Vivado从此开始（To Learn Vivado From Here）

本书围绕Vivado四大主题

- 设计流程
- 时序约束
- 时序分析
- Tcl脚本的使用

作者：高亚军（Xilinx战略应用高级工程师）

- 2012年2月，出版《基于FPGA的数字信号处理（第1版）》
- 2012年9月，发布网络视频课程《Vivado入门与提高》
- 2015年7月，出版《基于FPGA的数字信号处理（第2版）》
- 2016年7月，发布网络视频课程《跟Xilinx SAE学HLS》

- 内容翔实全面：涵盖Vivado所有基本功能
- 讲解深入浅出：结合大量案例，帮助读者加强对基本概念的理解
- 描述图文并茂：给出具体操作步骤，易于快速动手实践

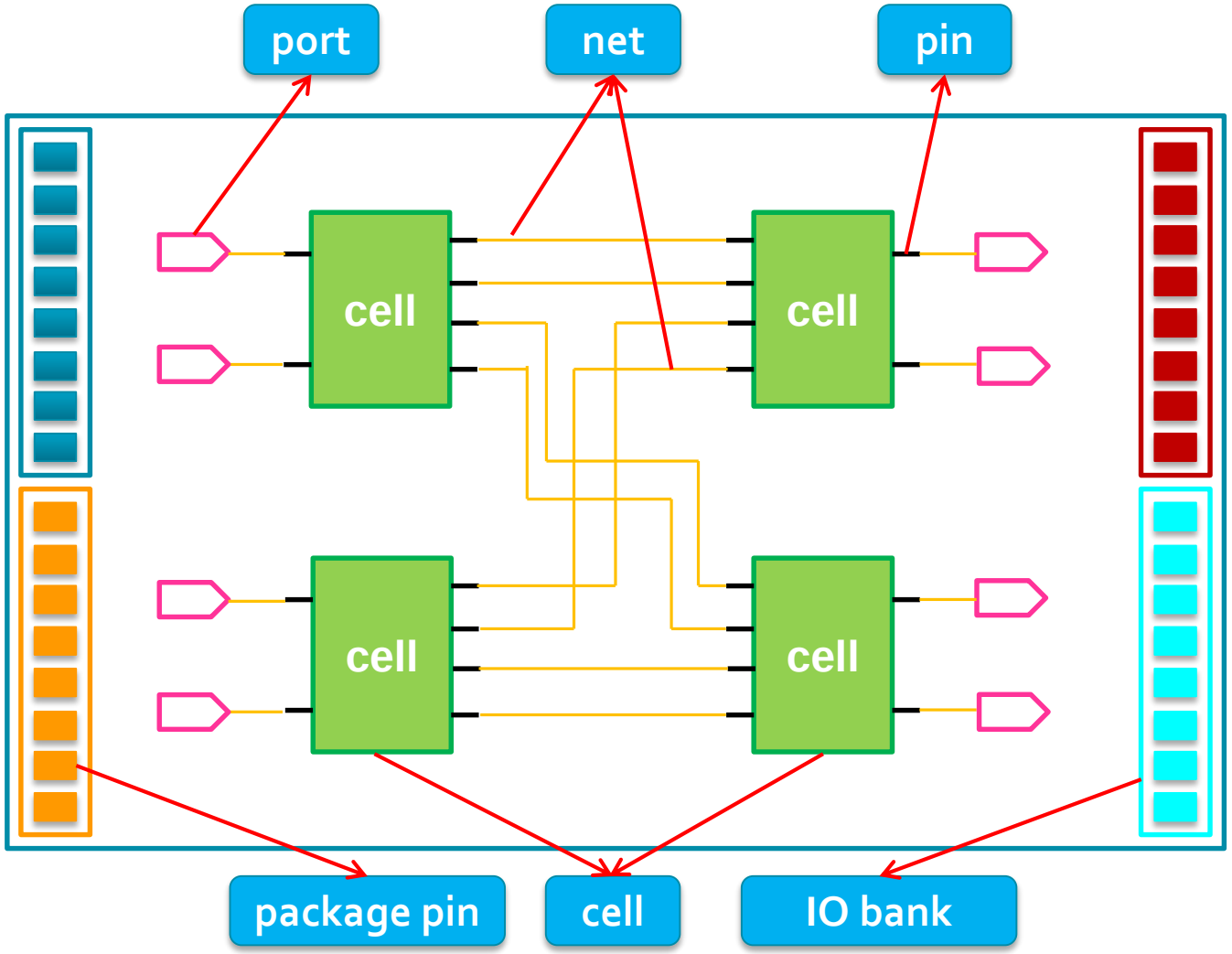
Five Most Commonly Used Tcl Commands in Vivado

Lauren Gao

Agenda

- Basic concept of objects in Vivado
- How to use five most important Tcl commands in design
  - get_cells, get_nets, get_ports, get_pins, get_clocks

Objects in Vivado: cell, port, net and pin



Basic Commands get_*

Command	-hierarchical	-regexp	-nocase	-filter	-of_objects
get_cells	√	√	√	√	√
get_nets	√	√	√	√	√
get_pins	√	√	√	√	√
get_ports	X	√	√	√	√
get_clocks	X	√	√	√	√

- *-hierarchical ←→ -hier*
- *-of_objects ←→ -of*
- *-filter: using properties to filter*

1.)

String Comparison	
equal	==
not equal	!=
match	=~
not match	!~

2.) Multiple filter expressions 3.) Boolean type properties

AND (&&), OR (||)

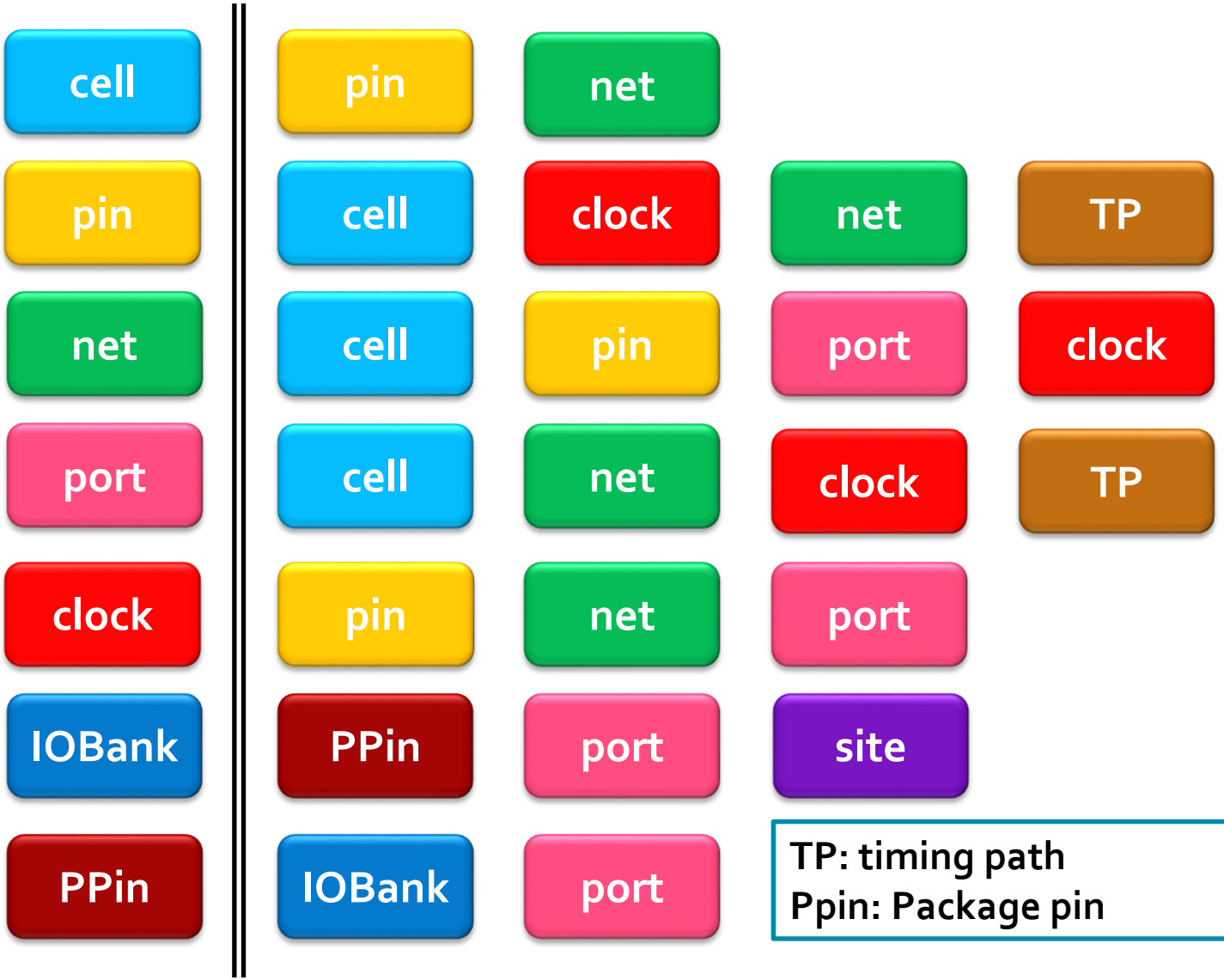
① *get_ports -filter {DIRECTION == IN && NAME !~ "*RESET*"}*

② *-filter {IS_PRIMITIVE && !IS_LOC_FIXED}*

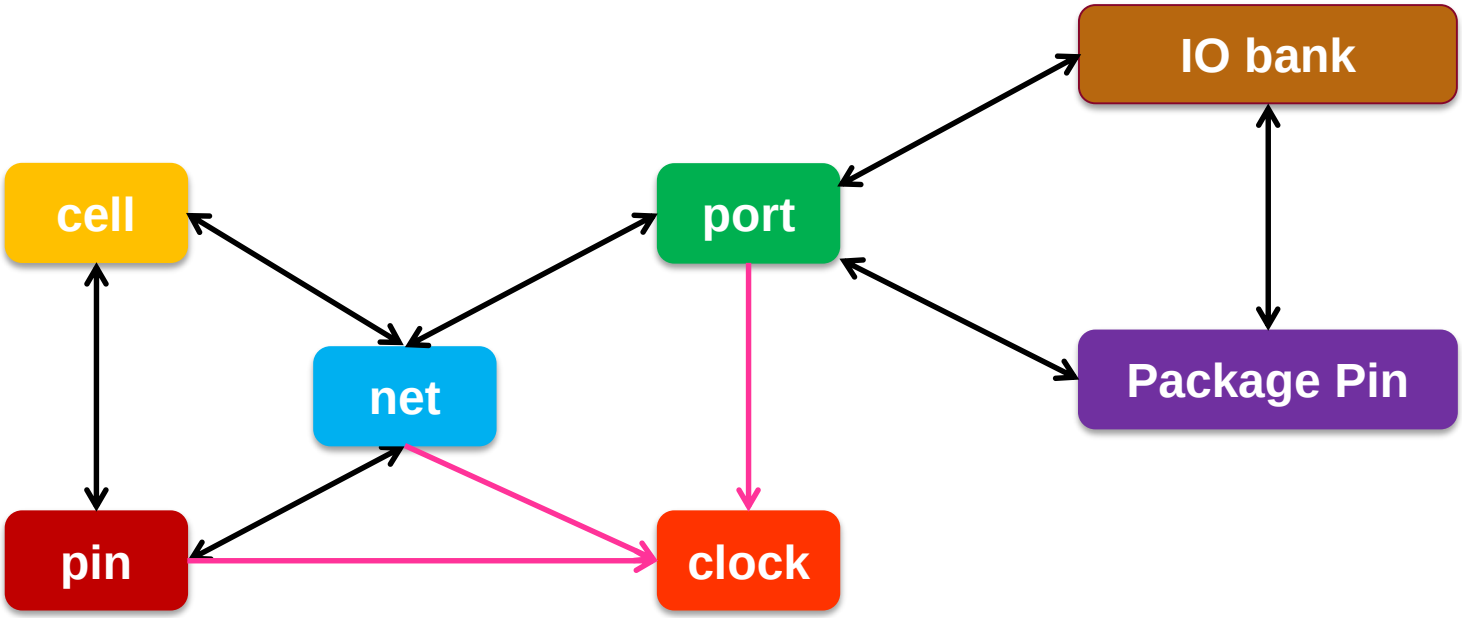
③ *get_cells -hier {*State* *reg*}*

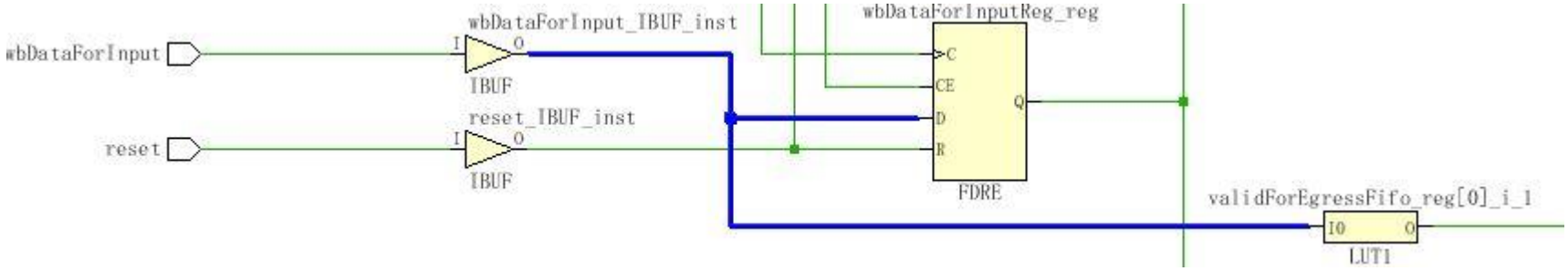
④ *get_cells ←→ get_cells **

-of_objects


cell
pin
net
pin
cell
clock
net
TP
net
cell
pin
port
clock
port
cell
net
clock
TP
clock
pin
net
port
IOBank
PPin
port
site
PPin
IOBank
port
TP: timing path
Ppin: Package pin

Object Relationships in Vivado Design Suite



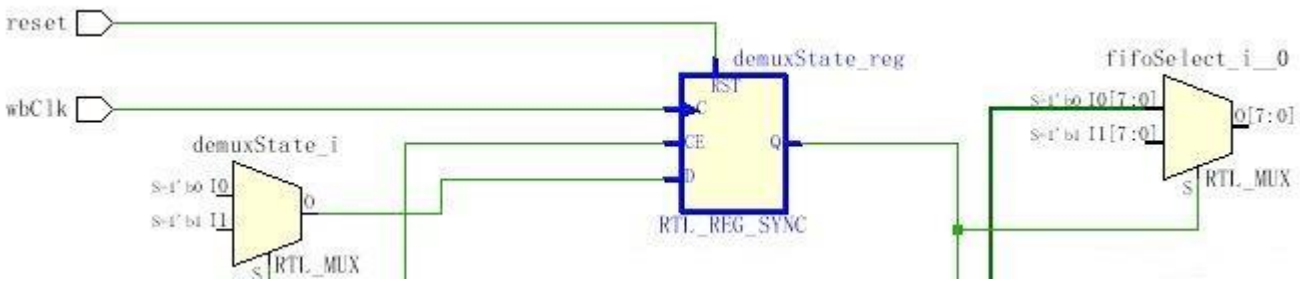


Example:

get_cells -of [get_nets -of [get_pins -of [get_cells wbDataForInput_IBUF_inst] -filter {DIRECTION==OUT}]]

wbDataForInputReg_reg validForEgressFifo_reg[0]_i_1 wbDataForInput_IBUF_inst

-of_objects Examples



- **Get the pins of this blue cell**

```
get_pins -of [get_cells demuxState_reg]
```

```
demuxState_reg/C demuxState_reg/CE demuxState_reg/RST
demuxState_reg/Q demuxState_reg/D
```

- **Get cell by the given pin D**

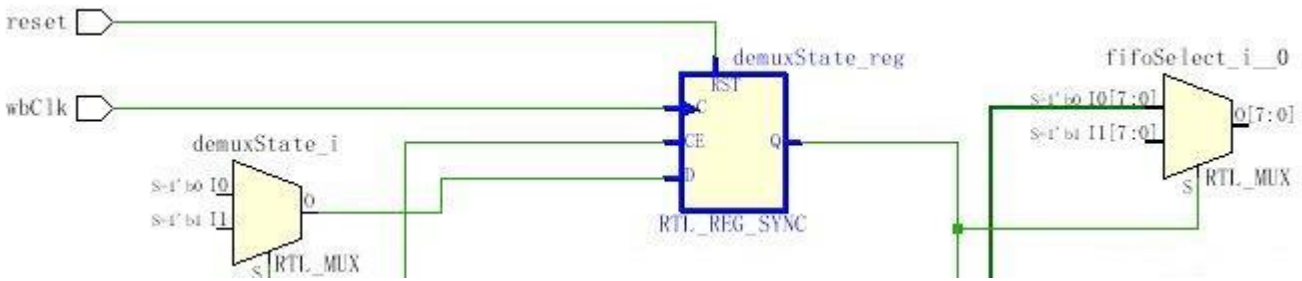
```
get_cells -of [get_pins demuxState_reg/D]
```

```
demuxState_reg
```

- **Get cell connecting to the given net wbClk**

```
get_cells -of [get_nets wbClk]
```

-of_objects Examples



- **Get nets connecting to the given cell**

```
get_nets -of [get_cells demuxState_reg]
```

```
wbClk demuxState1_out reset demuxState demuxState0_out
```

- **Get net connecting to the given pin D**

```
get_nets -of [get_pins demuxState_reg/D]
```

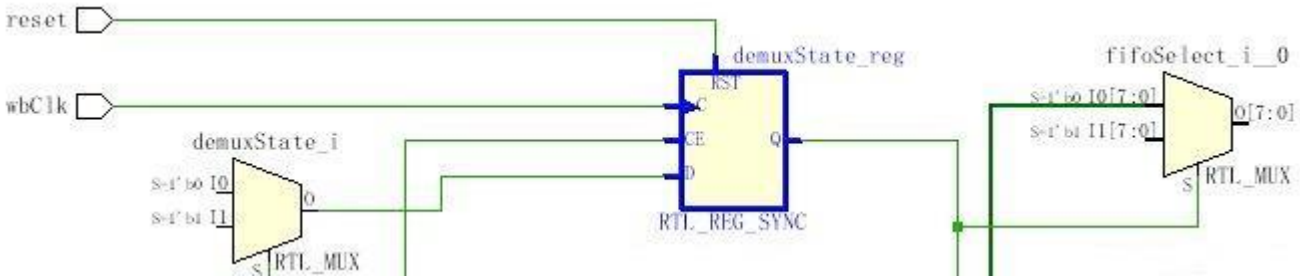
```
demuxState0_out
```

- **Get clocks connecting to the given pin C**

```
get_clocks -of [get_pins demuxState_reg/C]
```

```
wbClk
```

-of_objects Examples



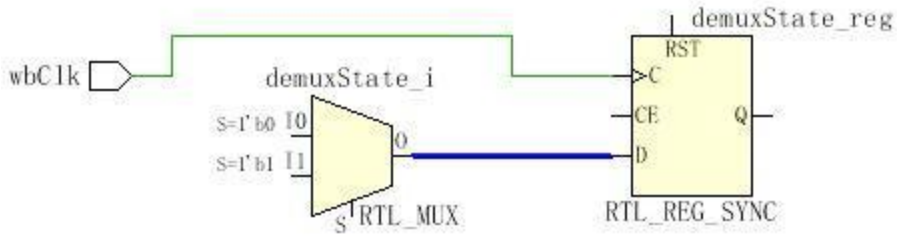
- Get ports connecting to the given clock wbClk

```
get_ports -of [get_clocks wbClk]
wbClk
```

- Get cells connecting to the given net

```
get_cells -of [get_nets demuxState0_out]
demuxState_reg demuxState_i
```

Questions



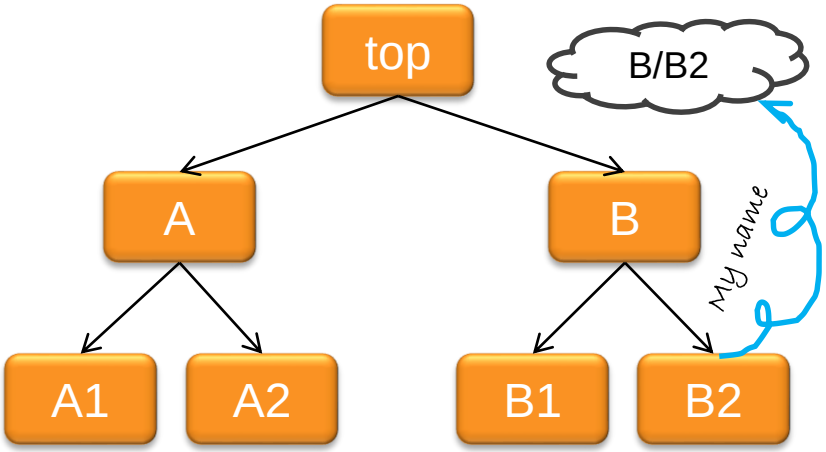
- **What's the answer?**

```
get_cells -of [get_nets demuxState0_out]
```

- **If we want to get the cell demuxState_reg by the net demuxStateo_out, what's the tcl command?**

```
get_cells -of \
[get_pins -of [get_nets demuxState0_out] -filter "DIRECTION==IN"]
```

-hierarchical Explanations



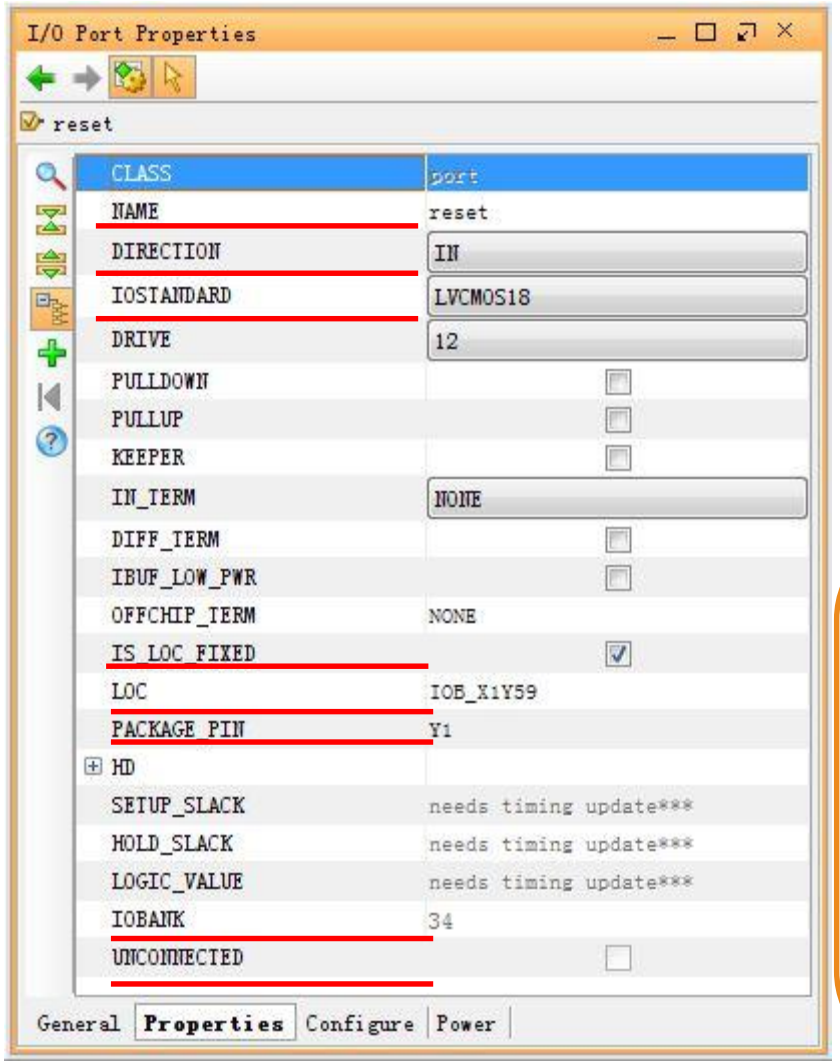
#the principle of -hierarchical

set result {}
foreach hcell [list "" A B A/a1 A/a2 B/b1 B/b2] {
current_instance $hcell ;# Move scope to $hcell
set result [concat $result [get_cells <pattern>]]
current_instance ;# Return scope to design top-level
}

```
get_cells {A* B*}
A B
get_cells -hier {A* B*}
A A/A1 A/A2 B B/B1 B/B2
get_cells B/*
B/B1 B/B2
get_cells -hier B/*
No cells matched 'B/*'
get_cells -hier -regexp B/.*
B/B1 B/B2
get_cells -hier -filter {NAME=~*B*}
B B/B1 B/B2
get_cells -hier -regexp .*B.*
B B/B1 B/B2
```

- match the specified name pattern at each level of the design hierarchy, and not against the full hierarchical name of an object
- The specified search pattern must not include the hierarchical separator otherwise no object will be returned
- When -hierarchical is used with -regexp, the specified search string is matched against the full hierarchical name

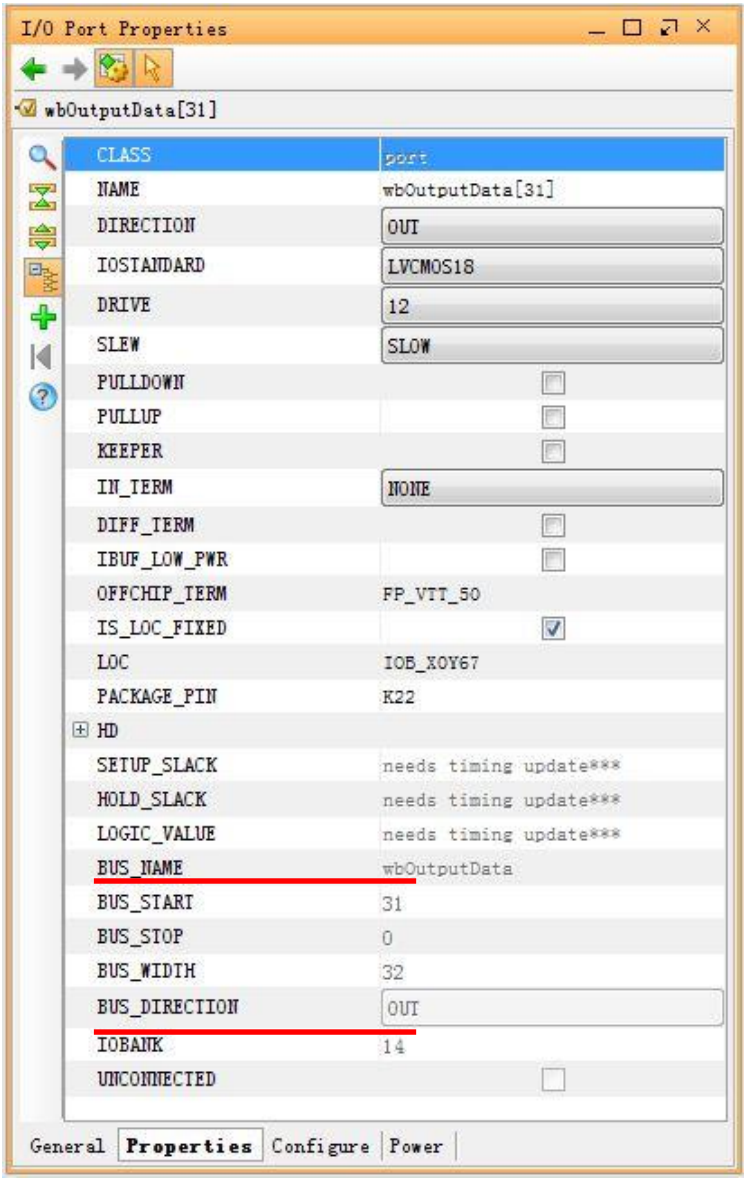
I/O Port Properties for Scalar Ports



- NAME: It is consistent with that described in source code
- DIRECTION: IN OUT or INOUT
- PACKAGE_PIN: ports pin location
- IOBANK: ports bank location
- UNCONNECTED: 1 or 0

```
get_ports -filter {DIRECTION==IN}
get_ports -filter {IOSTANDARD==LVCOMS18}
get_ports -filter {UNCONNECTED==1}
get_ports -filter {PACKAGE_PIN==""}
get_ports -filter {IOBANK==34}
```

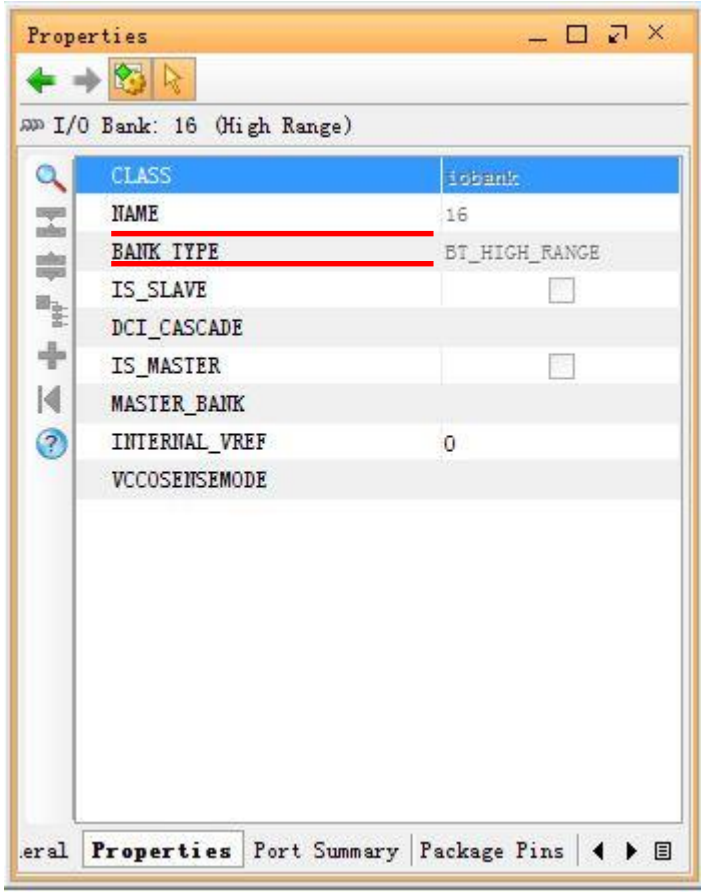
I/O Port Properties for Bus Ports



- NAME, BUS_NAME
- BUS_WIDTH
- BUS_DIRECTION

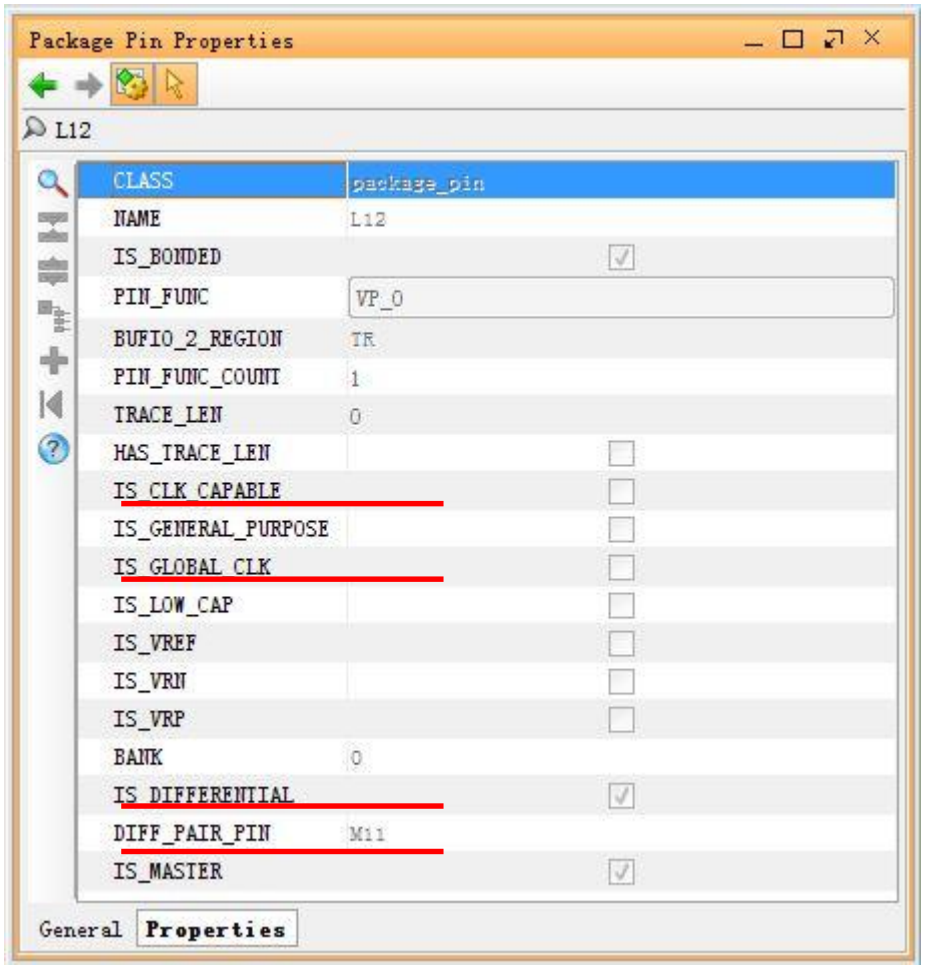
```
get_ports -filter {BUS_NAME != ""}
```

I/O Bank and Package Pin Properties



get_iobanks

```
get_iobanks -of [get_ports reset]
get_package_pins -of [get_ports bftClk
get_package_pins -of [get_iobanks 34]
```



get_package_pins

Example:

Get IOs having "Data" string in their names:

```
get_ports *data*
```

Get input ports with IO standards LVCOMS18:

```
get_ports -filter {DIRECTION==IN && IOSTANDARD==LVCMOS18}
```

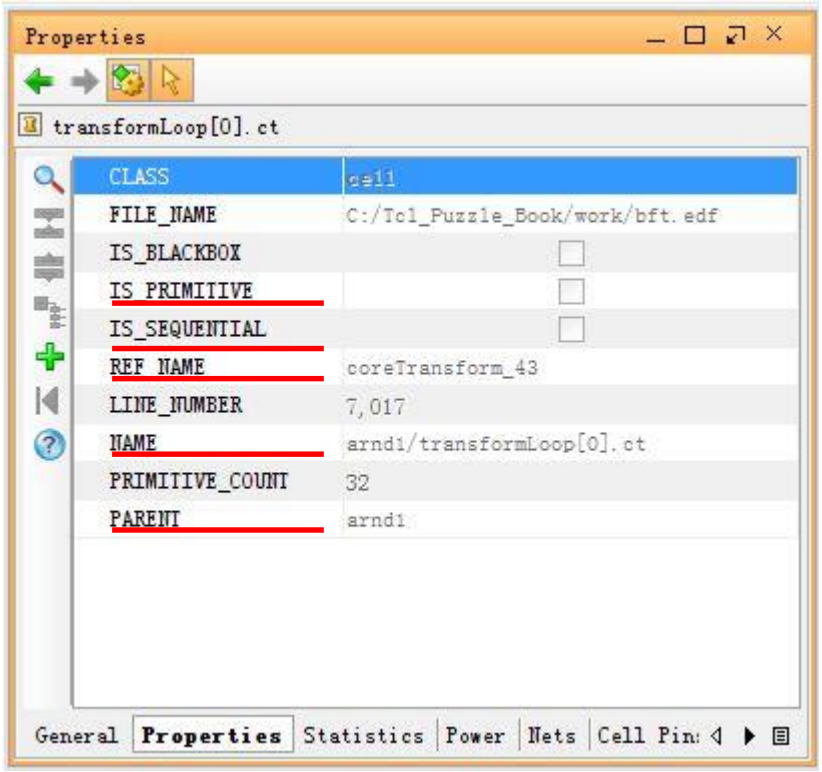
```
set myinport [all_inputs]
set myport [filter $myinport {IOSTANDARD==LVCMOS18}]
```

Get IO bank where H4 is located:

```
get_iobanks -of_objects [get_package_pins H4]
```

Working with Cells

- **current_instance:** Set the current instance in the design hierarchy to the specified instance cell or to the top module
- **get_cells:** Gets a list of cell objects in the current design that match a specified search pattern



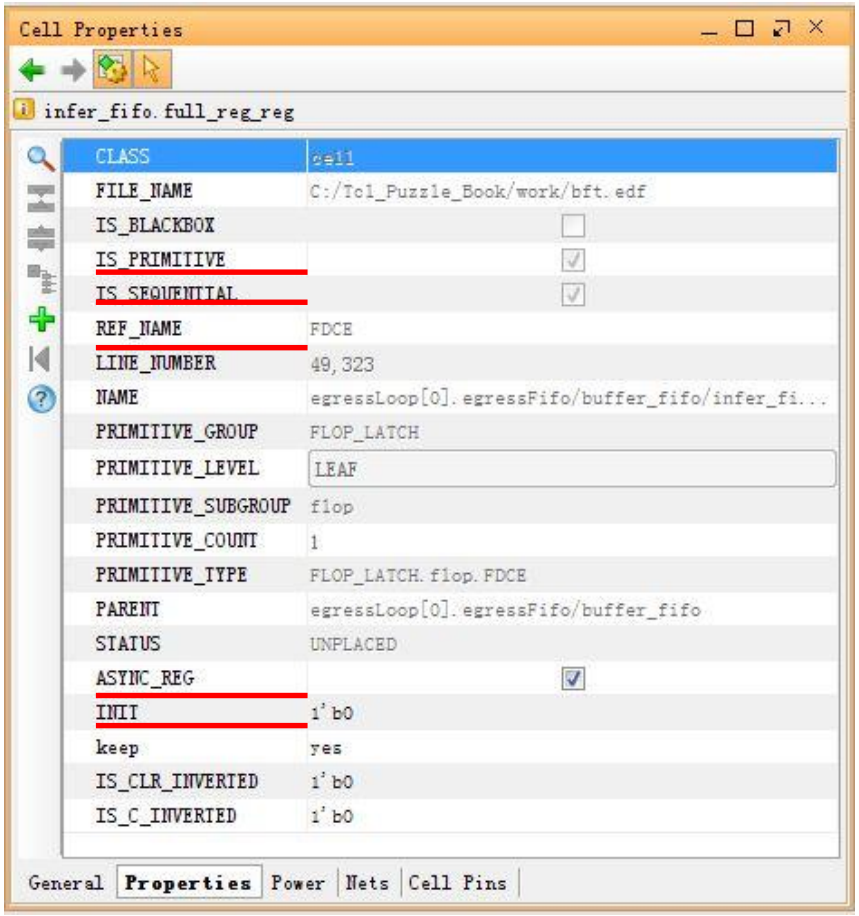
- **REF_NAME:**
1. **In verilog:**
   module counter
2. **In VHDL**
   entity counter is
- **NAME:**
   Instantiation name with hierarchy

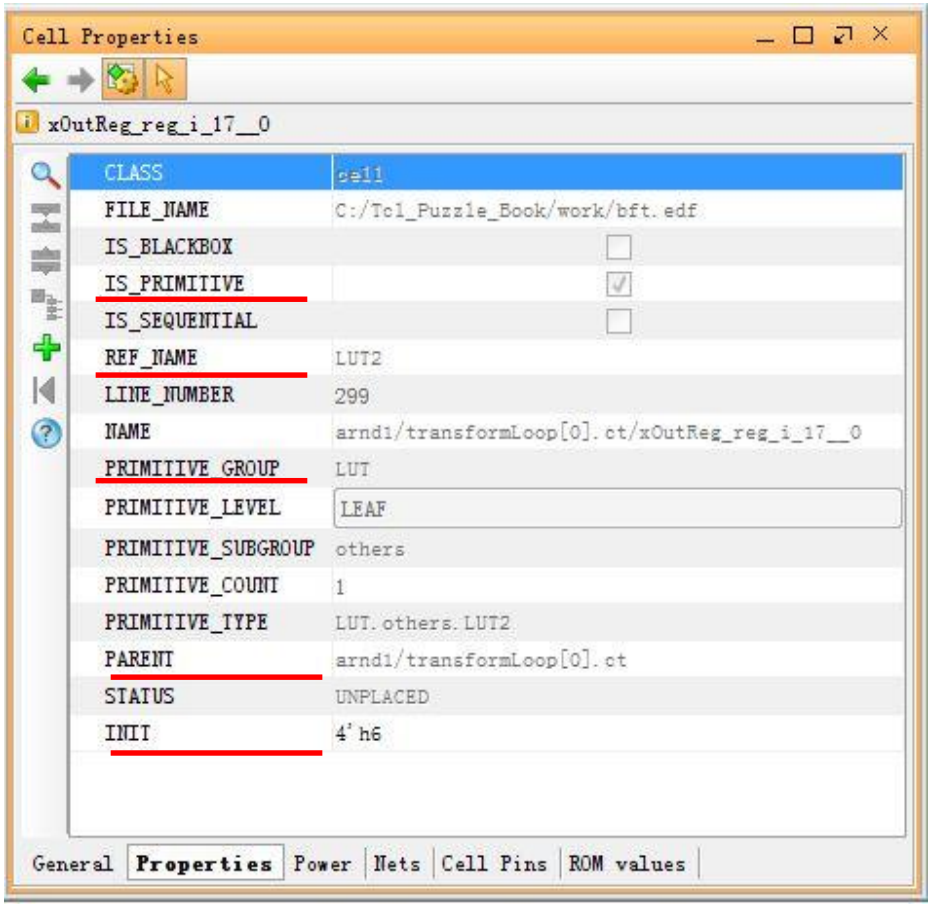
```
get_cells -quiet –hierarchical\
-filter {REF_NAME =~ FD*} *rd_*

get_cells -quiet –filter\
{REF_NAME =~ FD*}\
egressLoop[0].egressFifo/buffer_fifo/*
```

If the cell in the top level, PARENT will be empty.

Properties of FDCE and LUT





Examples

Get cells in arnd1

```
get_cells arnd1/*
```

```
current_instance -quiet [get_cells arnd1];
set hb1_cells [get_cells *]
current_instance -quiet
```

Get black box and primitive cells in top level

```
set all_top_level_blocks [get_cells *]
set hierb_list [filter -quiet $all_top_level_blocks {IS_BLACKBOX  == 0 && IS_PRIMITIVE  == 0}]
set primitive_list [filter -quiet $all_top_level_blocks {IS_BLACKBOX  == 0 && IS_PRIMITIVE == 1}]
set blackb_list [filter -quiet $all_top_level_blocks {IS_BLACKBOX  == 1 && IS_PRIMITIVE == 0}
```

Get all sequential cells from entire design and report its REF_NAME

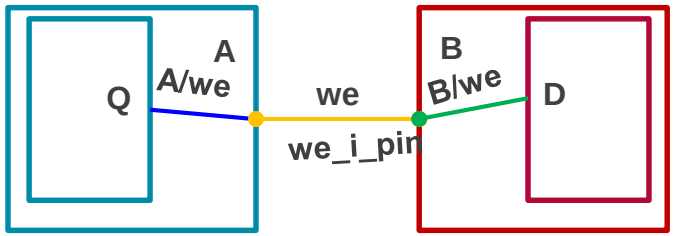
```
set list_seq_cells [get_cells -hierarchical -filter  {IS_SEQUENTIAL == 1} *]
set list_seq_type  [get_property REF_NAME $list_seq_cells]
```

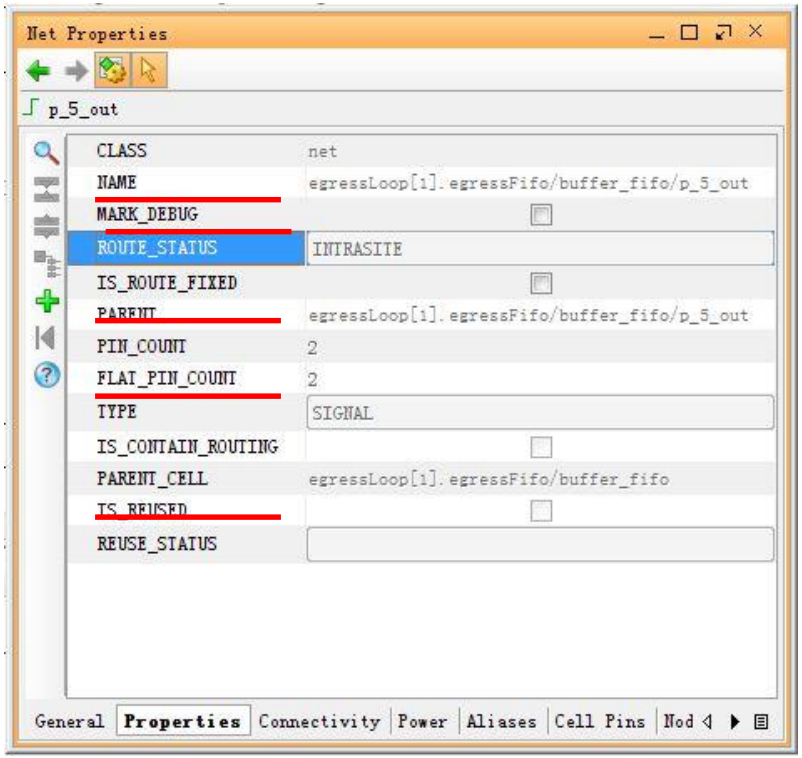
Get property and change it

```
get_property INIT [get_cells A/B/data_reg]
Set_property INIT 1’b0 [get_cells A/B/data_reg]
```

Working with Nets

- **get_nets :** Gets a list of nets in the current design that match a specified search pattern
  - -segments: Get all the segments of a hierarchical net, across all levels of the hierarchy
  - -boundary_type: Gets the net segment at the level of a specified hierarchical pin. Values: upper, lower and both. Default: upper





- **Only B/we is obtained**

```
get_nets B/we
```

- **A/we we B/we are all obtained**

```
get_nets -segment B/we
```

- **Get net 'we'**

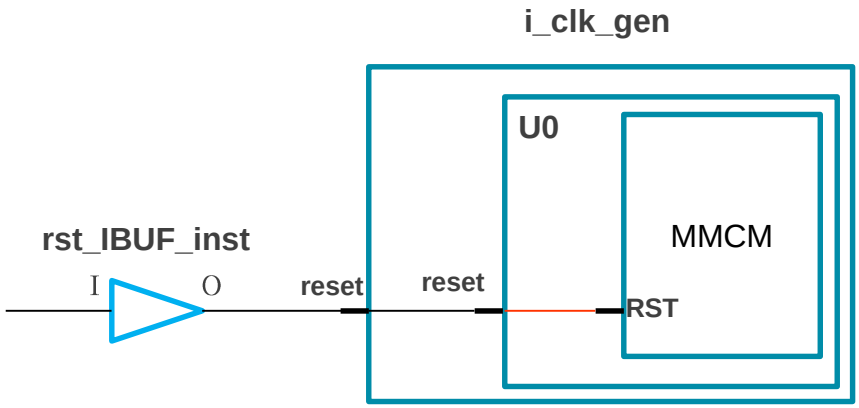
```
get_nets -boundary_type upper \
-of [get_pins B/we_i_pin]
```

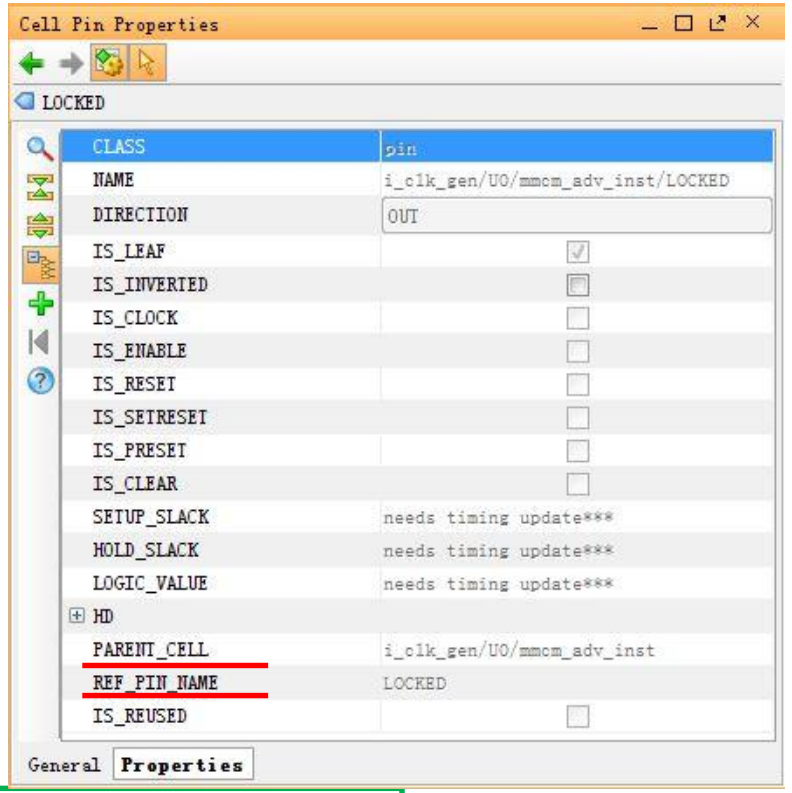
- **Set property**

```
set_property MARK_DEBUG true [get_net {control addr}]
```

Working with Pins

- **get_pins:** Gets a list of pin objects in the current design that match a specified search pattern
  - -leaf: Include leaf pins, from primitive or black box cells, for the objects specified with the -of_object argument



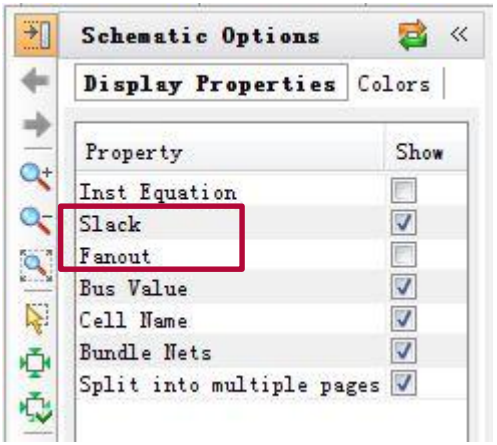


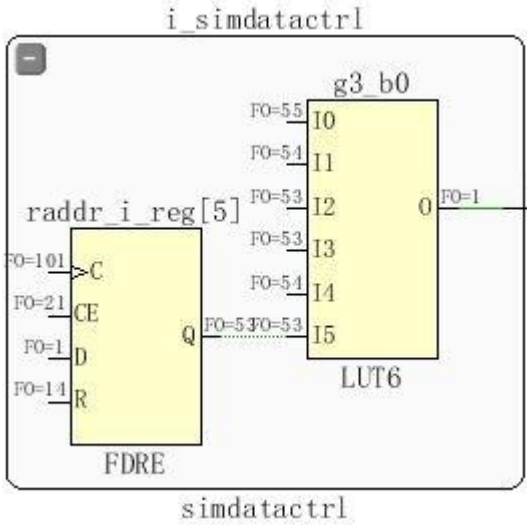

```
set mynet [get_nets i_clk_gen/U0/reset]
set mypina [get_pins -leaf -of $mynet]
set mypinb [get_pins -of $mynet]
puts "Leaf pins: $mypina"
puts "Pins: $mypinb
```

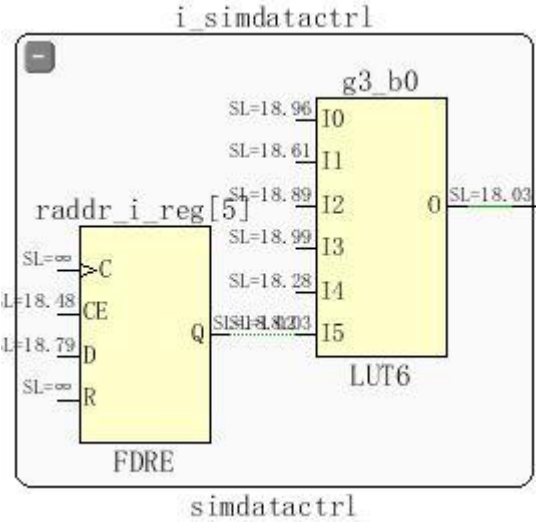
```
Leaf pins: i_clk_gen/U0/mmcm_adv_inst/RST rst_IBUF_inst/O
Pins: i_clk_gen/U0/mmcm_adv_inst/RST i_clk_gen/U0/reset
```

Working with Schematic Viewer

- **get_selected_objects:** Gets the objects currently selected in the Vivado IDE
- **select_objects:** Selects the specified object in the appropriate open views in the GUI mode
- **unselect_objects**: Unselects the specified object or objects that were previously selected by the select_objects command







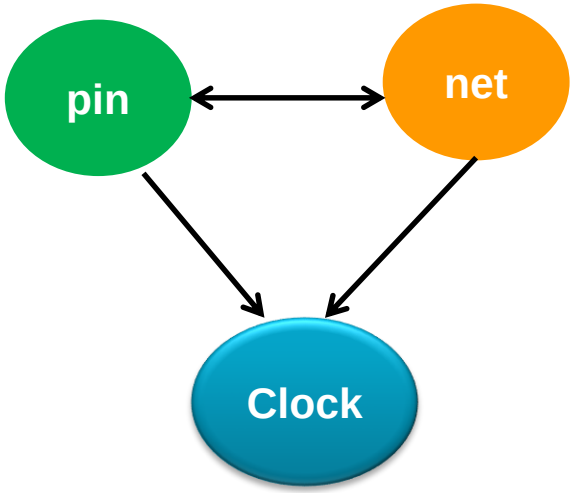
- **Before select objects, you should unselect the previous one firstly**

```
unselect_objects -quiet [get_selected_objects]
select_objects [get_cells i_simdatactrl]
```

After **report_timing_summary**, slack can be shown

Working with Clocks

report_clock_interaction	Report on inter clock timing paths and unclocked registers
report_clock_networks	Report clock networks
report_clock_utilization	Report information about clock nets in design
report_clocks	Report clocks
get_clocks	Get a list of clocks in the current design
all_clocks	Get a list of all clocks in the current design
get_generated_clocks	Get a list of generated clocks in the current design
create_clock	Create a clock object
create_generated_clock	Create a generated clock object



- **net: clock net**
- **pin: clock pin**

```
get_clocks {*clock *ck *Clk}
get_clocks -include_generated_clocks wbClk
report_property -all [get_clocks wbClk]
get_clocks -of [get_pins {i_firctrl/raddrcoe_i_reg[3]/C}]
get_clocks -of [get_nets i_firctrl/CLK]
get_clocks -filter {PERIOD<20.0}
```

Summary

- **Five Tcl commands are widely used in**
  - Timing constraints
  - Timing analysis
  - Debugging
- **They can be used under**
  - Elaborated design
  - Synthesized design
  - Implemented design