXpython method.
- Bitstream size depends on your design.

A congestion map is available with the button "Show Congestion Map"

- Vertical and horizontal congestion
- Available only when design is routed



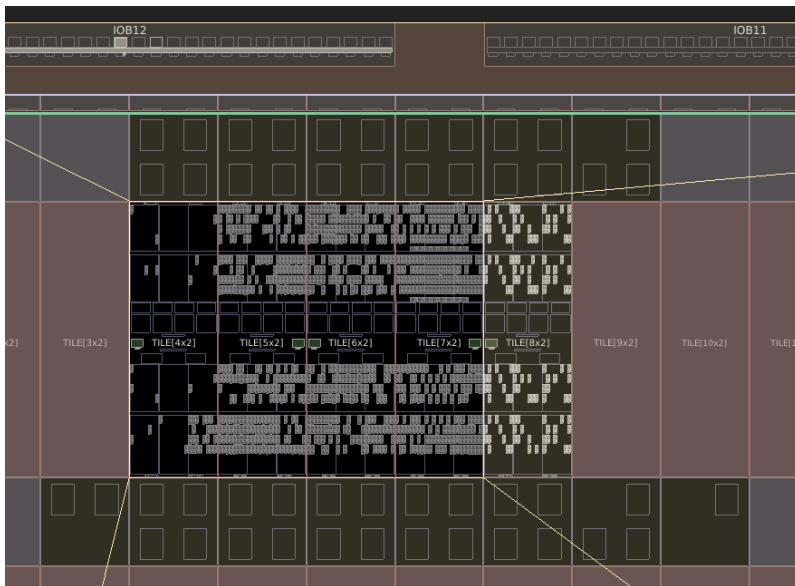
- MESH represents horizontal congestion.
- TILE and CGB represent vertical congestion.



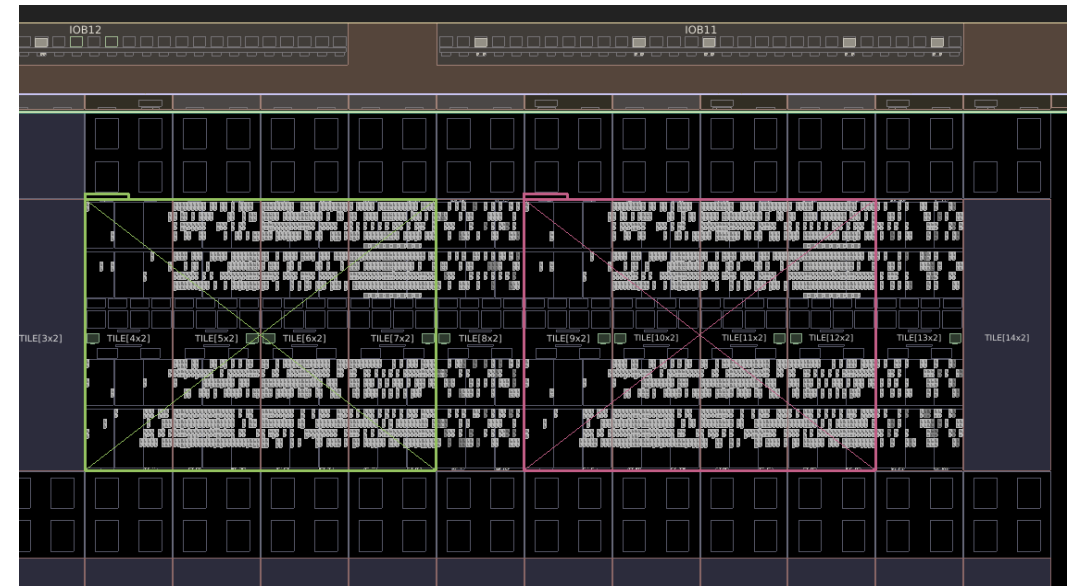
Integrates some optimized PreRouted IPs into a top design with the same performances.

For PreRoutedIP:

- The spot and route are saved
- The regions are exclusive

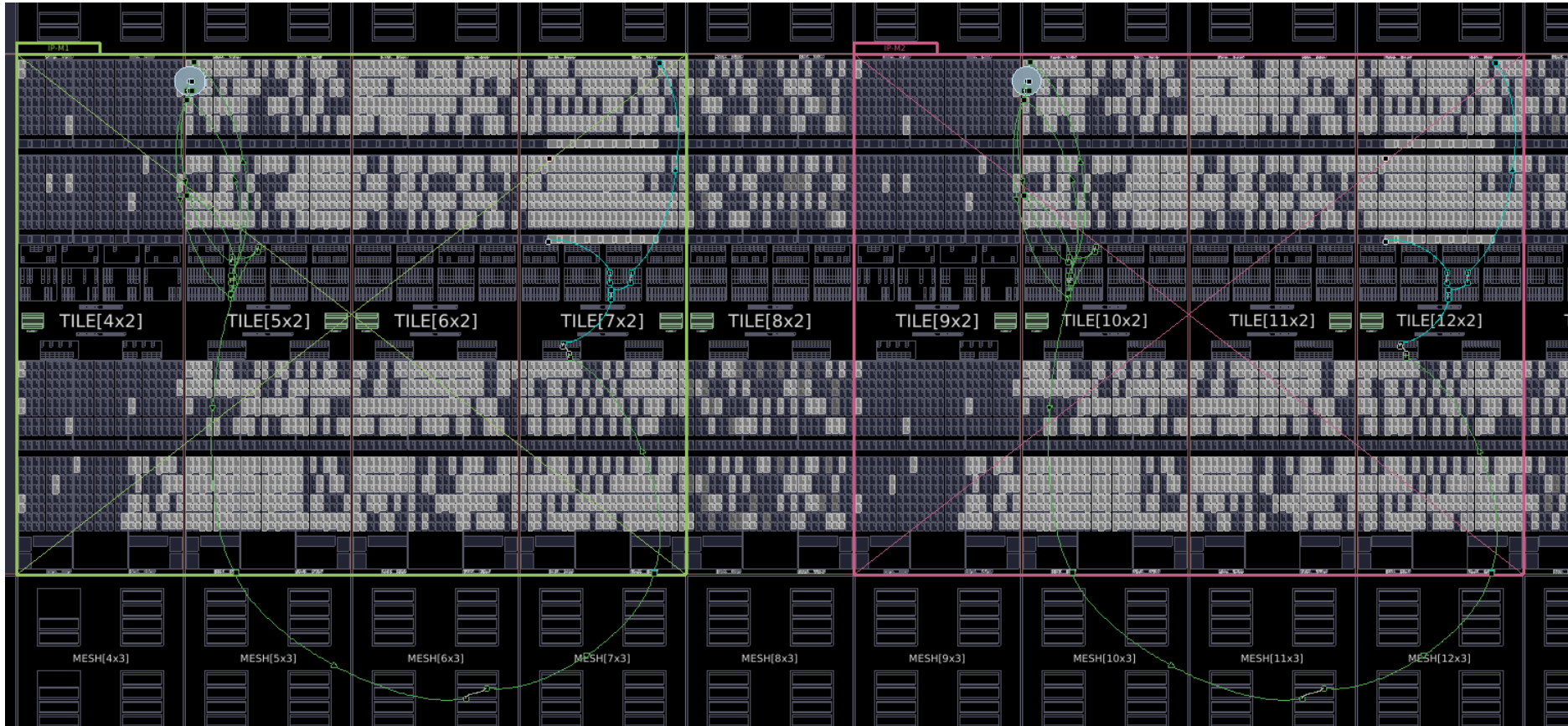


IP



TOP of another project

BETA



IP & Debug Tools



Clock Configuration

PLL  medium
 ultra
Ultra 300

NX Scope


Scope  medium
 ultra
Ultra 300

Memory


DFI  medium
 Asynchronous FIFO  medium

Interface









APB Slave Controller  medium

HSSL Wrapper for
 Spacefiber Interface 64 bits  ultra









QSPI Controller
 (Simulation only) **Ultra 300**

I2C  medium
 ultra
Ultra 300

Interface

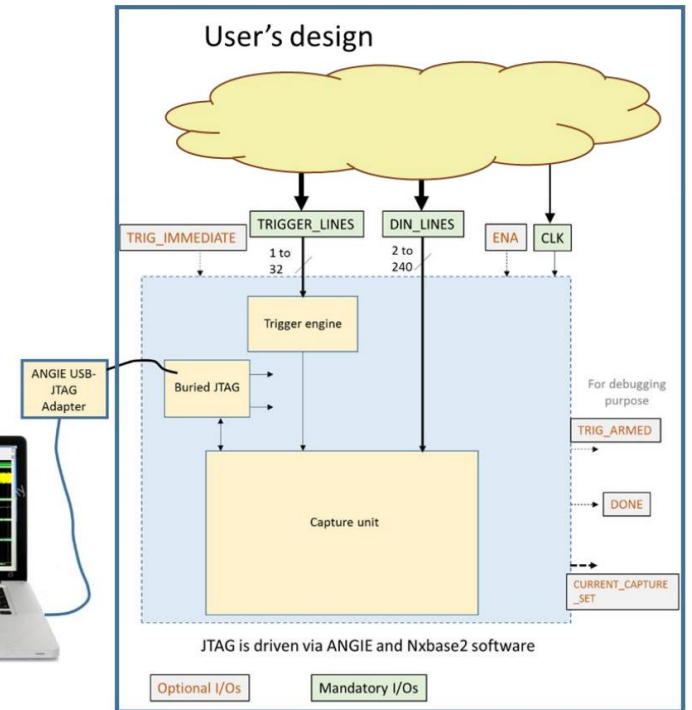
SpaceWireRMAP	 medium  ultra Ultra 300
SpaceWire Bank	 ultra Ultra 300
DDR3/DFI	 ultra
JESD204B Prototype	 ultra Ultra 300
SpaceWire IOM	 ultra Ultra 300
ESIstream Interface DAC and ADC	 ultra Ultra 300
Aurora	 ultra Ultra 300

Bridge

AXI4 to AHB	 medium  ultra Ultra 300
AHB to AXI4	 medium  ultra Ultra 300
AHB to IP1RW	 medium  ultra Ultra 300
AXI4 Full Interconnect	 medium  ultra Ultra 300

- NxScope is a NanoXplore IP Core.
 - It is an embedded logic analyzer which allows you to :
 - Sample a collection of internal data synchronously with a user's clock
 - Analyze the sampled results in a waveform viewer
- The results can be displayed and analyzed either with
 - ModelSim waveform viewer
 - GTKWave (free waveform display tool)

NxScope logic analyzer is built with the FPGA available logic resources.



Link => <https://nanoxplore-wiki.atlassian.net/wiki/spaces/NAN/pages/357467425/NxDesignSuite+23.5+NxCORE>

Documentation



Welcome to the NanoXplore Wiki





NanoXplore is a privately owned fabless company based in France. The company offers a comprehensive portfolio of SoCs and FPGA devices for aerospace, defense and industrial markets. Products include a leading radiation hardened FPGA portfolio.

Devices

NanoXplore develop devices focusing on three axes:

- A complete **Radiation Hardened FPGA** portfolio qualified for space applications and technologies including STM 65nm and 28nm FD-SOI
- Hard block embedded FPGA core IPs labelled as eFPGA products
- The development of ASIC devices

Our Radiation Hard FPGA range each include specific technical documentation, evaluation boards and DevKits to help discover and use all the available features.

NG-MEDIUM	NG-LARGE	NG-ULTRA	NG-ULTRA300
			
Documentation	Documentation	Documentation	Documentation
Evaluation boards & DevKits	Evaluation Boards & DevKits	Evaluation Boards & DevKits	Evaluation Boards & DevKits

Development Tools

To accompany the hardware portfolio, NanoXplore has conceived a complete dedicated software chain to map a hierarchical design logic application into the programmable core matrix of any NX FPGA device:

- NXmap was designed by NanoXplore as the main software tool to perform a full design flow compilation to programme FPGAs, including synthesis, place, route, static timing analysis and bitstream generation
- NXbase2 is a command line tool that can interact with evaluation kit boards for NanoXplore's chips. It provides a way to upload bitstream files into the chip and is able to control some of the hardware features of the related evaluation kits. The communication between computer and evaluation kit board is done by JTAG via the ANGIE USB-JTAG adapter provided with the evaluation kit board. NXboard is a graphical user interface, using NXbase2, that allows easy interaction with the evaluation kit boards
- SDK gathers embedded software tools and features for Development Kits to use efficiently complex parts of dedicated FPGAs like the R5 processor for NG-Large and the NG-Ultra System On Chip (SOC) functionalities
- NanoXplore provides its own License Manager Daemon (NXLMD) which must be installed to use NanoXplore tools

NXmap	NXbase2	SDK	Server license
Documentation & Download versions	Documentation & Download versions	Documentation & Download	Documentation & Download versions

IPs

NanoXplore develops its own portfolio of soft IPs designed to configure FPGA dedicated functionalities:

- DDR2 DFI IP
- SerDes IP
- SpaceWire bank IP

NanoXplore also provides IPs designed by partners.

Current list of devices are RH FPGA:

- NG-MEDIUM
- NG-LARGE
- NG-ULTRA
- ULTRA300

For each device, there is a main page with overview and links to download:

- Datasheet
- Pinout Allocation
- Ball Out
- IBIS models
- Power Estimator

Other documents are available:

- Configuration Guide
- Application Notes
- Cookbook
- Radiative Test Report

Link: <https://nanoxplore-wiki.atlassian.net/>

No need to subscribe !

Conclusion





Training Package

Shows several design examples.
Helps to learn how to use our software.

NxPython

STA

NxCore

NxBase2

NxScope

Link => <https://nanoxplore-wiki.atlassian.net/wiki/spaces/NAN/pages/357468388/NxDesignSuite+23.5+Training+Package>

Any questions about our products or licenses?

- Contact customer support by mail at support@nanoxplore.com
- Create an account on support platform at <https://support.nanoxplore.com>



Thank you