Management and analysis of physics datasets, Part. 1

Fourth Laboratory

Stefano Pavinato
5/12/2018

Istituto Nazionale di Fisica Nucleare
Laboratori Nazionali di Legnaro

Outline

Outline

Goals

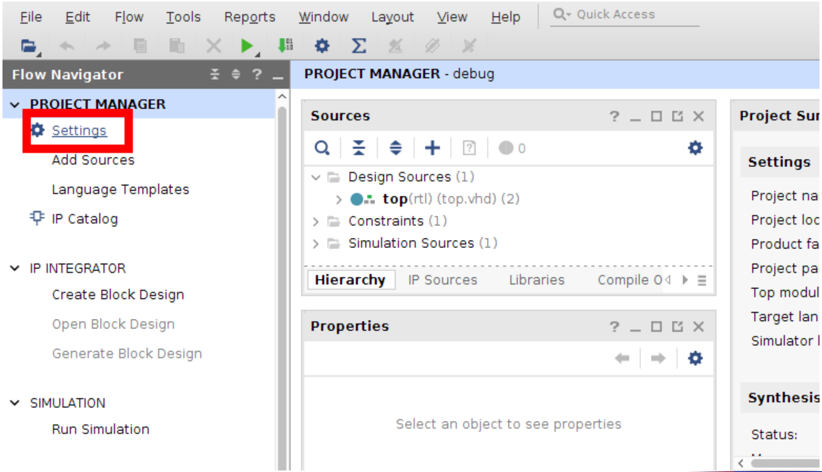
- Become familiar with the debug tools:
    1. ILA (Integrated Logic Analyzer);
    2. VIO (Virtual Input Output).

VHDL naming convention

Signals/components	Name
Clock	*clk*
Reset	*rst*
Input Port	*port_in*
Output Port	*port_out*
VHDL file name	*entityname.vhd*
Test bench file name	*tb_entityname.vhd*
Signal between 2 comps	*sign_cmp1_cmp2*
ila signal	*ila_signal*
vio signal	*vio_signal*
...	...

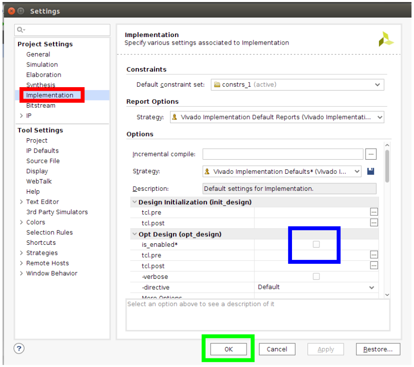
Implementation settings (1)

In order to have a faster implementation process, click on "Settings" →.



Implementation settings (2)

Click on "Implementation", uncheck "is_enabled" under the section "Opt Design" and "OK". **Please keep valid this configuration in all the next laboratories**.


Settings

Project Settings: General, Simulation, Elaboration, Synthesis, Implementation, Bitstream, IP

Tool Settings: Project, IP Defaults, Source File, Display, WebTalk, Help, Text Editor, 3rd Party Simulators, Colors, Selection Rules, Shortcuts, Strategies, Remote Hosts, Window Behavior

Implementation
Specify various settings associated to Implementation

Constraints
Default constraint set: constrs_1 (active)

Report Options
Strategy: Vivado Implementation Default Reports (Vivado Implementati...

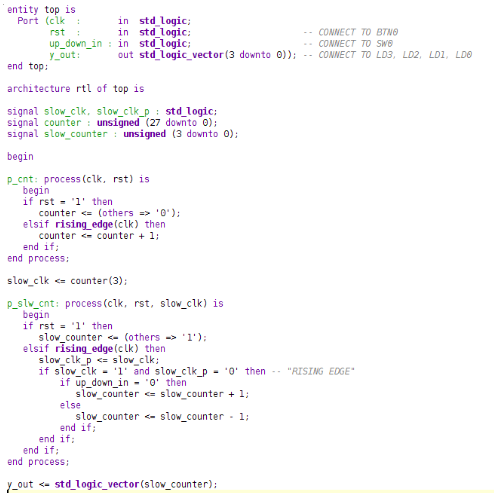
Options
Incremental compile:
Strategy: Vivado Implementation Defaults* (Vivado I...
Description: Default settings for Implementation.

Design Initialization (init_design)
- tcl.pre
- tcl.post

Opt Design (opt_design)
- is_enabled* ☐
- tcl.pre
- tcl.post
- -verbose ☐
- -directive: Default
- More Options

Select an option above to see a description of it

Code example



```vhdl
entity top is
  Port (clk  :        in  std_logic;
        rst  :        in  std_logic;                    -- CONNECT TO BTN0
        up_down_in :  in  std_logic;                    -- CONNECT TO SW0
        y_out:        out std_logic_vector(3 downto 0)); -- CONNECT TO LD3, LD2, LD1, LD0
end top;

architecture rtl of top is

signal slow_clk, slow_clk_p : std_logic;
signal counter : unsigned (27 downto 0);
signal slow_counter : unsigned (3 downto 0);

begin

p_cnt: process(clk, rst) is
   begin
   if rst = '1' then
      counter <= (others => '0');
   elsif rising_edge(clk) then
      counter <= counter + 1;
   end if;
end process;

slow_clk <= counter(3);

p_slw_cnt: process(clk, rst, slow_clk) is
   begin
   if rst = '1' then
      slow_counter <= (others => '1');
   elsif rising_edge(clk) then
      slow_clk_p <= slow_clk;
      if slow_clk = '1' and slow_clk_p = '0' then -- "RISING EDGE"
          if up_down_in = '0' then
             slow_counter <= slow_counter + 1;
          else
             slow_counter <= slow_counter - 1;
          end if;
      end if;
   end if;
end process;

y_out <= std_logic_vector(slow_counter);
```

Two questions

1. If you want reset the counter to a configurable value, how do you implement it?
2. Does this code work in the evaluation board?

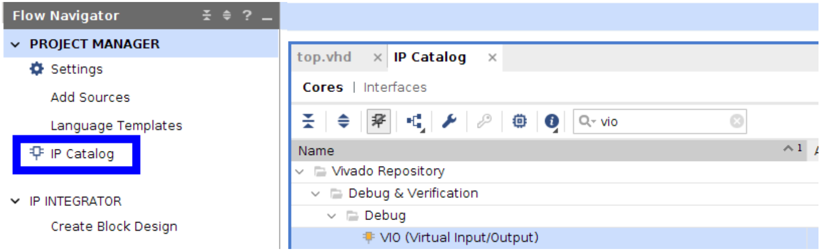
Outline

VIO

- It is a configurable core that can drive internal signals inside the FPGA.
- It can also monitor the signals. But in this course it is used only to drive the inputs. Essentially it substitutes the buttons and the switches used so far.
- With the VIO you can test the design already in the board.
- It allows hardware tests using jtag (basically the usb cable).
- Summarizing from the GUI, through the VIO, you control the project downloaded in the FPGA.

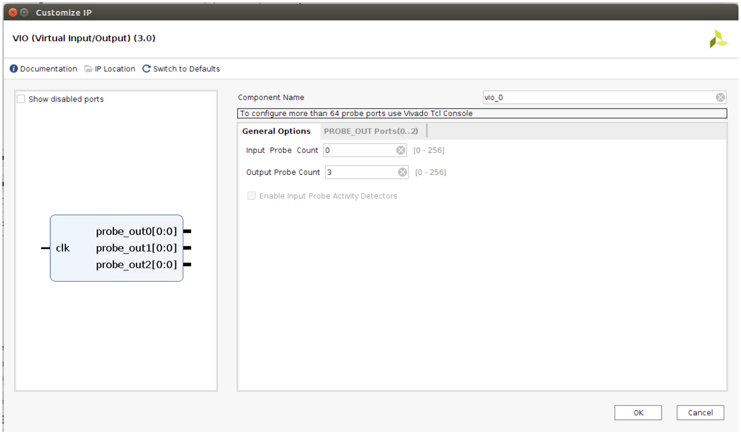
VIO core creation (1)

IP Catalog →, search "VIO" and double click.



VIO core creation (2)

In "Input Probe count" insert ALWAYS zero and in this example in "Ouput Probe count" insert 3.



Customize IP

VIO (Virtual Input/Output) (3.0)

Component Name: vio_0

To configure more than 64 probe ports use Vivado Tcl Console

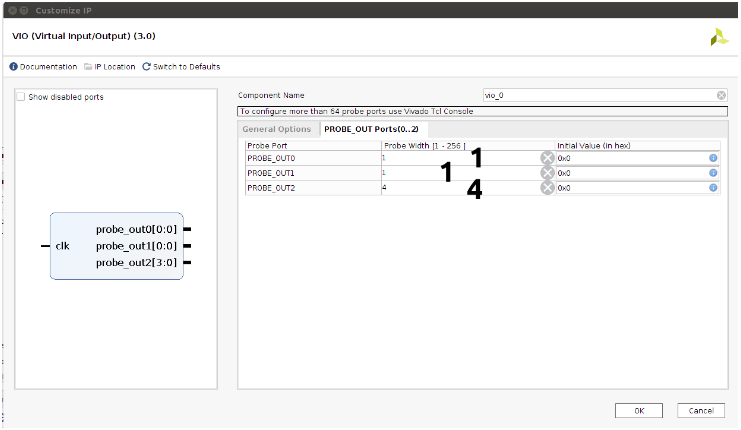
Input Probe Count: 0 [0 - 256]

Output Probe Count: 3 [0 - 256]

Enable Input Probe Activity Detectors

VIO core creation (3)

Open the window "PROBE_OUT Ports" (0 ... N) and insert the width for each port. Then "OK".



Customize IP

VIO (Virtual Input/Output) (3.0)

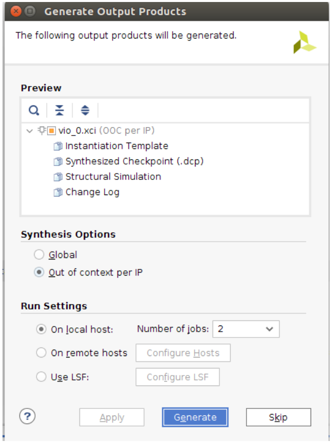
Component Name: vio_0

To configure more than 64 probe ports use Vivado Tcl Console

Probe Port	Probe Width [1 - 256]	Initial Value (in hex)
PROBE_OUT0	1	0x0
PROBE_OUT1	1	0x0
PROBE_OUT2	4	0x0

VIO core creation (4)

Click in "Generate".



Generate Output Products

The following output products will be generated.

Preview

- vio_0.xci (OOC per IP)
  - Instantiation Template
  - Synthesized Checkpoint (.dcp)
  - Structural Simulation
  - Change Log

Synthesis Options

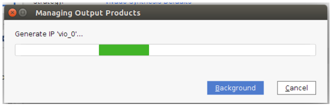
- ○ Global
- ● Out of context per IP

Run Settings

- ● On local host: Number of jobs: 2
- ○ On remote hosts
- ○ Use LSF:

VIO core creation (5)

Click in "Background".

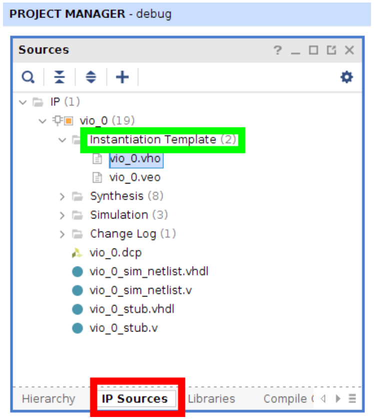

Managing Output Products

Generate IP 'vio_0'...

And the "OK" $\rightarrow$ in the window "Out-of-context module run was launched for generating output products".

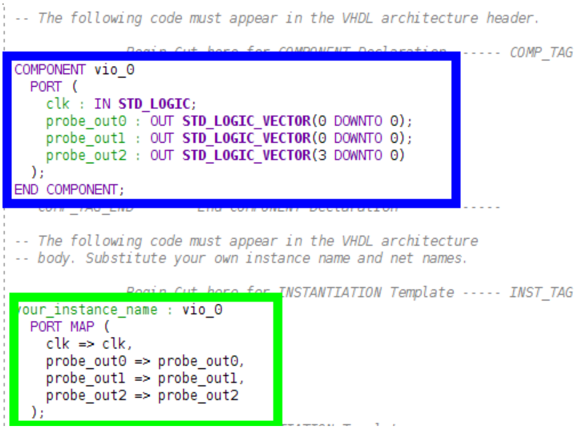
VIO declaration and instantiation (1)

Open the file "vio_0.vho" in the window "IP Sources", under the section "Instantiation Template".



VIO declaration and instantiation (2)

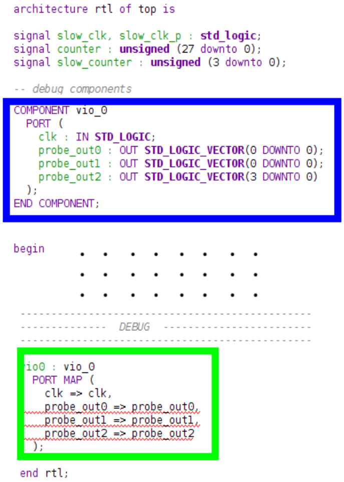
Copy the code inside the blue rectangle in the top module under the "architecture line" and the code inside the green rectangle in the top module above the end of the "architecture."



```
-- The following code must appear in the VHDL architecture header.

------------- Begin Cut here for COMPONENT Declaration ----- COMP_TAG
COMPONENT vio_0
  PORT (
    clk : IN STD_LOGIC;
    probe_out0 : OUT STD_LOGIC_VECTOR(0 DOWNTO 0);
    probe_out1 : OUT STD_LOGIC_VECTOR(0 DOWNTO 0);
    probe_out2 : OUT STD_LOGIC_VECTOR(3 DOWNTO 0)
  );
END COMPONENT;
-- COMP_TAG_END ------ End COMPONENT Declaration ------------

-- The following code must appear in the VHDL architecture
-- body. Substitute your own instance name and net names.

------------- Begin Cut here for INSTANTIATION Template ----- INST_TAG
your_instance_name : vio_0
  PORT MAP (
    clk => clk,
    probe_out0 => probe_out0,
    probe_out1 => probe_out1,
    probe_out2 => probe_out2
  );
-- INST_TAG_END ------ End INSTANTIATION Template ---------
```

VIO declaration and instantiation (3)

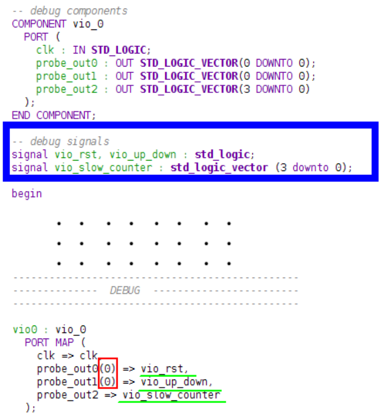
Rename the instantiation name. For example "vio0".



```
architecture rtl of top is

signal slow_clk, slow_clk_p : std_logic;
signal counter : unsigned (27 downto 0);
signal slow_counter : unsigned (3 downto 0);

-- debug components
COMPONENT vio_0
  PORT (
    clk : IN STD_LOGIC;
    probe_out0 : OUT STD_LOGIC_VECTOR(0 DOWNTO 0);
    probe_out1 : OUT STD_LOGIC_VECTOR(0 DOWNTO 0);
    probe_out2 : OUT STD_LOGIC_VECTOR(3 DOWNTO 0)
  );
END COMPONENT;

begin        . . . . . . . .
             . . . . . . . .
             . . . . . . . .
 ----------------------------------------------
 -------------  DEBUG  ----------------------
 ----------------------------------------------
 io0 : vio_0
  PORT MAP (
    clk => clk,
    probe_out0 => probe_out0,
    probe_out1 => probe_out1,
    probe_out2 => probe_out2
  );

 end rtl;
```

VIO internal signals (1)

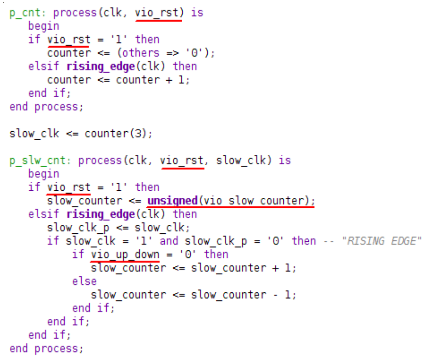
Inside the blue rectangle there is the declaration of the internal signals to be connected to the VIO probes.



```
-- debug components
COMPONENT vio_0
  PORT (
    clk : IN STD_LOGIC;
    probe_out0 : OUT STD_LOGIC_VECTOR(0 DOWNTO 0);
    probe_out1 : OUT STD_LOGIC_VECTOR(0 DOWNTO 0);
    probe_out2 : OUT STD_LOGIC_VECTOR(3 DOWNTO 0)
  );
END COMPONENT;

-- debug signals
signal vio_rst, vio_up_down : std_logic;
signal vio_slow_counter : std_logic_vector (3 downto 0);

begin

      .   .   .   .   .   .   .   .
      .   .   .   .   .   .   .   .
      .   .   .   .   .   .   .   .

-----------------------------------------------
--------------   DEBUG   ----------------------
-----------------------------------------------

vio0 : vio_0
  PORT MAP (
    clk => clk,
    probe_out0(0) => vio_rst,
    probe_out1(0) => vio_up_down,
    probe_out2 => vio_slow_counter
  );
```

VIO internal signals (2)

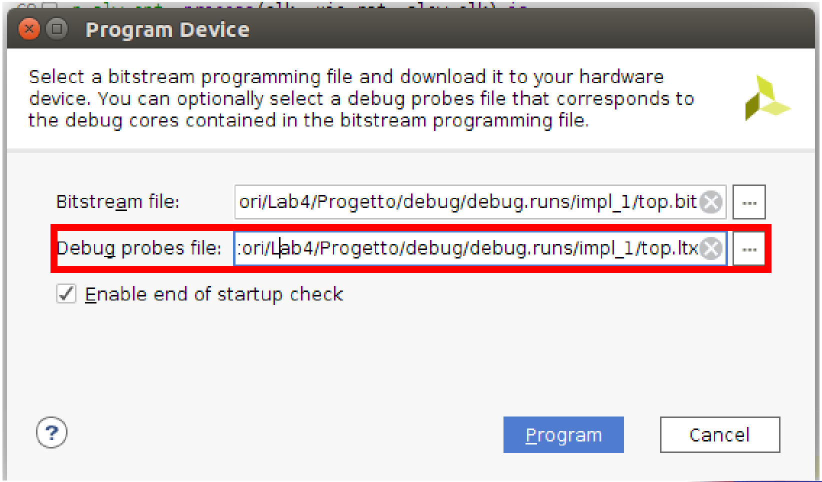
Now connect the VIO probes to the VHDL code. That is substitute the input signals with the VIO signals.



```
p_cnt: process(clk, vio_rst) is
   begin
   if vio_rst = '1' then
      counter <= (others => '0');
   elsif rising_edge(clk) then
      counter <= counter + 1;
   end if;
end process;

slow_clk <= counter(3);

p_slw_cnt: process(clk, vio_rst, slow_clk) is
   begin
   if vio_rst = '1' then
      slow_counter <= unsigned(vio_slow_counter);
   elsif rising_edge(clk) then
      slow_clk_p <= slow_clk;
      if slow_clk = '1' and slow_clk_p = '0' then -- "RISING EDGE"
          if vio_up_down = '0' then
             slow_counter <= slow_counter + 1;
          else
             slow_counter <= slow_counter - 1;
          end if;
      end if;
   end if;
end process;
```

VIO programing (1)

Program the FPGA.



Program Device

Select a bitstream programming file and download it to your hardware device. You can optionally select a debug probes file that corresponds to the debug cores contained in the bitstream programming file.

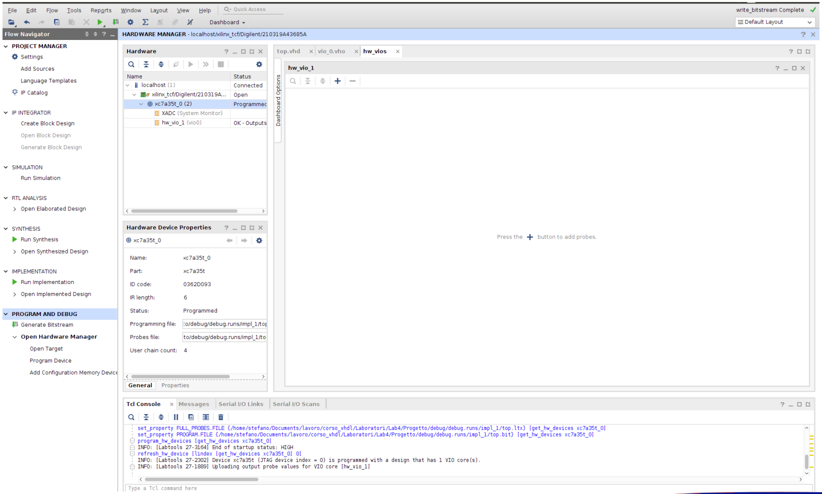
Bitstream file: ori/Lab4/Progetto/debug/debug.runs/impl_1/top.bit

Debug probes file: :ori/Lab4/Progetto/debug/debug.runs/impl_1/top.ltx

✓ Enable end of startup check

VIO programing (2)

Click on "+".



HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210319A43685A
hw_vio_1
Press the + button to add probes.

Hardware

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/210319A...	Open
xc7a35t_0 (2)	Programmed
XADC (System Monitor)	
hw_vio_1 (vio0)	OK - Outputs

Hardware Device Properties

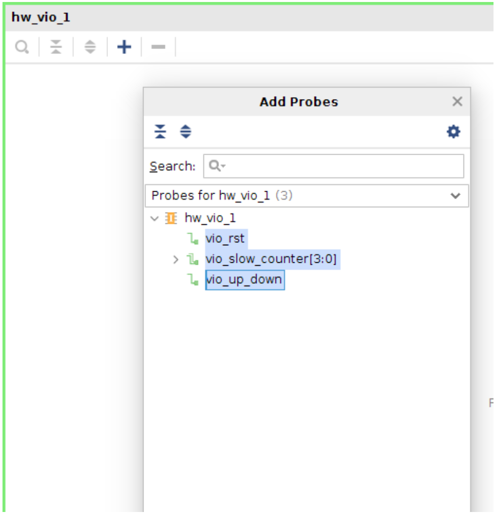
Name:	xc7a35t_0
Part:	xc7a35t
ID code:	0362D093
IR length:	6
Status:	Programmed
Programming file:	:o/debug/debug.runs/impl_1/top
Probes file:	to/debug/debug.runs/impl_1/to
User chain count:	4

Tcl Console

```
set_property FULL_PROBES.FILE {/home/stefano/Documents/lavoro/corso_vhdl/Laboratori/Lab4/Progetto/debug/debug.runs/impl_1/top.ltx} [get_hw_devices xc7a35t_0]
set_property PROGRAM.FILE {/home/stefano/Documents/lavoro/corso_vhdl/Laboratori/Lab4/Progetto/debug/debug.runs/impl_1/top.bit} [get_hw_devices xc7a35t_0]
program_hw_devices [get_hw_devices xc7a35t_0]
INFO: [Labtools 27-3164] End of startup status: HIGH
refresh_hw_device [lindex [get_hw_devices xc7a35t_0] 0]
INFO: [Labtools 27-2302] Device xc7a35t (JTAG device index = 0) is programmed with a design that has 1 VIO core(s).
INFO: [Labtools 27-1889] Uploading output probe values for VIO core [hw_vio_1]
```

VIO programing (3)

Select the VIO probes.


hw_vio_1

Add Probes

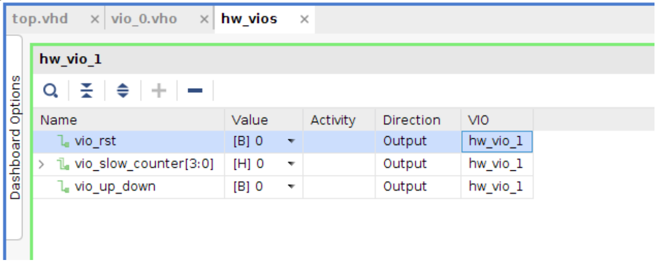
Search:

Probes for hw_vio_1 (3)

- hw_vio_1
  - vio_rst
  - vio_slow_counter[3:0]
  - vio_up_down

VIO programing (4)

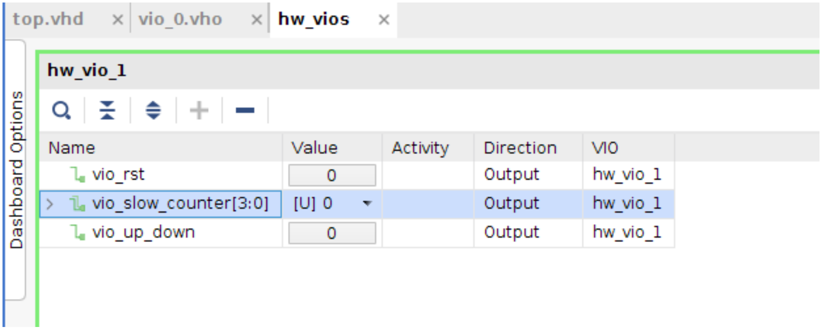
Right click on each of the three signals. Change the option of the *vio_rst* and *vio_up_down* in "Toggle Button" and the option of the *vio_slow_counter* in "Radix" → "Unsigned Decimal".



Name	Value	Activity	Direction	VIO
vio_rst	[B] 0		Output	hw_vio_1
vio_slow_counter[3:0]	[H] 0		Output	hw_vio_1
vio_up_down	[B] 0		Output	hw_vio_1

VIO programing (5)

Try to reset the board and change in real-time the reset value of the counter.



top.vhd × vio_0.vho × hw_vios ×

Dashboard Options

hw_vio_1

Name	Value	Activity	Direction	VIO
vio_rst	0		Output	hw_vio_1
vio_slow_counter[3:0]	[U] 0		Output	hw_vio_1
vio_up_down	0		Output	hw_vio_1

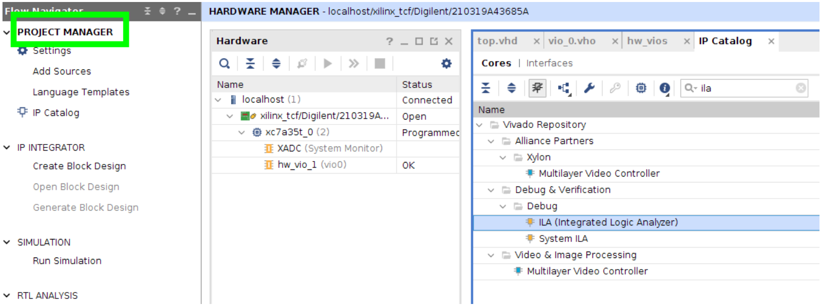
Outline

ILA

- It is a configurable core that can monitor the internal signals inside the FPGA. Essentially it substitutes the leds used so far.
- It can be considered an oscilloscope inside the FPGA.
- It allows hardware tests using jtag (basically the usb cable).
- Summarizing from the GUI, through the ILA, you can monitor and check the evolution of the output and internal signals.

ILA core creation (1)

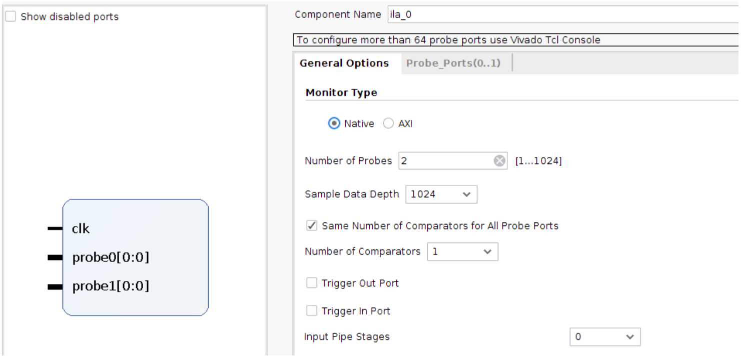
Probably you have to click on "PROJECT MANAGER". Then IP Catalog →, search "ILA" and double click.



Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/210319A...	Open
xc7a35t_0 (2)	Programmed
XADC (System Monitor)	
hw_vio_1 (vio0)	OK

ILA core creation (2)

In "Number of Probes" insert, for this example, 2.



Show disabled ports

Component Name ila_0

To configure more than 64 probe ports use Vivado Tcl Console

Monitor Type

[✓ Native] AXI

Number of Probes 2 [1...1024]

Sample Data Depth 1024

[✓] Same Number of Comparators for All Probe Ports

Number of Comparators 1

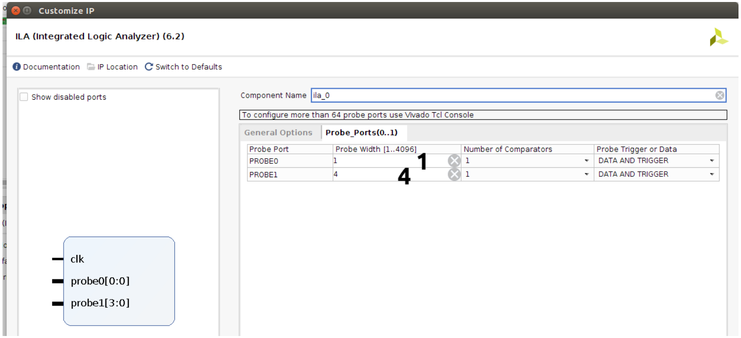
[ ] Trigger Out Port

[ ] Trigger In Port

Input Pipe Stages 0

ILA core creation (3)

Open the window "PROBE_Ports" (0 ... N) and insert in the fields "Probe Width" 1 for PROBE0 and 4 for PROBE1. Then "OK".



Customize IP

ILA (Integrated Logic Analyzer) (6.2)

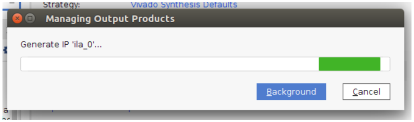
Component Name: ila_0

To configure more than 64 probe ports use Vivado Tcl Console

Probe Port	Probe Width [1..4096]	Number of Comparators	Probe Trigger or Data
PROBE0	1	1	DATA AND TRIGGER
PROBE1	4	1	DATA AND TRIGGER

VIO core creation (4)

Click in "Generate". Then click in "Background".



Strategy: Vivado Synthesis Defaults

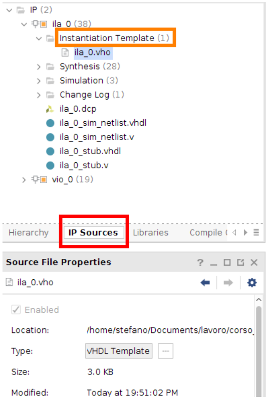
Managing Output Products

Generate IP 'ila_0'...

And the "OK" → in the window "Out-of-context module run was launched for generating output products".

ILA declaration and instantiation (1)

Open the file "ila_0.vho" in the window "IP Sources", under the section "Instantiation Template".



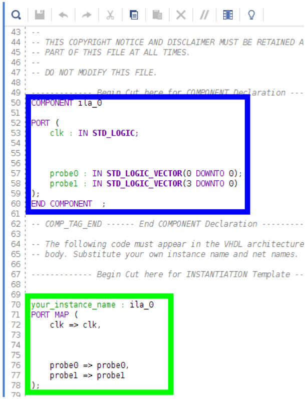
```
IP (2)
  ila_0 (38)
    Instantiation Template (1)
      ila_0.vho
    Synthesis (28)
    Simulation (3)
    Change Log (1)
    ila_0.dcp
    ila_0_sim_netlist.vhdl
    ila_0_sim_netlist.v
    ila_0_stub.vhdl
    ila_0_stub.v
  vio_0 (19)
```

Hierarchy | **IP Sources** | Libraries | Compile

Source File Properties

ila_0.vho

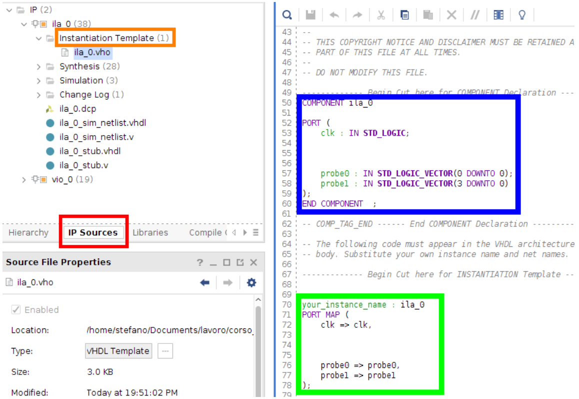
✓ Enabled	
Location:	/home/stefano/Documents/lavoro/corso
Type:	VHDL Template
Size:	3.0 KB
Modified:	Today at 19:51:02 PM



```
--
-- THIS COPYRIGHT NOTICE AND DISCLAIMER MUST BE RETAINED A
-- PART OF THIS FILE AT ALL TIMES.
--
-- DO NOT MODIFY THIS FILE.

------------- Begin Cut here for COMPONENT Declaration ---
COMPONENT ila_0

PORT (
    clk : IN STD_LOGIC;


    probe0 : IN STD_LOGIC_VECTOR(0 DOWNTO 0);
    probe1 : IN STD_LOGIC_VECTOR(3 DOWNTO 0)
);
END COMPONENT  ;

-- COMP_TAG_END ------ End COMPONENT Declaration --------

-- The following code must appear in the VHDL architecture
-- body. Substitute your own instance name and net names.

------------- Begin Cut here for INSTANTIATION Template --

your_instance_name : ila_0
PORT MAP (
    clk => clk,


    probe0 => probe0,
    probe1 => probe1
);
```

ILA declaration and instantiation (2)

Copy the code inside the blue rectangle in the top module under the "architecture line" and the code inside the green rectangle in the top module above the instantiation of the VIO.

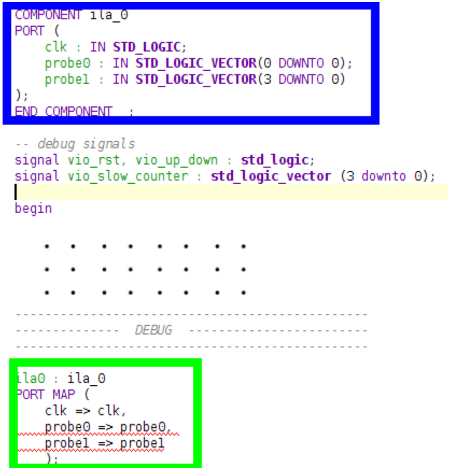


Enabled
Location: /home/stefano/Documents/lavoro/corso
Type: VHDL Template
Size: 3.0 KB
Modified: Today at 19:51:02 PM

```
--
-- THIS COPYRIGHT NOTICE AND DISCLAIMER MUST BE RETAINED A
-- PART OF THIS FILE AT ALL TIMES.
--
-- DO NOT MODIFY THIS FILE.

------------- Begin Cut here for COMPONENT Declaration ---
COMPONENT ila_0

PORT (
    clk : IN STD_LOGIC;



    probe0 : IN STD_LOGIC_VECTOR(0 DOWNTO 0);
    probe1 : IN STD_LOGIC_VECTOR(3 DOWNTO 0)
);
END COMPONENT  ;

-- COMP_TAG_END ------ End COMPONENT Declaration --------

-- The following code must appear in the VHDL architecture
-- body. Substitute your own instance name and net names.

------------- Begin Cut here for INSTANTIATION Template --


your_instance_name : ila_0
PORT MAP (
    clk => clk,



    probe0 => probe0,
    probe1 => probe1
);
```

ILA declaration and instantiation (3)

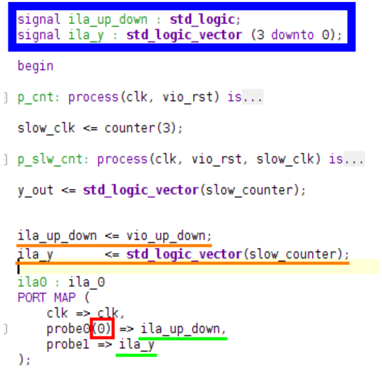
Rename the instantiation name. For example "ila0".



```
COMPONENT ila_0
PORT (
    clk : IN STD_LOGIC;
    probe0 : IN STD_LOGIC_VECTOR(0 DOWNTO 0);
    probe1 : IN STD_LOGIC_VECTOR(3 DOWNTO 0)
);
END COMPONENT  ;

-- debug signals
signal vio_rst, vio_up_down : std_logic;
signal vio_slow_counter : std_logic_vector (3 downto 0);

begin

    .  .  .  .  .  .  .  .
    .  .  .  .  .  .  .  .
    .  .  .  .  .  .  .  .

----------------------------------------------
-------------  DEBUG  -----------------------
----------------------------------------------

ila0 : ila_0
PORT MAP (
    clk => clk,
    probe0 => probe0,
    probe1 => probe1
    );
```

ILA internal signals (1)

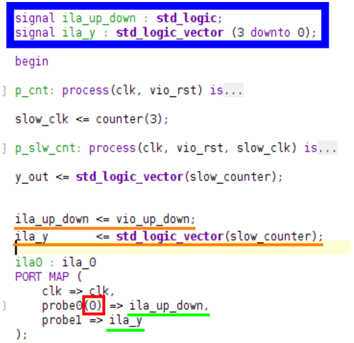
Inside the blue rectangle there is the declaration of the internal signals to be connected to the ILA probes.



```
signal ila_up_down : std_logic;
signal ila_y : std_logic_vector (3 downto 0);

begin

p_cnt: process(clk, vio_rst) is...

slow_clk <= counter(3);

p_slw_cnt: process(clk, vio_rst, slow_clk) is...

y_out <= std_logic_vector(slow_counter);


ila_up_down <= vio_up_down;
ila_y       <= std_logic_vector(slow_counter);

ila0 : ila_0
PORT MAP (
    clk => clk,
    probe0(0) => ila_up_down,
    probe1 => ila_y
);
```

ILA internal signals (2)

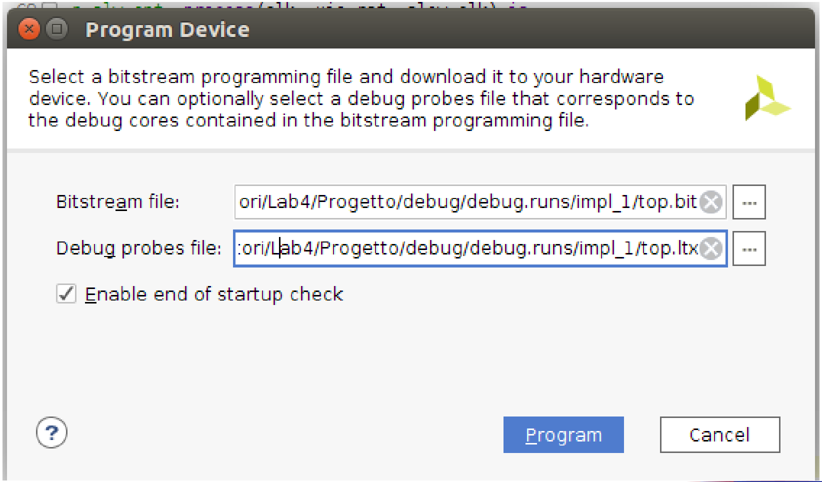
Now connect the ILA probes to the VHDL code.



```
signal ila_up_down : std_logic;
signal ila_y : std_logic_vector (3 downto 0);

begin

p_cnt: process(clk, vio_rst) is...

slow_clk <= counter(3);

p_slw_cnt: process(clk, vio_rst, slow_clk) is...

y_out <= std_logic_vector(slow_counter);


ila_up_down <= vio_up_down;
ila_y       <= std_logic_vector(slow_counter);

ila0 : ila_0
PORT MAP (
    clk => clk,
    probe0(0) => ila_up_down,
    probe1 => ila_y
);
```

ILA programing (1)

Program the FPGA.



Program Device

Select a bitstream programming file and download it to your hardware device. You can optionally select a debug probes file that corresponds to the debug cores contained in the bitstream programming file.

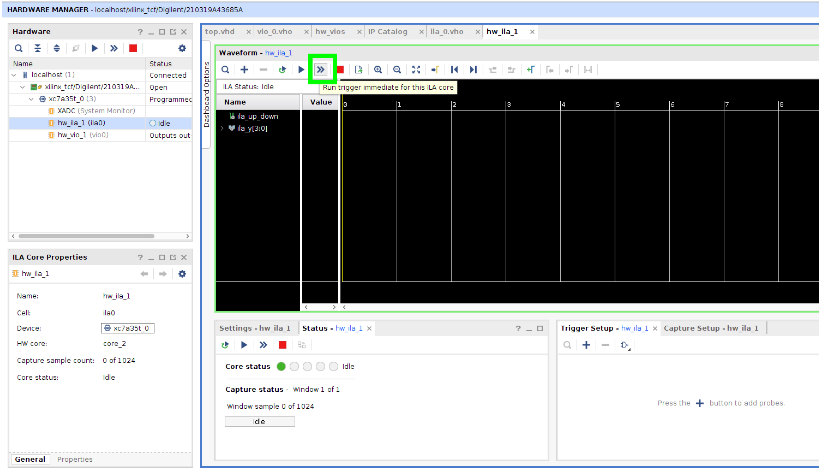
Bitstream file: ori/Lab4/Progetto/debug/debug.runs/impl_1/top.bit

Debug probes file: :ori/Lab4/Progetto/debug/debug.runs/impl_1/top.ltx

✓ Enable end of startup check

ILA programing (2)

Click on "Run trigger immediate for this ILA core". Symbol >>.



HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210319A43685A

Hardware

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/210319A...	Open
xc7a35t_0 (3)	Programmed
XADC (System Monitor)	
hw_ila_1 (ila0)	Idle
hw_vio_1 (vio0)	Outputs out

ILA Core Properties

hw_ila_1

Name:	hw_ila_1
Cell:	ila0
Device:	xc7a35t_0
HW core:	core_2
Capture sample count:	0 of 1024
Core status:	Idle

General Properties

top.vhd | vio_0.vho | hw_vios | IP Catalog | ila_0.vho | hw_ila_1

Waveform - hw_ila_1

ILA Status: Idle

Run trigger immediate for this ILA core

Name	Value
ila_up_down	
ila_y[3:0]	

Settings - hw_ila_1 | Status - hw_ila_1

Core status ● ○ ○ ○ ○ Idle

Capture status - Window 1 of 1

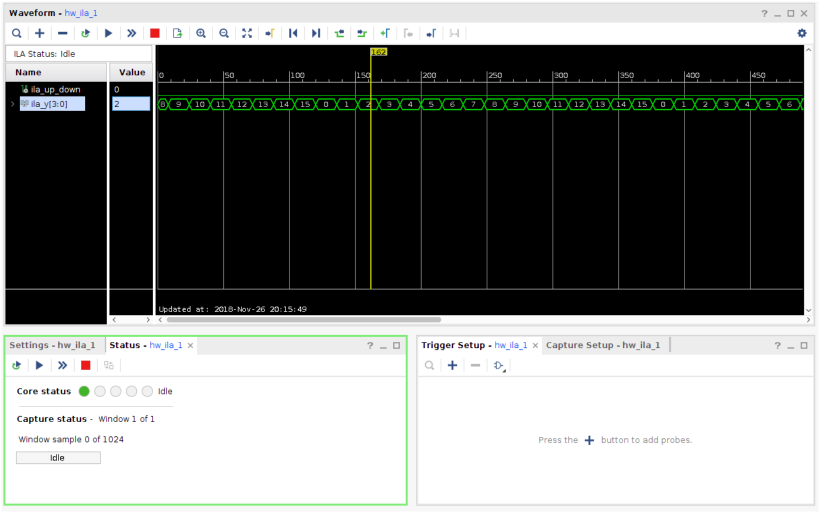
Window sample 0 of 1024

Idle

Trigger Setup - hw_ila_1 | Capture Setup - hw_ila_1

Press the + button to add probes.

ILA programing (3)



```
Updated at: 2018-Nov-26 20:15:49
```

Core status ● ○ ○ ○ ○ Idle

Capture status - Window 1 of 1

Window sample 0 of 1024

Press the + button to add probes.

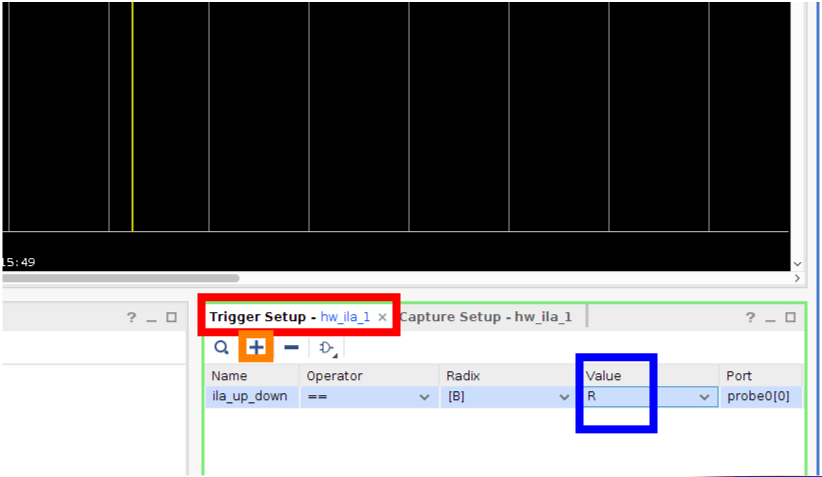
ILA trigger (1)

- ILA core has an important feature: the trigger option.
- The most used option is the positive edge transition trigger.
- Basically with this option you say to the ILA core to start the acquisition when the trigger event happens.

For example, in this laboratory, we say to the ILA "Start the acquisition when the up_down signal changes from '0' to '1'".

ILA trigger (2)

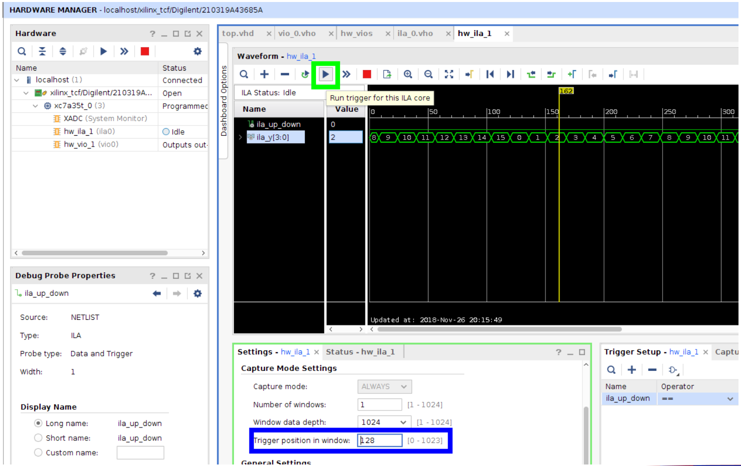
In the window "Trigger Setup", press "+", select ila_up_down signal and change the "Value" in "R".



Name	Operator	Radix	Value	Port
ila_up_down	==	[B]	R	probe0[0]

ILA trigger (3)

Set the "Trigger position in window", for example, at 128 and now press "Run trigger for this ILA core".



HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210319A43685A

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/210319A...	Open
xc7a35t_0 (3)	Programmed
XADC (System Monitor)	
hw_ila_1 (ila0)	Idle
hw_vio_1 (vio0)	Outputs out

Debug Probe Properties

ila_up_down

Source:	NETLIST
Type:	ILA
Probe type:	Data and Trigger
Width:	1

Display Name

[✓ Long name:]	ila_up_down
Short name:	ila_up_down
Custom name:	

Waveform - hw_ila_1

ILA Status: Idle

Run trigger for this ILA core

Name	Value
ila_up_down	0
ila_y[3:0]	2

8 9 10 11 12 13 14 15 0 1 2 3 4 5 6 7 8 9 10 11

Updated at: 2018-Nov-26 20:15:49

Settings - hw_ila_1 | Status - hw_ila_1

Capture Mode Settings

Capture mode:	ALWAYS	
Number of windows:	1	[1 - 1024]
Window data depth:	1024	[1 - 1024]
Trigger position in window:	128	[0 - 1023]

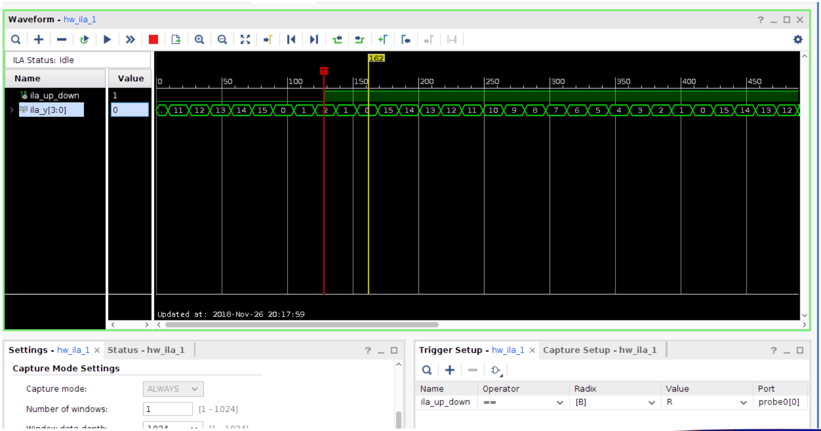
General Settings

Trigger Setup - hw_ila_1

Name	Operator
ila_up_down	==

ILA trigger (4)

In the window "hw_vios" toggle the button "vio_up_down" in order to have a transition 0 → 1 and then check what happened in the window "hw_ila_1".



Waveform - hw_ila_1

ILA Status: Idle

Name	Value
ila_up_down	1
ila_y[3:0]	0

```
Updated at: 2018-Nov-26 20:17:59
```

Settings - hw_ila_1 | Status - hw_ila_1

Capture Mode Settings

Capture mode: ALWAYS

Number of windows: 1 [1 - 1024]

Window data depth: 1024 [1 - 1024]

Trigger Setup - hw_ila_1 | Capture Setup - hw_ila_1

Name	Operator	Radix	Value	Port
ila_up_down	==	[B]	R	probe0[0]

Outline

Suggested exercises

- Redo this exercise and the exercises of the past laboratories (in particular the 4 bit adder) driving the inputs with the VIO core and monitoring the outputs with the ILA core.