Vivado从此开始（To Learn Vivado From Here）

Vivado
从此开始
To Learn Vivado From Here
高亚军 编著
· 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
中国工信出版集团
電子工業出版社

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

XILINX

ALL PROGRAMMABLE™

TCL, Vivado One World

Part 1

Lauren Gao

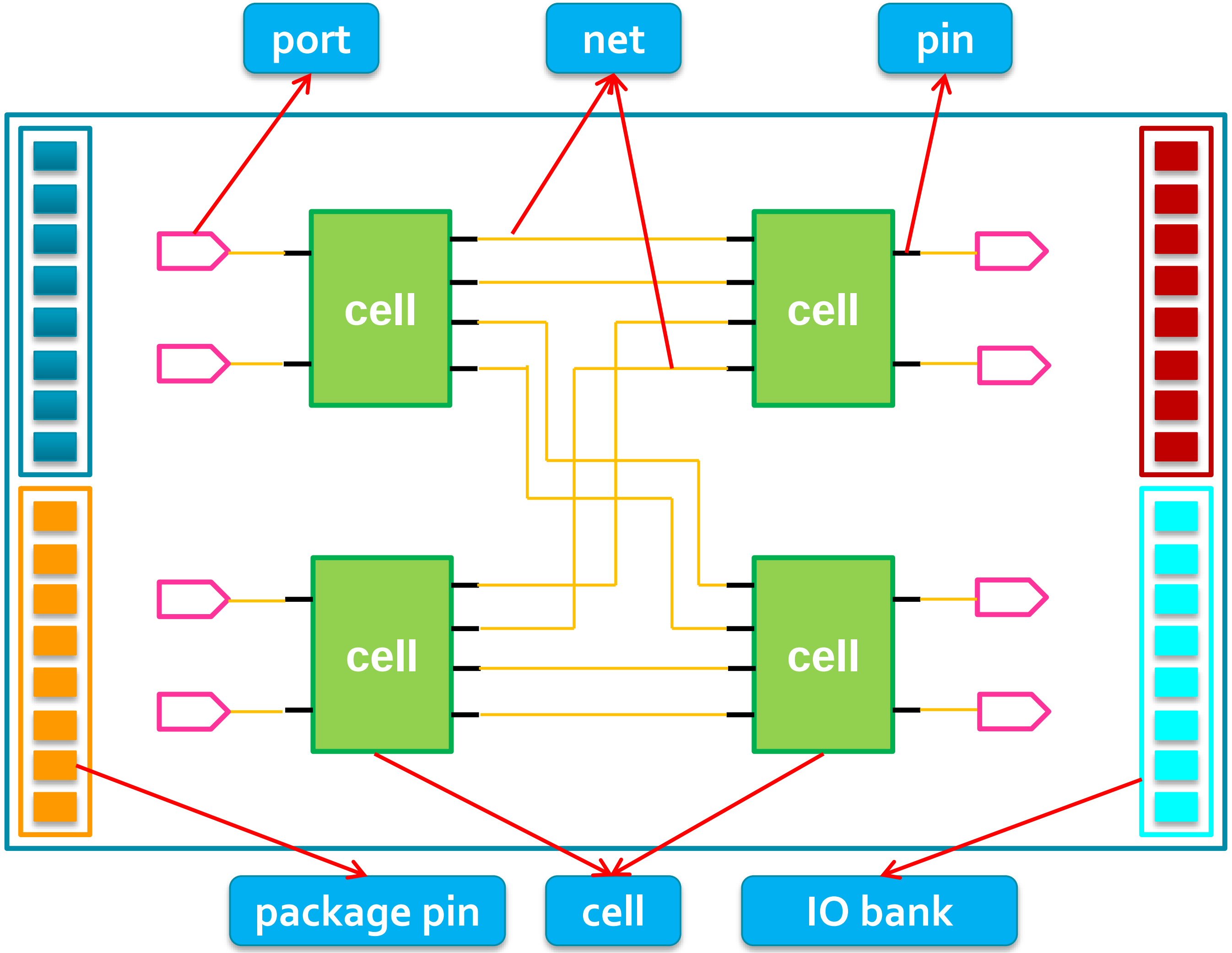
Agenda

- TCL background from Vivado view
- Edit synthesized netlist with TCL in Vivado
- Customize various reports with TCL in Vivado
- Interact with Vivado by TCL

Agenda

- **TCL background from Vivado view**
- **Edit synthesized netlist with TCL in Vivado**
- **Customize various reports with TCL in Vivado**
- **Interact with Vivado by TCL**

Objects in Vivado Netlist



- Each object has its own property
- Some properties are read-only
- Some properties are editable
- Object can be found by filtering with certain property

Five Commonly Used TCL Commands in Vivado

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

A.

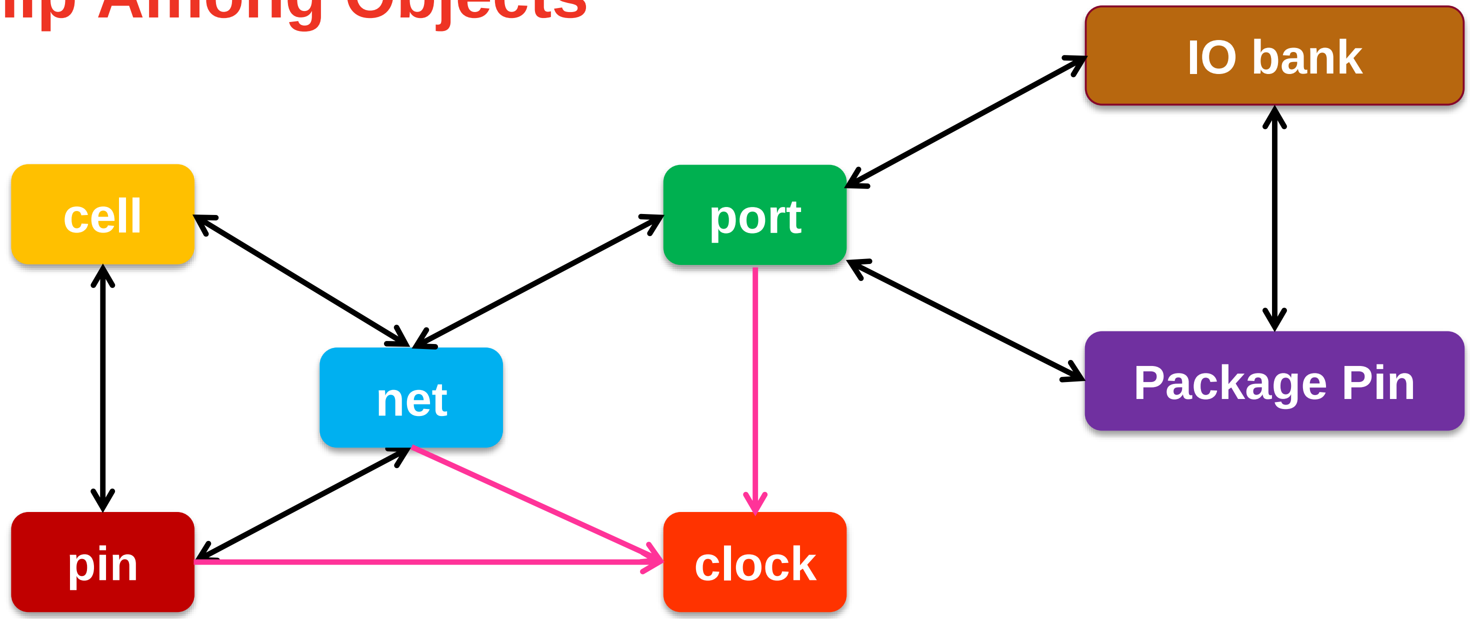
String Comparison	
equal	==
not equal	!=
match	=~
not match	!~

B. Multiple filter expressions

C. Boolean type properties: AND(&&), OR(||)

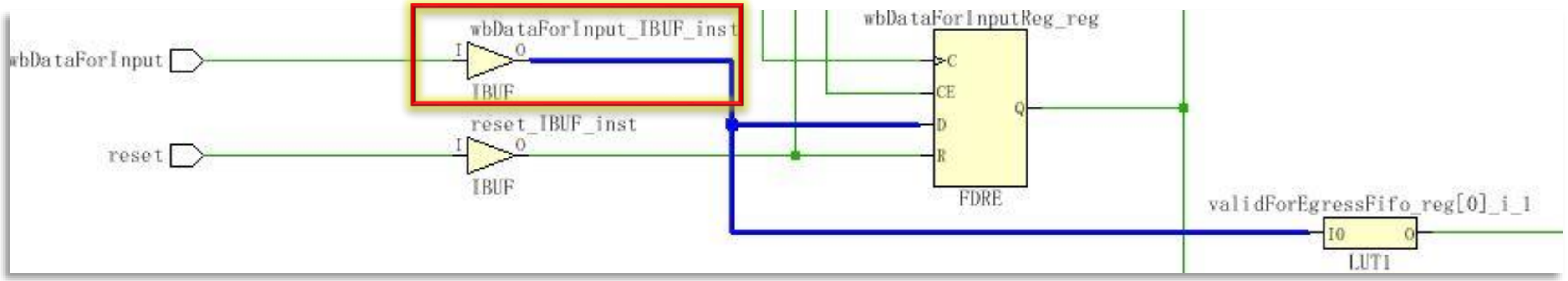
```
①  get_ports -filter {DIRECTION == IN && NAME !~ "*RESET*"}
②  get_cells -filter {IS_PRIMITIVE && !IS_SEQUENTIAL}
③  get_cells -hier {*State*  *reg*}
④  get_cells ←→ get_cells *
```

Relationship Among Objects



```
get_cells -of_objects {pins, timing paths, nets, bels or sites}
get_clocks -of_objects {nets, ports, or pins}
get_nets -of_objects {pins, ports, cells, timing paths or clocks}
get_pins -of_objects {cells, nets, bel pins, timing paths or clocks}
get_ports -of_objects {nets, instances, sites, clocks, timing paths, io standards, io
banks, package pins}
```

Relationship Among Objects



Example:

Input:

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

Output:

```
wbDataForInputReg_reg validForEgressFifo_reg[0]_i_1 wbDataForInput_IBUF_inst
```

Agenda

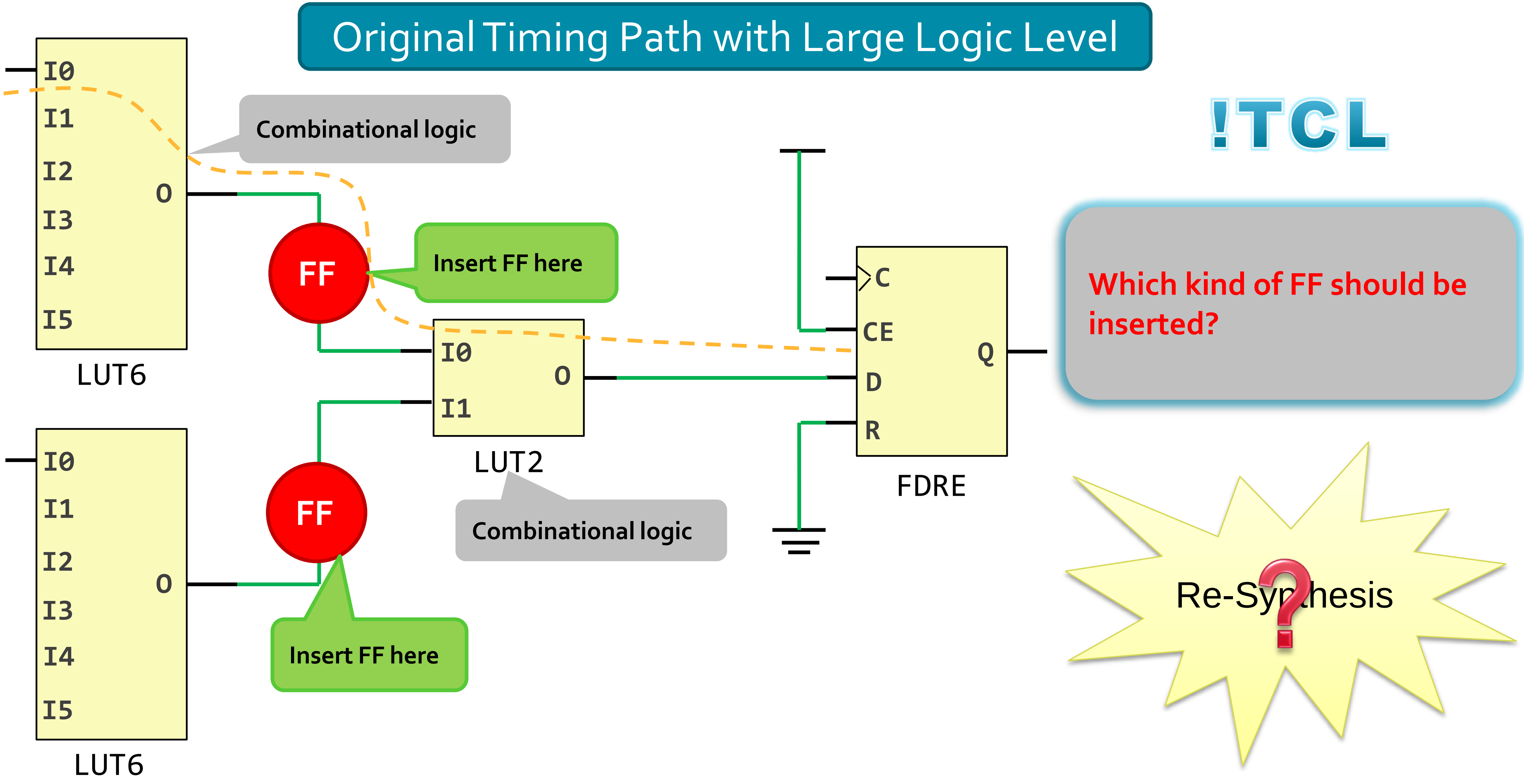
- TCL background from Vivado view
- **Edit synthesized netlist with TCL in Vivado**
- Customize various reports with TCL in Vivado
- Interact with Vivado by TCL

Some Applications of Editing Synthesized Design Netlist with TCL

- **Insert FF in the netlist**
  - Insert FF in the large logic level timing path
  - Insert FF before/after DSP48E1
  - Insert FF before/after RAMB36E1
- **Reduce fanout for large fanout nets**
  - Replicate register for large fanout nets
  - Insert BUFG for large fanout nets
- **Modify probe net for test**
  - Export internal net to FPGA pad for test with scope or spectrum analyzer
- **Remove unwanted objects from the netlist**
  - Remove cells and nets

- ➢ **Save run time**
  - Do not re-synthesize design
- ➢ **Identify issue fast**
  - Avoid different synthesis result

Example 1: Insert FF in the Large Logic Level Timing Path



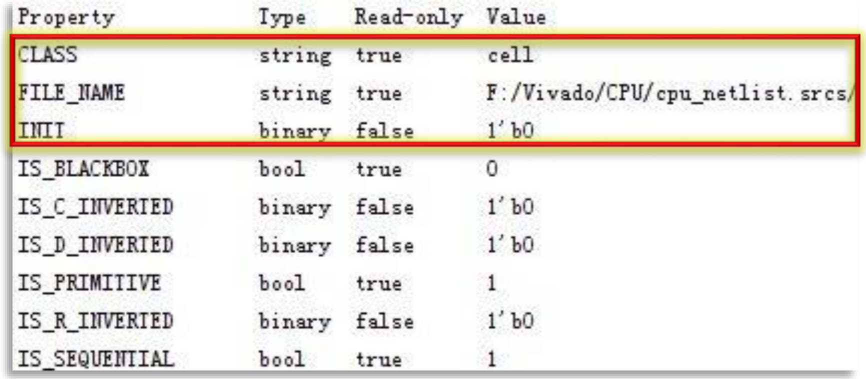
Confirm the Type of New FF

FDCE	Asynchronous Clear	D, CE, C	CLR
FDPE	Asynchronous Preset	D, CE, C	PRE
FDRE	Synchronous Reset	D, CE, C	R
FDSE	Synchronous Set	D, CE, C	S

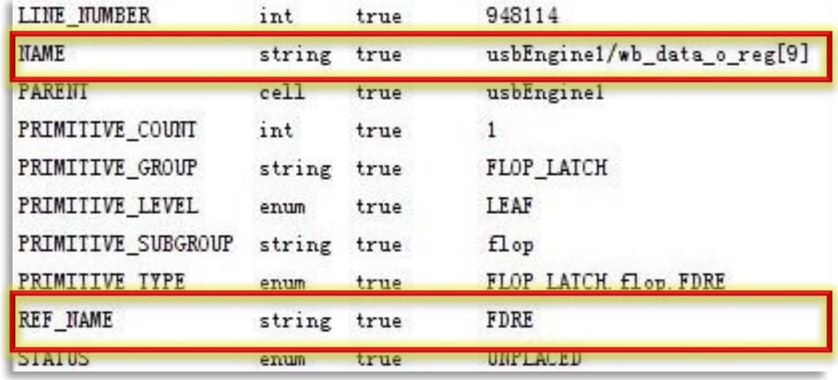
- **Three items about new FF**
  - NAME
  - REF_NAME
  - Initial value
- **Original FF**
  - NAME: `local_if/data_buffer/raddr_reg`
    - `file dirname [get_property NAME [get_cells local_if/data_buffer/raddr_reg]]`
  - REF_NAME: FDRE
    - `get_property REF_NAME [get_cells local_if/data_buffer/raddr_reg]`
  - Initial Value: 1’b0
    - `get_property  INIT [get_cells local_if/data_buffer/raddr_reg]]`

Set Properties for New FF

- **Three items about new FF**
  - NAME and REF_NAME
    - `create_cell -ref FDRE $new_FF_name`
  - Initial value
    - `set_property INIT $INIT_value [get_cells $new_FF_name]`
- **Three very useful Tcl scripts in Vivado**
  - `report_property`, `get_properety` and `set_property`

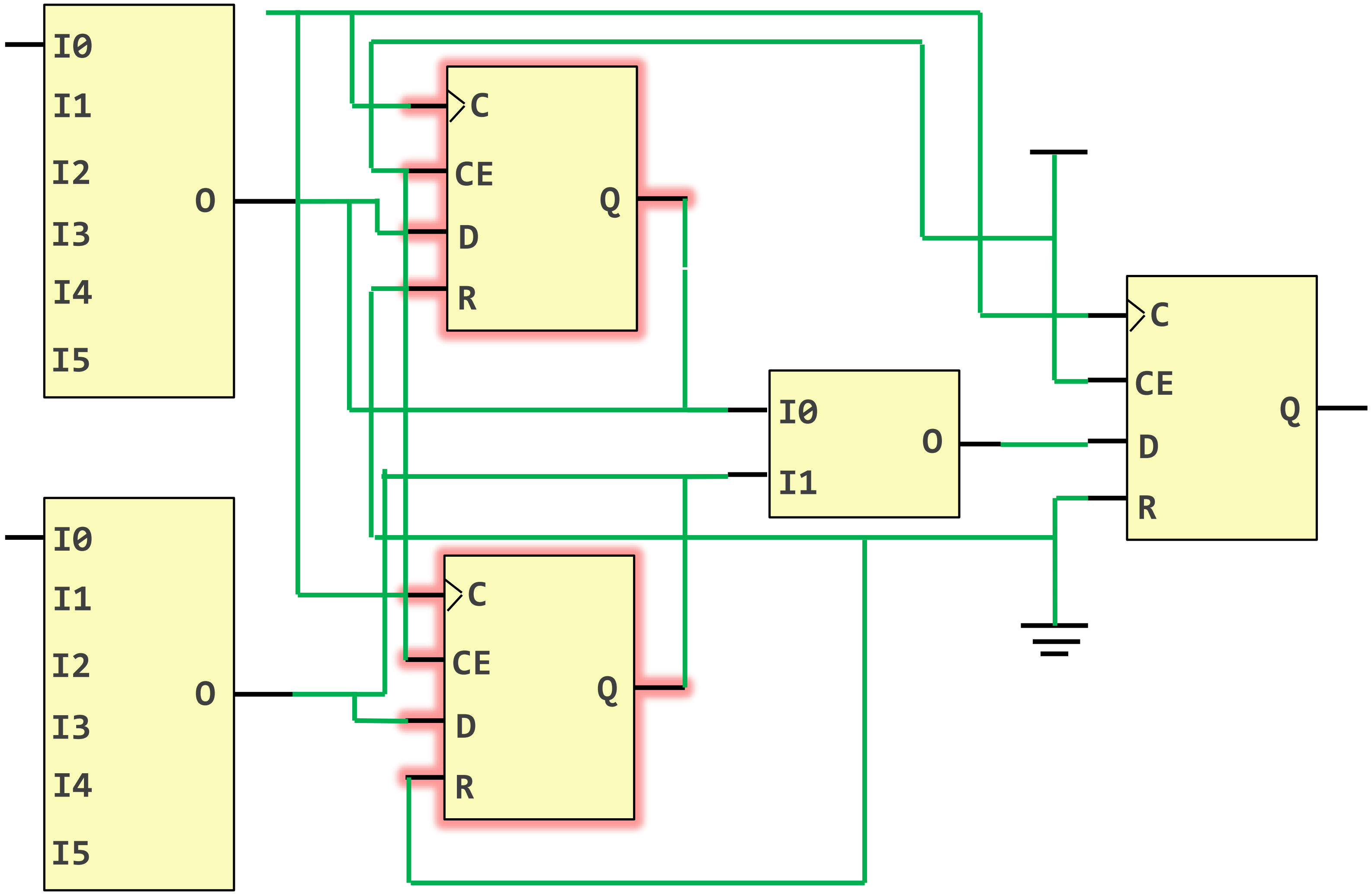


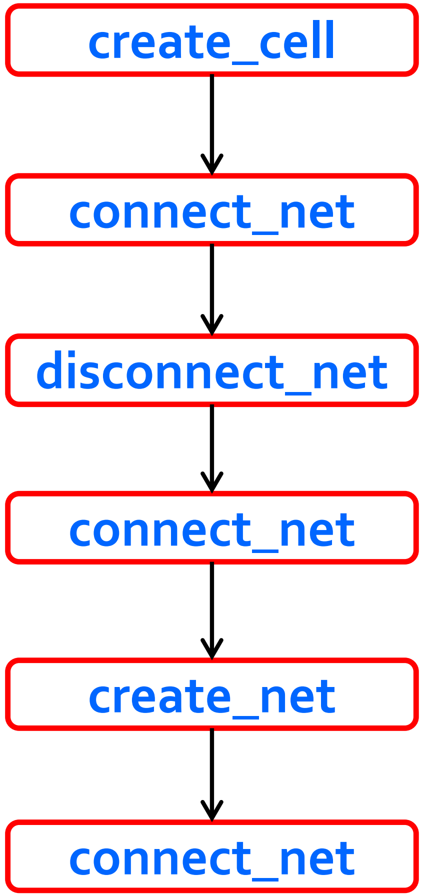
Property	Type	Read-only	Value
CLASS	string	true	cell
FILE_NAME	string	true	F:/Vivado/CPU/cpu_netlist.srcs/
INIT	binary	false	1'b0
IS_BLACKBOX	bool	true	0
IS_C_INVERTED	binary	false	1'b0
IS_D_INVERTED	binary	false	1'b0
IS_PRIMITIVE	bool	true	1
IS_R_INVERTED	binary	false	1'b0
IS_SEQUENTIAL	bool	true	1



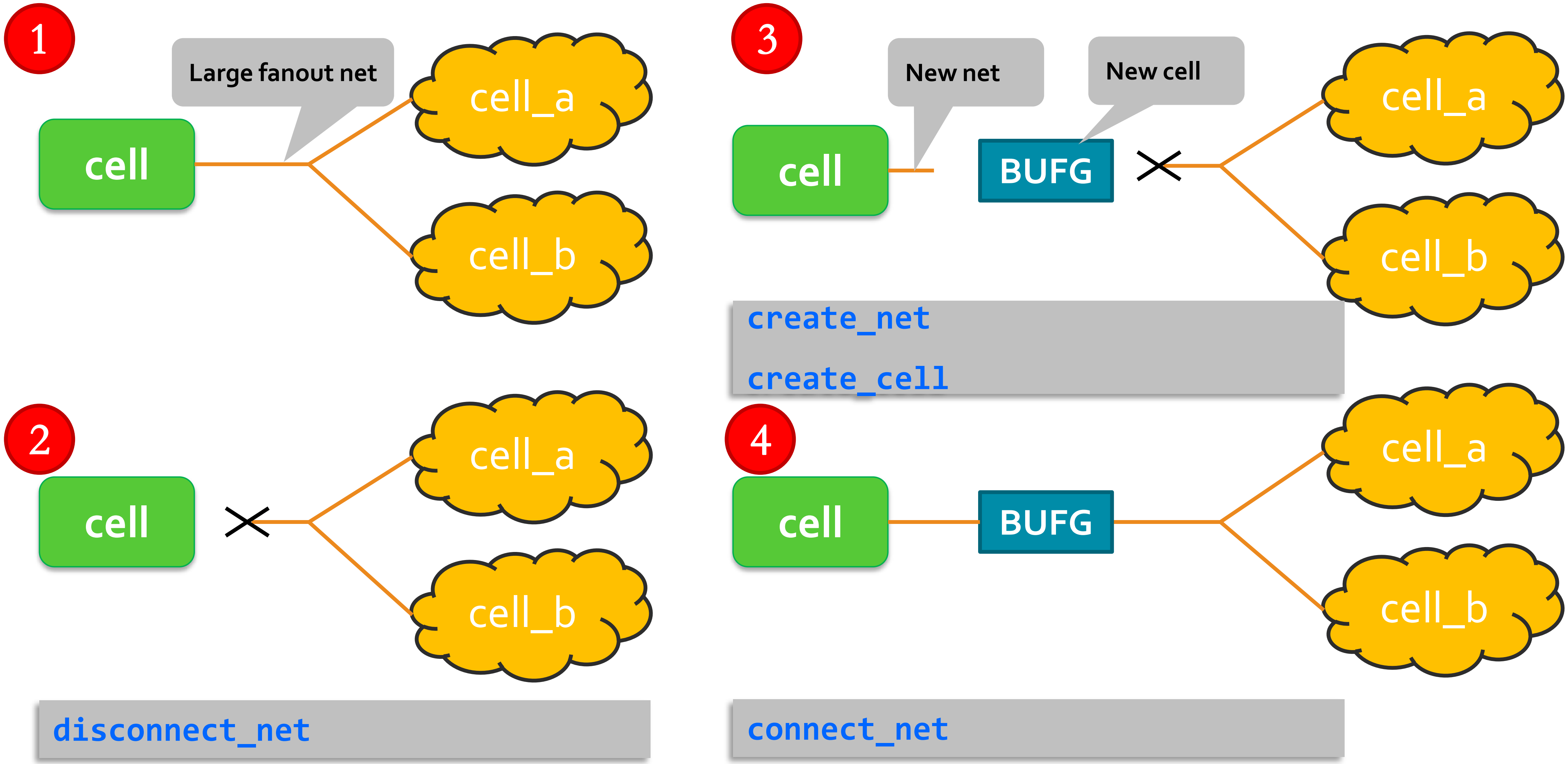
LINE_NUMBER	int	true	948114
NAME	string	true	usbEngine1/wb_data_o_reg[9]
PARENT	cell	true	usbEngine1
PRIMITIVE_COUNT	int	true	1
PRIMITIVE_GROUP	string	true	FLOP_LATCH
PRIMITIVE_LEVEL	enum	true	LEAF
PRIMITIVE_SUBGROUP	string	true	flop
PRIMITIVE_TYPE	enum	true	FLOP_LATCH.flop.FDRE
REF_NAME	string	true	FDRE
STATUS	enum	true	UNPLACED

Insert New FF in the Original Netlist Schematic View





Example 2 : Insert BUFG for Large Fanout Nets

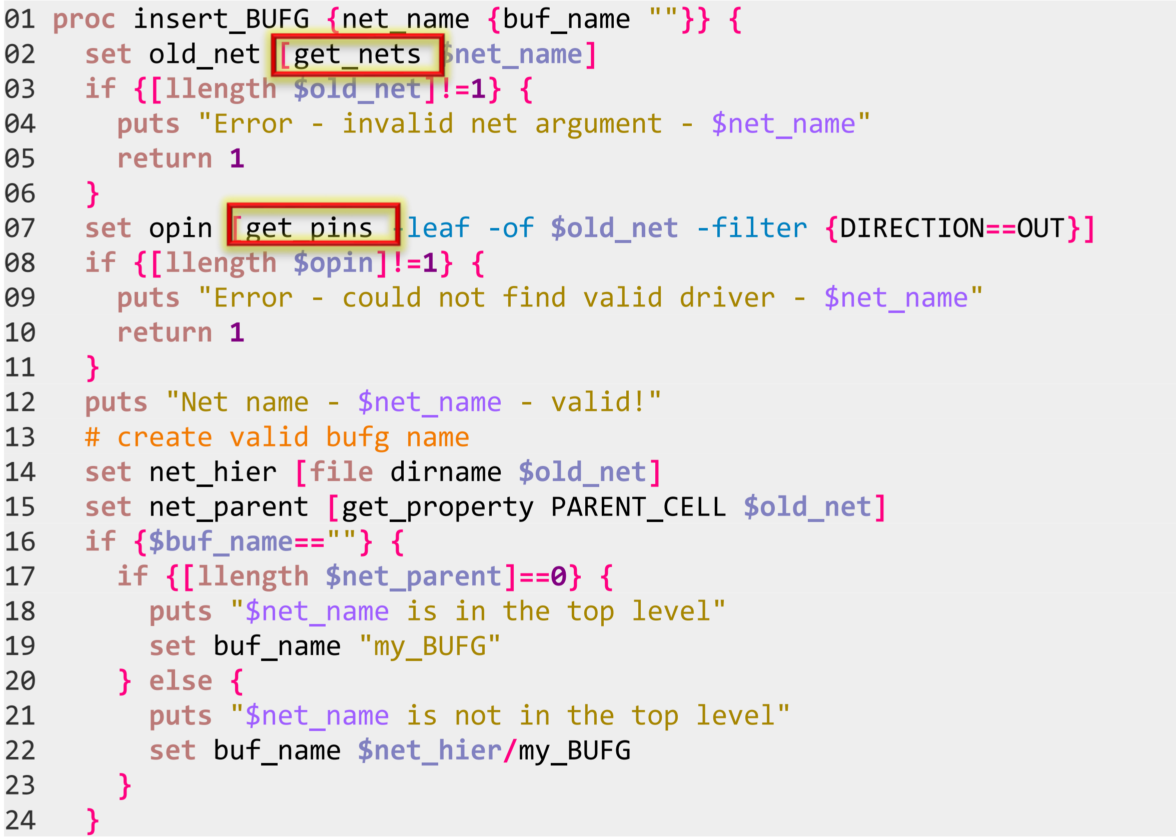


```
create_net
create_cell
```

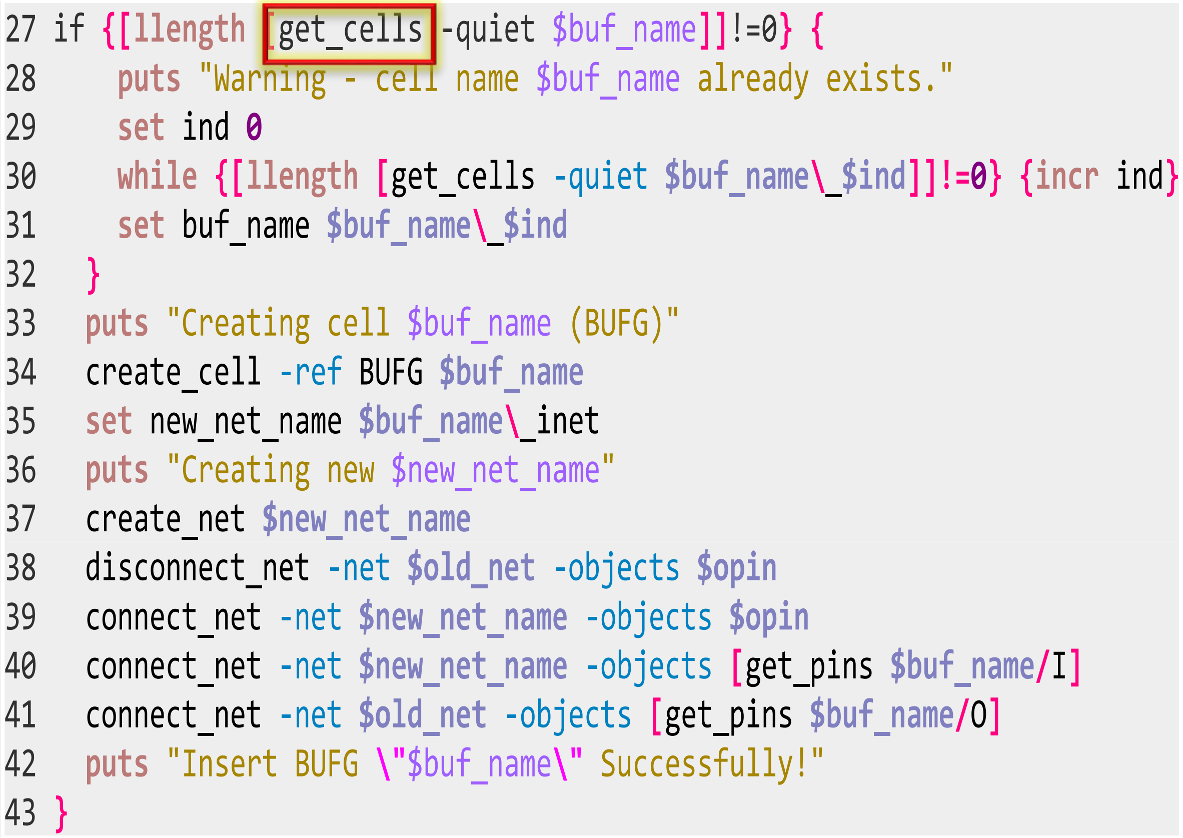
```
disconnect_net
```

```
connect_net
```

Insert BUFG TCL Source Code



```tcl
proc insert_BUFG {net_name {buf_name ""}} {
  set old_net [get_nets $net_name]
  if {[llength $old_net]!=1} {
    puts "Error - invalid net argument - $net_name"
    return 1
  }
  set opin [get_pins -leaf -of $old_net -filter {DIRECTION==OUT}]
  if {[llength $opin]!=1} {
    puts "Error - could not find valid driver - $net_name"
    return 1
  }
  puts "Net name - $net_name - valid!"
  # create valid bufg name
  set net_hier [file dirname $old_net]
  set net_parent [get_property PARENT_CELL $old_net]
  if {$buf_name==""} {
    if {[llength $net_parent]==0} {
      puts "$net_name is in the top level"
      set buf_name "my_BUFG"
    } else {
      puts "$net_name is not in the top level"
      set buf_name $net_hier/my_BUFG
    }
  }
```



```tcl
if {[llength [get_cells -quiet $buf_name]]!=0} {
  puts "Warning - cell name $buf_name already exists."
  set ind 0
  while {[llength [get_cells -quiet $buf_name\_$ind]]!=0} {incr ind}
  set buf_name $buf_name\_$ind
 }
 puts "Creating cell $buf_name (BUFG)"
 create_cell -ref BUFG $buf_name
 set new_net_name $buf_name\_inet
 puts "Creating new $new_net_name"
 create_net $new_net_name
 disconnect_net -net $old_net -objects $opin
 connect_net -net $new_net_name -objects $opin
 connect_net -net $new_net_name -objects [get_pins $buf_name/I]
 connect_net -net $old_net -objects [get_pins $buf_name/O]
 puts "Insert BUFG \"$buf_name\" Successfully!"
}
```

Some Items Should Be Cared

- How to get large fanout net
- How to get timing report through large fanout net
- How to confirm BUFG available