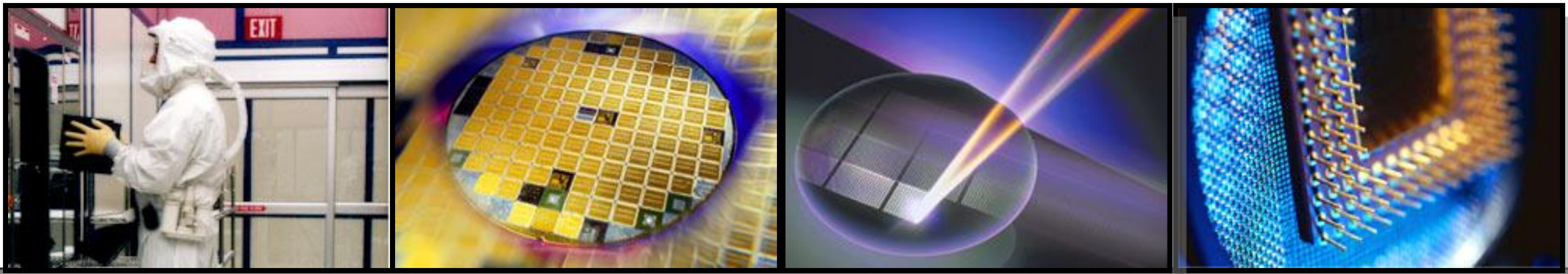
FPGA时序优化方法



课程安排

- 时序收敛流程
- 如何解决FPGA中存在的时序问题
- 通过FPGA设计工具进行时序优化

课程安排

- 时序收敛流程
- 如何解决FPGA中存在的时序问题
- 通过FPGA设计工具进行时序优化

成功的FPGA设计

设计完成后，如何判断一个成功的设计？

- 设计是否满足面积要求---是否能在选定的器件中实现；通常资源占用率不要超过85%。
- 设计是否满足性能要求---能否达到要求的工作频率。
- 管脚定义是否满足要求---信号名、位置、电平标准及数据流方向等。

面积报告

如何判断设计适合所选芯片？

- 所选芯片是否有足够的资源容纳更多的逻辑？如果有，有多少？ Memory资源有多少BITS?
- 如果适合所选芯片，能否完全成功布通?

手段：查看 *Map Report* 或者 *Place & Route Report*

时序收敛

- 在设计的过程中，为了达到时序要求，前端综合与后端的布局布线过程是反复的
- 时序在反复中延时逐渐变小，从而满足设计要求，这一个过程称为时序收敛

    -通过对综合工具设置

    -采用合适的优化技术

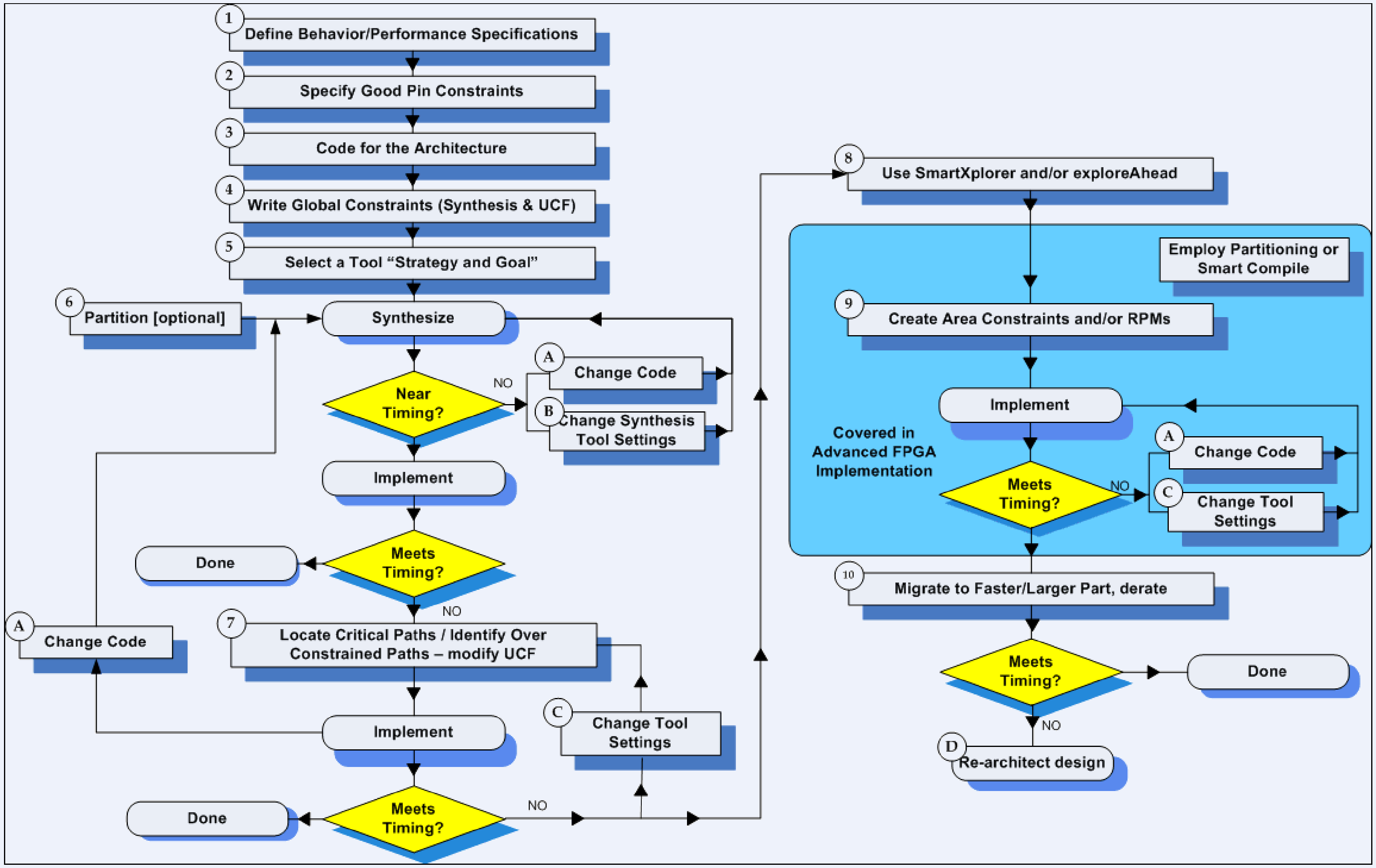
    -修改布局布线

    可以通过上述方式达到时序收敛

时序收敛流程

- **Project Navigator** 产生两种时序报告：
  - ***Post-Map Static Timing Report***
  - ***Post-Place & Route Static Timing Report***
- 时序报告包含没有满足时序要求的详细路径的描述，用于分析判断时序要求没有得到满足的原因。
- **Timing Analyzer**用于建立和阅读时序报告。

时序收敛流程



1 Define Behavior/Performance Specifications
2 Specify Good Pin Constraints
3 Code for the Architecture
4 Write Global Constraints (Synthesis & UCF)
5 Select a Tool “Strategy and Goal”
6 Partition [optional]
Synthesize
Near Timing?
NO
A Change Code
B Change Synthesis Tool Settings
Implement
Meets Timing?
Done
NO
A Change Code
7 Locate Critical Paths / Identify Over Constrained Paths – modify UCF
Implement
C Change Tool Settings
Done
Meets Timing?
NO
8 Use SmartXplorer and/or exploreAhead
Employ Partitioning or Smart Compile
9 Create Area Constraints and/or RPMs
Implement
Covered in Advanced FPGA Implementation
A Change Code
Meets Timing?
NO
C Change Tool Settings
10 Migrate to Faster/Larger Part, derate
Meets Timing?
Done
NO
D Re-architect design

时序收敛流程

性能突破重点在三步:

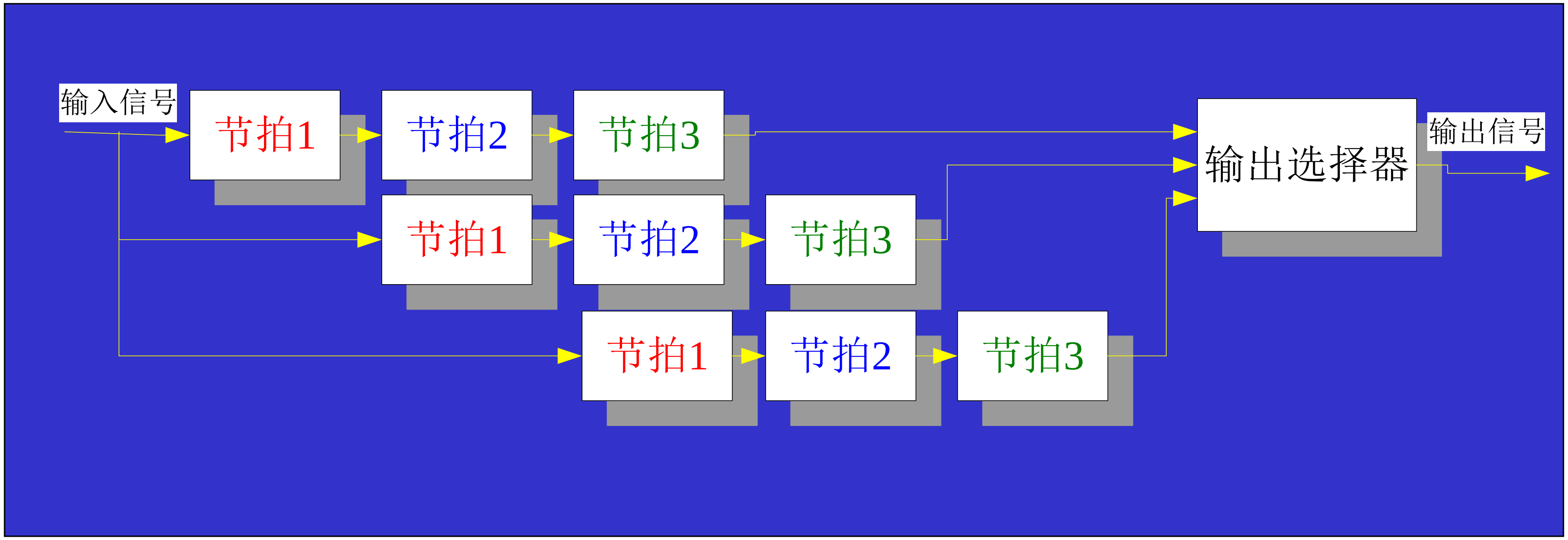
1. 充分利用IP资源
- ✓ DSP48, PowerPC processor, EMAC，SDR/DDR Controller FIFO, block RAM等等。

2. 具有良好的代码风格
- ✓ Use synchronous design methodology
- ✓ Ensure the code is written optimally for critical paths
- ✓ Pipeline

3. 充分利用synthesis工具和Place & Route工具参数选择
- ✓ Try different optimization techniques
- ✓ Add critical timing constraints in synthesis
- ✓ Apply full and correct constraints
- ✓ Use High effort

流水线操作

- 流水线技术几乎是最常用的提供系统工作速率的强有力手段。
- 它是面积换取速度思想的又一种具体体现。

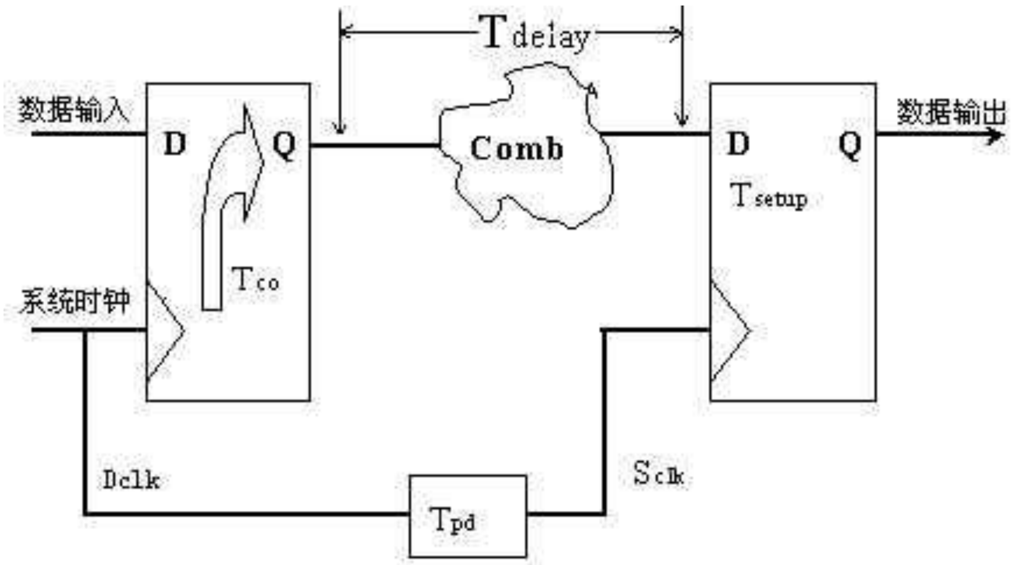


流水线操作

- 考虑一个电路每个时钟周期执行N个操作，工作频率为F。我们可以认为吞吐量为 N*F ops/sec。
- Pipeline本质上是通过增加F来提高吞吐量，达到latency和area的tradeoff。
- 什么决定了最高工作频率呢？
  - 回忆static timing analysis
  - 在flip flops之间的最大延时

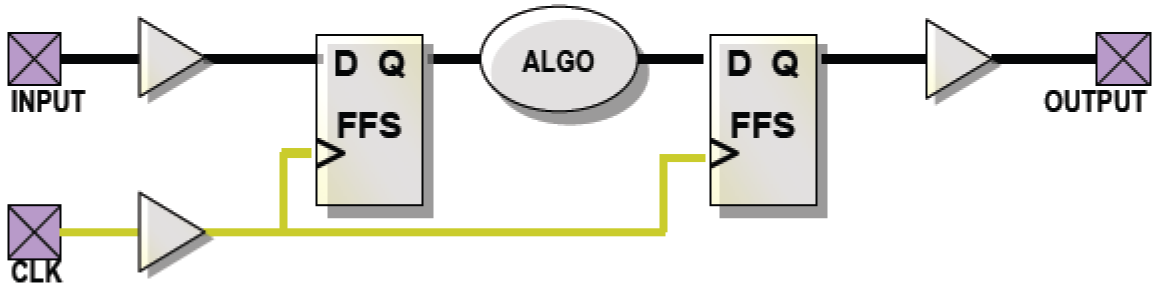
流水线操作（STA）

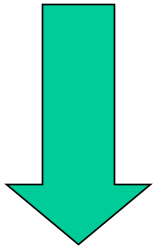
- 时序电路之间的组合电路延时决定整个电路的速度，即最大工作时钟频率，故不可太复杂。



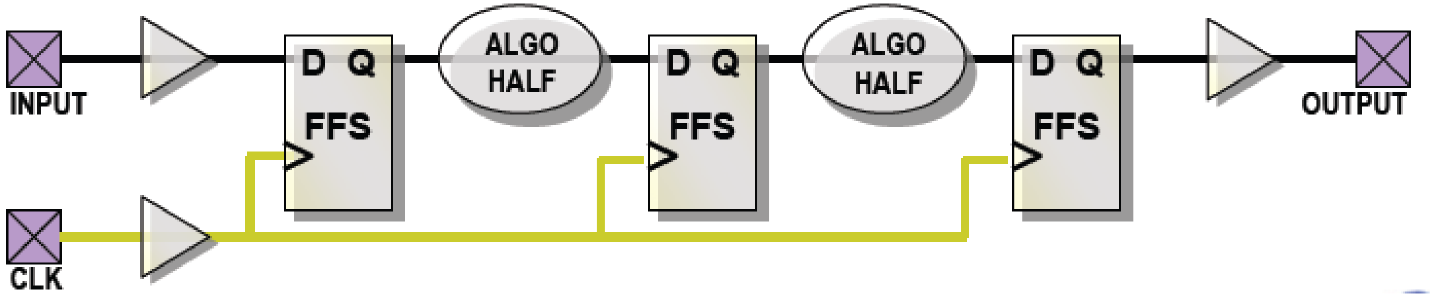
最小时钟周期：T=Tco+Tdelay+Tsetup
最快时钟频率：F= 1/T

解决方式（PIPELINE技术）





通过将Algorithm分为两个部分，在中间插入寄存器

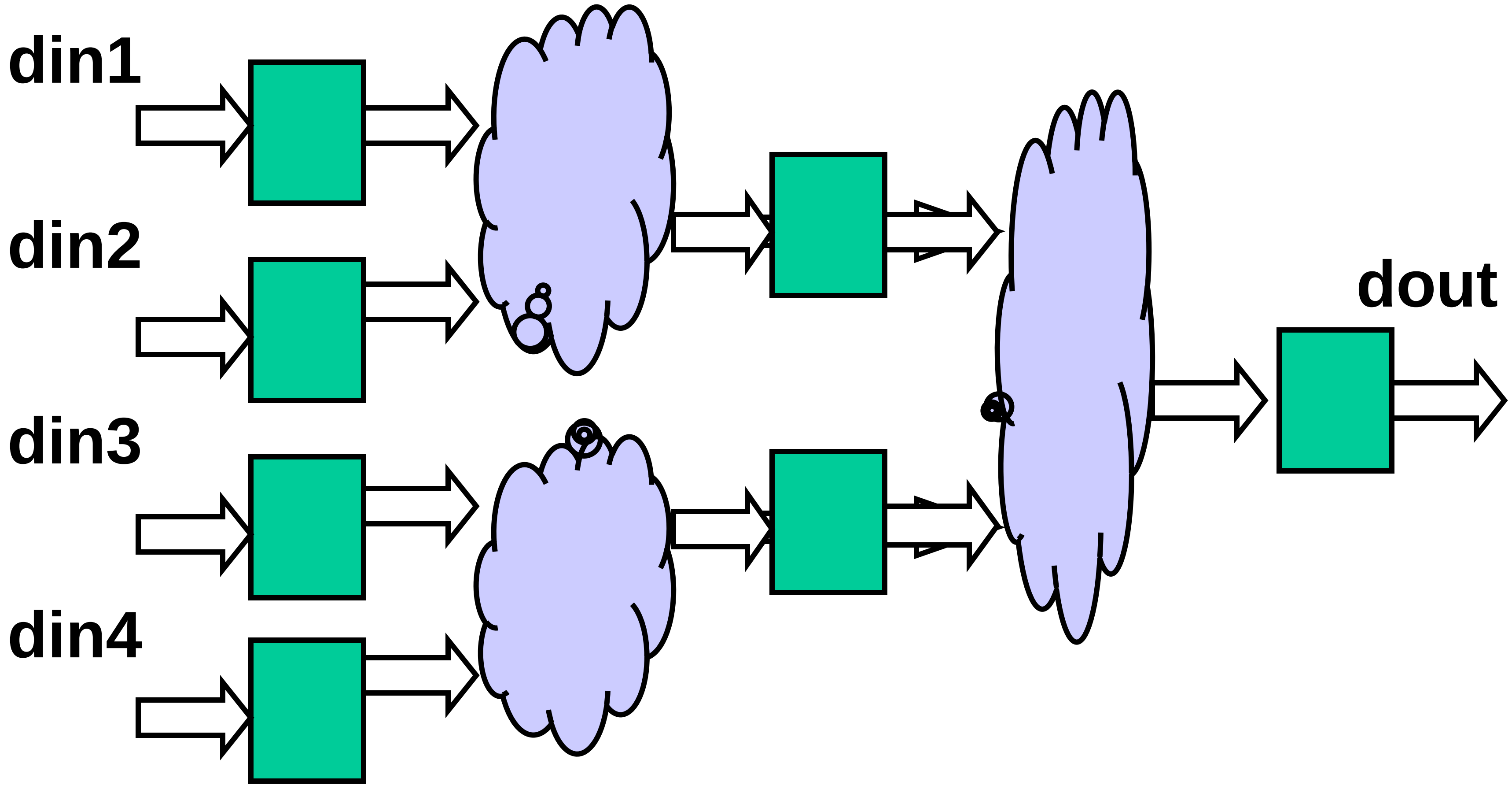


流水线技术

- 其思想是利用寄存器将一条长路径切分成几段小路径，从而达到提高工作速率的作用。
- 假设原路径延时为t，加入2级流水线并且假设路径切割均匀，则路径延时可以减少到约t/3，从而系统速率可以提高到原来的3倍左右。
- 当然要注意的是输出同时会往后推迟3个时钟周期。所以采用流水线技术时，要记得进行时序调整。

流水线设计例子

- 设计din1 + din2 + din3 + din4结果输出给dout。



PIPELINE程序实例

- 未用PIPELINE技术前的程序：

```
Always @(posedge clk)
     if((a+b+c) == d)
     ………..
```

- 采用PIPELINE技术后的程序：

```
always @(posedge clk)
begin
      e <= a+b+c;
      if(e ==d)
      ………….. 
end
```

编码注意事项

- Use pipeline stages－more bandwidth
- Use synchronous reset－better system control
- Use inferable resources
  - Multiplexer
  - Shift Register LUT (SRL)
  - Block RAM, LUT RAM
  - Cascade DSP
- Avoid high-level constructs (loops, for example) in code
  - Many synthesis tool produce slow implementations

面向综合的RTL开发

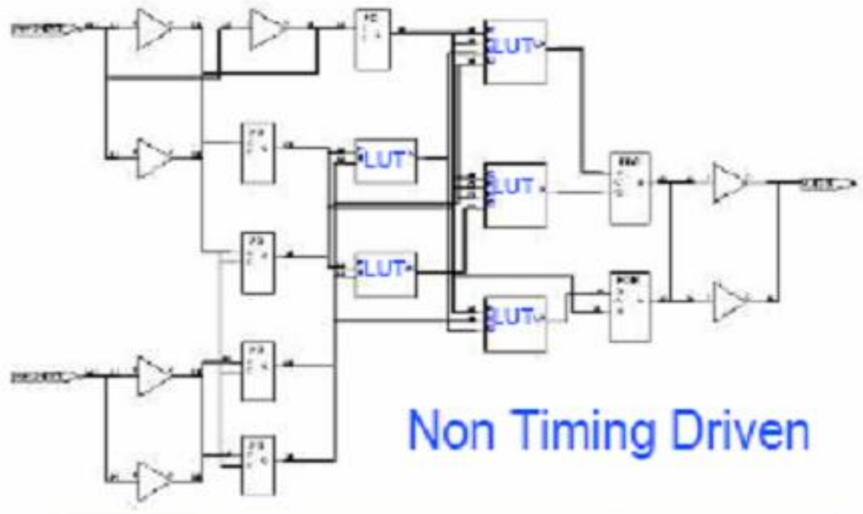
- 敏感信号表应尽可能完整。当敏感信号表不完整时，综合前后的网表仿真可能会不对应。在组合逻辑块中，敏感信号表一定要包含这个过程或块读入的每个信号。对于时序块，敏感信号表一定要包含时钟和其他控制信号。另外，也要避免敏感信号表中出现不必要的信号，因为它们会降低仿真速度。
- 在Verilog中，对于时钟块应使用非阻塞赋值，否则，RTL级和门级的行为会不同。
- RTL代码应该以一定的方式划分，使综合过程更有效，时序要求更容易实现。例如，编写一个状态机，代码可以分为两个过程——组合逻辑和时序逻辑。
- 相关的组合逻辑应该在同一模块中；不同综合目标的模块应该分开；将相关逻辑功能分在一组也能避免块之间的时序依赖关系。
- 应该避免在顶层设计中调用门级逻辑。顶层逻辑阻止工具优化带有块逻辑的门级逻辑。如果在SOC中使用了多个相同的核。胶联逻辑应该放在核内部。例如，时钟域缓冲器和总线接口门可以放在核内，使胶联逻辑变成简单的互连线。

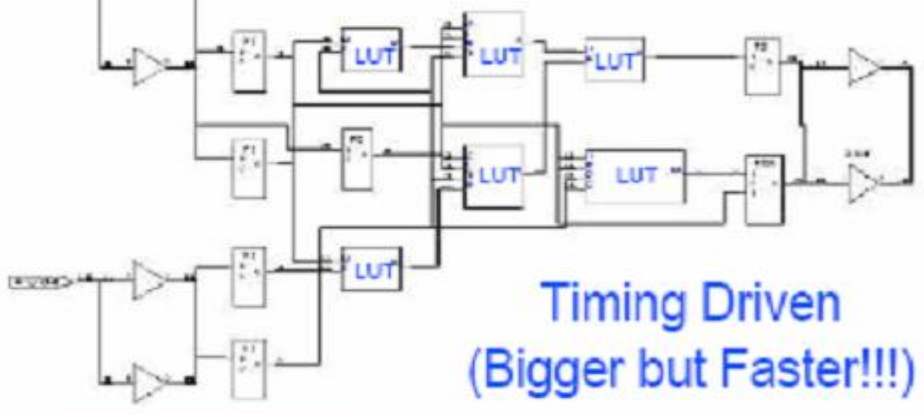
Synthesis guidelines

- Use timing constraints
- Define tight but realistic individual clock constraints
- Put unrelated clocks into different clock groups
- Use proper options and attributes
- Turn off resource sharing
- Move flip-flops from IOBs closer to logic
- Turn on FSM optimization
- Use the *retiming* option

约束的影响

Non timing-constrained designs can be optimized for area rather than performance





Total LUTs: 5
Clock Freq: 423.7 MHz

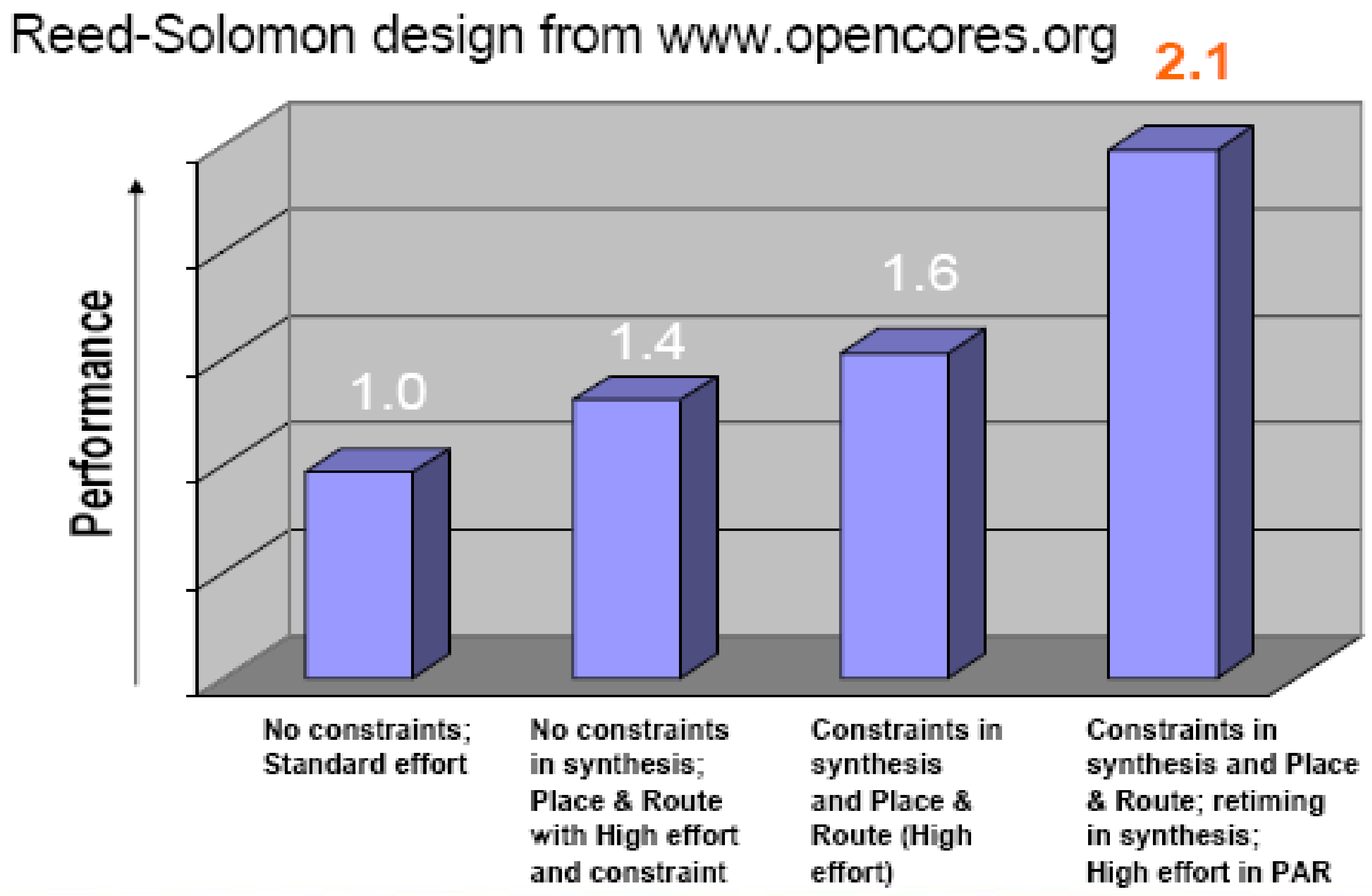
Total LUTs: 6
Clock Freq: 591.7 MHz (+ 40%)

Place & Route Guidelines

- Timing constraints
  - Use tight, realistic constraints
- Recommended options
  - High-effort Place & Route
  - By default, effort is set to *Standard*
  - Timing-driven MAP
- Tools to help meet timing
  - Floorplanning(Use the PACE and PlanAhead software tools)
  - Physical synthesis tools
- Other available options:
  - Incremental design

Using the correct Place & Route options can have a dramatic impact on design performance

工具中Constraints的影响



了解FPGA特性，正确设定目标

- 每种型号的FPGA都有性能极限，工艺不同，性能也不同
  - 峰值频率（仅 1 logic level）
    - Spartan-6 (400 MHz)
    - Virtex-6 (650 MHz)
    - Virtex-5 (550 MHz)
    - 以上频率是非常理想的情况，布线延迟很短
  - 那么典型值是多少呢?
    - 要求良好的 timing estimate

性能预估

- Performance estimates are available before implementation is complete
- Synthesis Report
  - Logic delays are accurate
  - Routing delays are estimated based on fanout
  - Reported performance is generally accurate to within 30 percent
- Post-Map Static Timing Report
  - Logic delays are accurate
  - Routing delays are estimated based on placement and fanout

性能预估

- Synthesis tools have access to logic delays, but not net delays
  - To resolve this, synthesis tools use a loading model as a net delay estimate
    - Up to 50% uncertainty
    - Xilinx still recommends that you review your synthesis tools timing estimate
- Experienced FPGA designers know that another estimate is to u se the 50/50 rule
  - This assumes that your logic) will typically equal an average net delay
    - From the Virtex-6 data sheet (using the -3 speed grade, fastest device)
      - Tlogic = .77ns and Troute ~ .77ns for an estimate of 1.54ns for 1 logic level (this corresponds to the 650 MHz estimate)
      - Likewise, 2 logic level ~3.08ns and 3 logic level ~ 4.62ns

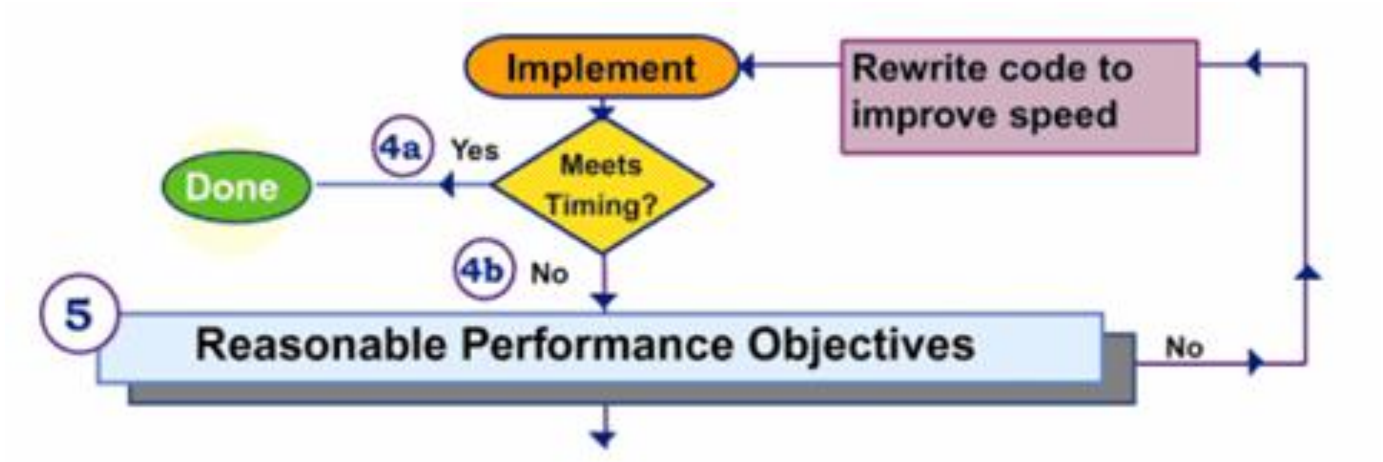
性能预估

- 请记住…
  - A logic level is a clock-to-out on a CLB register, plus a LUT del ay, plus a setup time on a CLB register
    - Tckdi + Tilo + Tsu
  - Your performance is greatly impacted by the number of logic lev els
    - FPGA experts know if you want to improve your system spe ed, first make sure you have evaluated the number of logic l evels on your timing **critical path**

Performance Estimates

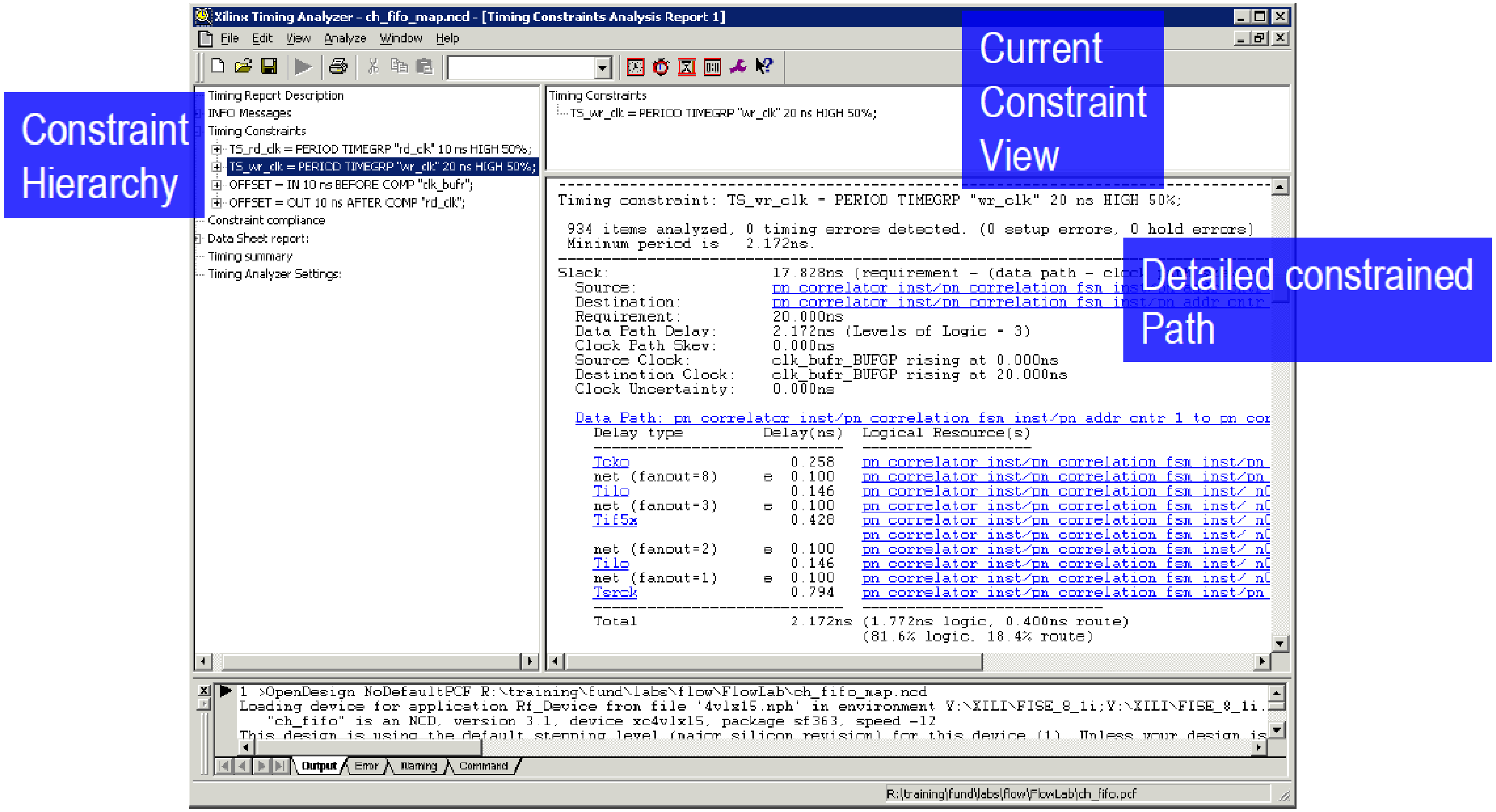
- 经验缺乏的工程师typically design their worst case path at 3 to 5 logic levels
  - Worst case path ~ 7.7ns = 129 MHz
- 具有经验的工程师typically design their worst case path at 2 logic levels
  - Worst case path ~ 4.62 = 324 MHz
  - But this will depend on the effort you put in to follow good HDL coding techniques and optimize your design for your FPGA architecture
    - Replicating logic to reduce high fanout net delays
    - Pipeline to reduce logic levels
    - Using alternative design techniques to reduce logic levels
    - Using your synthesis options to reduce logic levels
    - Using advanced implementation options to improve the place and route solution

静态时序分析


Implement
Rewrite code to improve speed
4a
Yes
Done
Meets Timing?
4b
No
5
Reasonable Performance Objectives
No

- Post-map：Map后，使用Post-map timing report确定关键路径的逻辑延迟
- Post-PAR：PAR后，使用Post-PAR static timing report确定时序约束是否满足
- Logic delay Vs. Routing delay：60%/40%原则
- Timing Analyzer可以读取时序报告，查找关键路径，并与Floorplanner协同解决时序问题

使用Timing Analyzer 查看时序报告



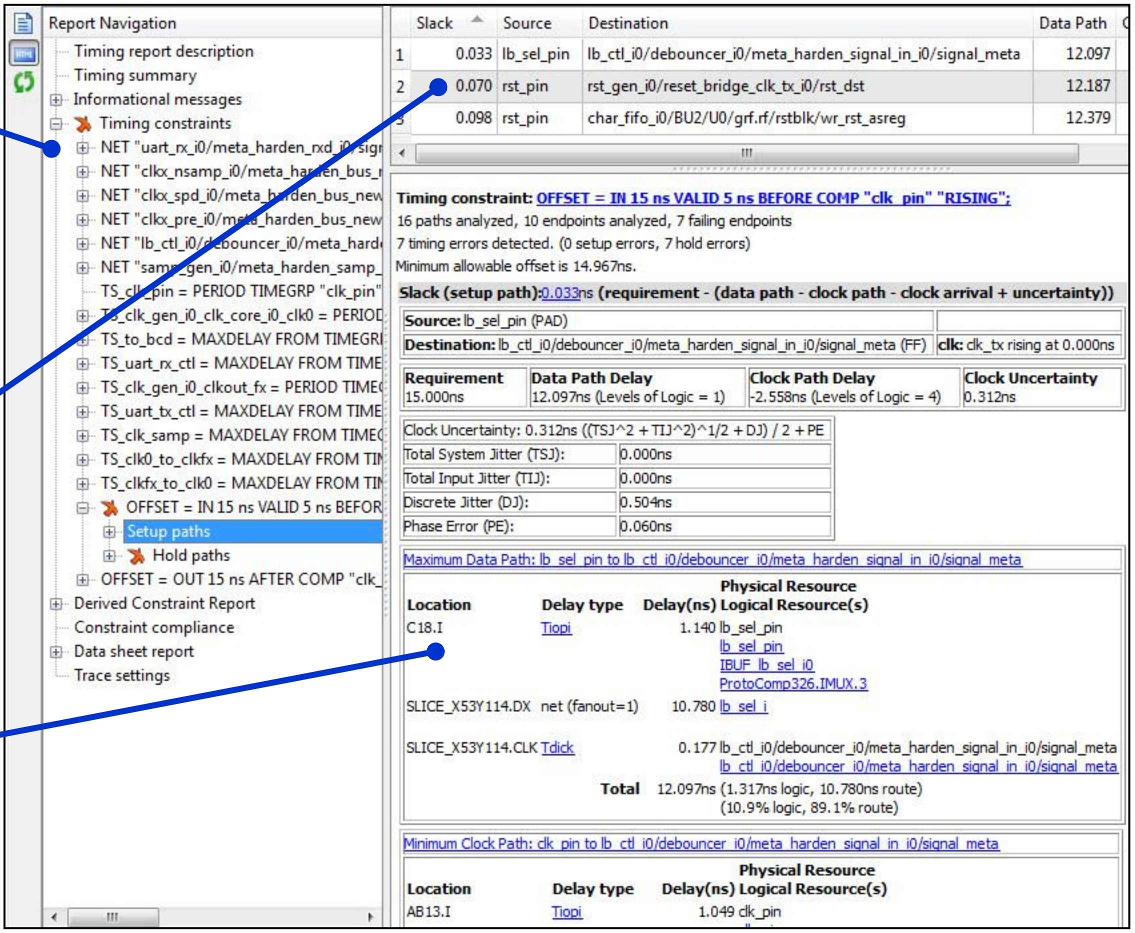
Xilinx Timing Analyzer - ch_fifo_map.ncd - [Timing Constraints Analysis Report 1]
File Edit View Analyze Window Help
Constraint Hierarchy
Current Constraint View
Detailed constrained Path
Timing Report Description
INFO Messages
Timing Constraints
TS_rd_clk = PERIOD TIMEGRP "rd_clk" 10 ns HIGH 50%;
TS_wr_clk = PERIOD TIMEGRP "wr_clk" 20 ns HIGH 50%;
OFFSET = IN 10 ns BEFORE COMP "clk_bufr";
OFFSET = OUT 10 ns AFTER COMP "rd_clk";
Constraint compliance
Data Sheet report:
Timing summary
Timing Analyzer Settings:
Timing Constraints
TS_wr_clk = PERIOD TIMEGRP "wr_clk" 20 ns HIGH 50%;
Output
Error
Warning
Command
R:\training\fund\labs\flow\FlowLab\ch_fifo.pcf

```
--------------------------------------------------------------------------------
Timing constraint: TS_wr_clk = PERIOD TIMEGRP "wr_clk" 20 ns HIGH 50%;

 934 items analyzed, 0 timing errors detected. (0 setup errors, 0 hold errors)
 Minimum period is   2.172ns.
--------------------------------------------------------------------------------
Slack:                  17.828ns (requirement - (data path - cl
  Source:               pn correlator inst/pn correlation fsm i
  Destination:          pn correlator inst/pn correlation fsm inst/pn addr cntr
  Requirement:          20.000ns
  Data Path Delay:      2.172ns (Levels of Logic = 3)
  Clock Path Skew:      0.000ns
  Source Clock:         clk_bufr_BUFGP rising at 0.000ns
  Destination Clock:    clk_bufr_BUFGP rising at 20.000ns
  Clock Uncertainty:    0.000ns

  Data Path: pn correlator inst/pn correlation fsm inst/pn addr cntr 1 to pn cor
    Delay type         Delay(ns)  Logical Resource(s)
    ----------------------------  -------------------
    Tcko                  0.258   pn correlator inst/pn correlation fsm inst/pn
    net (fanout=8)     e  0.100   pn correlator inst/pn correlation fsm inst/pn
    Tilo                  0.146   pn correlator inst/pn correlation fsm inst/ n
    net (fanout=3)     e  0.100   pn correlator inst/pn correlation fsm inst/ n
    Tif5x                 0.428   pn correlator inst/pn correlation fsm inst/ n
                                  pn correlator inst/pn correlation fsm inst/ n
    net (fanout=2)     e  0.100   pn correlator inst/pn correlation fsm inst/ n
    Tilo                  0.146   pn correlator inst/pn correlation fsm inst/ n
    net (fanout=1)     e  0.100   pn correlator inst/pn correlation fsm inst/ n
    Tcrck                 0.794   pn correlator inst/pn correlation fsm inst/pn
    ----------------------------  ---------------------------
    Total                 2.172ns (1.772ns logic, 0.400ns route)
                                  (81.6% logic, 18.4% route)
```

```
1 >OpenDesign NoDefaultPCF R:\training\fund\labs\flow\FlowLab\ch_fifo_map.ncd
Loading device for application Rf_Device from file '4vlx15.nph' in environment V:\XILI\FISE_8_1i;V:\XILI\FISE_8_1i.
   "ch_fifo" is an NCD, version 3.1, device xc4vlx15, package sf363, speed -12
This design is using the default stepping level (major silicon revision) for this device (1). Unless your design is
```

Timing Analyzer GUI

- Hierarchical browser
  - Quickly navigate to specific report sections
  - Failing constraints indicated with a red “X”
- Timing objects window
  - Summarizes the path displayed in the path detail window
- Report text
  - Logic highlighted in blue can be cross-probed



Report Navigation
- Timing report description
- Timing summary
- Informational messages
- Timing constraints
  - NET "uart_rx_i0/meta_harden_rxd_i0/sigr
  - NET "clkx_nsamp_i0/meta_harden_bus_r
  - NET "clkx_spd_i0/meta_harden_bus_new
  - NET "clkx_pre_i0/meta_harden_bus_new
  - NET "lb_ctl_i0/debouncer_i0/meta_hard
  - NET "samp_gen_i0/meta_harden_samp_
  - TS_clk_pin = PERIOD TIMEGRP "clk_pin"
  - TS_clk_gen_i0_clk_core_i0_clk0 = PERIOD
  - TS_to_bcd = MAXDELAY FROM TIMEGRI
  - TS_uart_rx_ctl = MAXDELAY FROM TIME
  - TS_clk_gen_i0_clkout_fx = PERIOD TIME(
  - TS_uart_tx_ctl = MAXDELAY FROM TIME
  - TS_clk_samp = MAXDELAY FROM TIME(
  - TS_clk0_to_clkfx = MAXDELAY FROM TIN
  - TS_clkfx_to_clk0 = MAXDELAY FROM TIN
  - OFFSET = IN 15 ns VALID 5 ns BEFOR
    - Setup paths
    - Hold paths
  - OFFSET = OUT 15 ns AFTER COMP "clk_
- Derived Constraint Report
- Constraint compliance
- Data sheet report
- Trace settings

	Slack	Source	Destination	Data Path
1	0.033	lb_sel_pin	lb_ctl_i0/debouncer_i0/meta_harden_signal_in_i0/signal_meta	12.097
2	0.070	rst_pin	rst_gen_i0/reset_bridge_clk_tx_i0/rst_dst	12.187
3	0.098	rst_pin	char_fifo_i0/BU2/U0/grf.rf/rstblk/wr_rst_asreg	12.379

Timing constraint: OFFSET = IN 15 ns VALID 5 ns BEFORE COMP "clk_pin" "RISING";
16 paths analyzed, 10 endpoints analyzed, 7 failing endpoints
7 timing errors detected. (0 setup errors, 7 hold errors)
Minimum allowable offset is 14.967ns.

Slack (setup path):0.033ns **(requirement - (data path - clock path - clock arrival + uncertainty))**

Source: lb_sel_pin (PAD)	
Destination: lb_ctl_i0/debouncer_i0/meta_harden_signal_in_i0/signal_meta (FF)	**clk:** clk_tx rising at 0.000ns

Requirement	Data Path Delay	Clock Path Delay	Clock Uncertainty
15.000ns	12.097ns (Levels of Logic = 1)	-2.558ns (Levels of Logic = 4)	0.312ns

Clock Uncertainty: 0.312ns ((TSJ^2 + TIJ^2)^1/2 + DJ) / 2 + PE	
Total System Jitter (TSJ):	0.000ns
Total Input Jitter (TIJ):	0.000ns
Discrete Jitter (DJ):	0.504ns
Phase Error (PE):	0.060ns

Maximum Data Path: lb_sel_pin to lb_ctl_i0/debouncer_i0/meta_harden_signal_in_i0/signal_meta

Location	Delay type	Delay(ns)	Physical Resource Logical Resource(s)
C18.I	Tiopi	1.140	lb_sel_pin lb_sel_pin IBUF_lb_sel_i0 ProtoComp326.IMUX.3
SLICE_X53Y114.DX	net (fanout=1)	10.780	lb_sel_i
SLICE_X53Y114.CLK	Tdick	0.177	lb_ctl_i0/debouncer_i0/meta_harden_signal_in_i0/signal_meta lb_ctl_i0/debouncer_i0/meta_harden_signal_in_i0/signal_meta
	Total	12.097ns	(1.317ns logic, 10.780ns route) (10.9% logic, 89.1% route)

Minimum Clock Path: clk_pin to lb_ctl_i0/debouncer_i0/meta_harden_signal_in_i0/signal_meta

Location	Delay type	Delay(ns)	Physical Resource Logical Resource(s)
AB13.I	Tiopi	1.049	clk_pin

Timing Reports

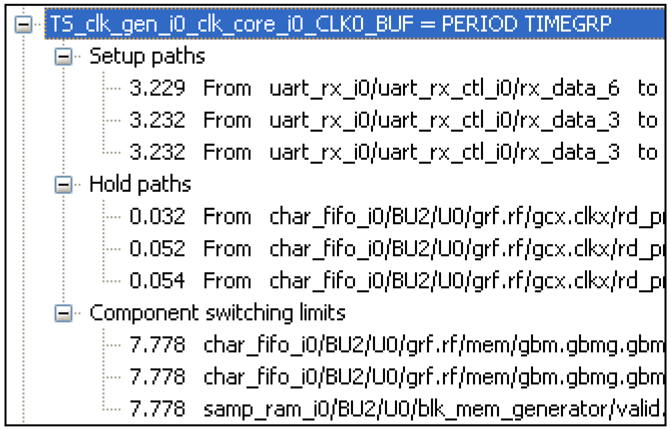
- Timing reports help you determine why your design fails to meet its constraints
  - Reports contain detailed descriptions of paths that fail their constraints
- The implementation tools can create timing reports at two points in t he design flow
  - Post-Map Static Timing Report
    - Use for an early indication as to whether your design might meet timing
  - Post-Place & Route Static Timing Report
    - Use as a final analysis of whether your design has met timing
- The Timing Analyzer is a utility for creating and reading timing repo rts

Timing Report Structure

- Timing constraints
  - Number of paths covered and number of paths that failed for each const raint
  - Detailed descriptions of the longest paths
- Data sheet report
  - Setup, hold, and clock-to-out times for each I/O pin
- Timing summary
  - Timing errors (number of failing paths)
  - Timing score (total number of ps of all constraints that were missed)
- Timing report description
  - Allows you to easily duplicate the report

Paths Reported

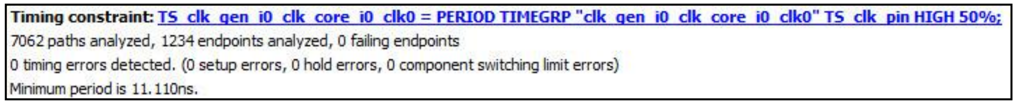
- Setup paths
  - Slowest delay paths for each constraint
  - Defaults to the three longest paths
- Hold paths
  - Fastest delay paths for each constraint
- Component switching limits
  - Checks that the toggle rate and duty cycle are in limits with specifi cation



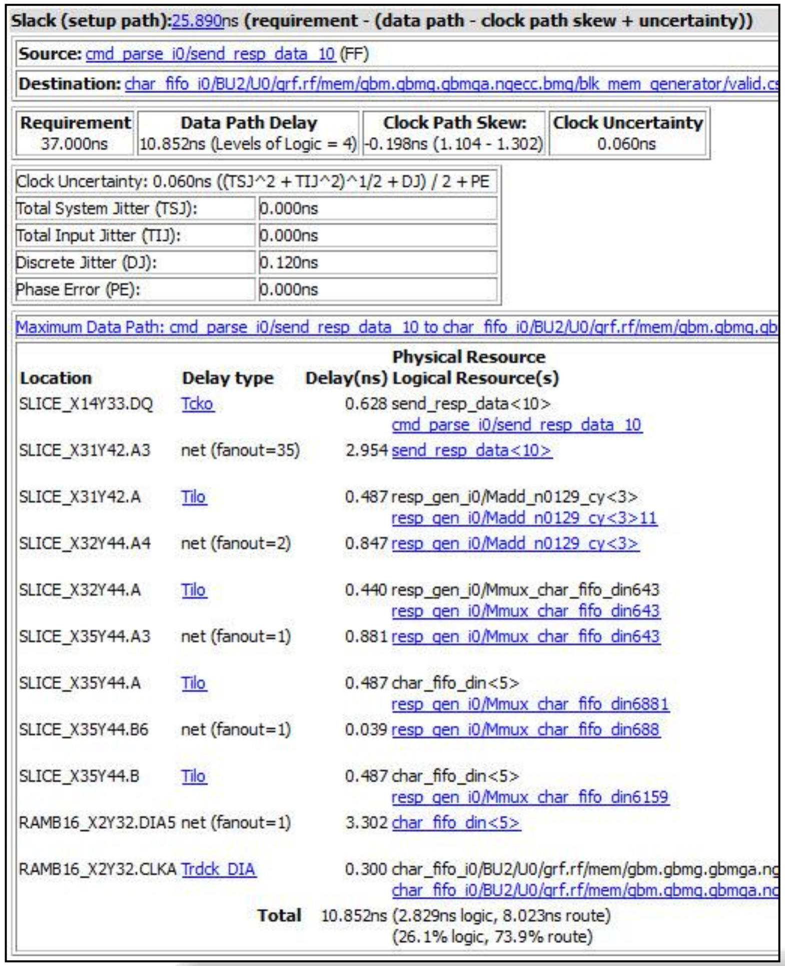
```
TS_clk_gen_i0_clk_core_i0_CLK0_BUF = PERIOD TIMEGRP
  Setup paths
    3.229  From  uart_rx_i0/uart_rx_ctl_i0/rx_data_6  to
    3.232  From  uart_rx_i0/uart_rx_ctl_i0/rx_data_3  to
    3.232  From  uart_rx_i0/uart_rx_ctl_i0/rx_data_3  to
  Hold paths
    0.032  From  char_fifo_i0/BU2/U0/grf.rf/gcx.clkx/rd_p
    0.052  From  char_fifo_i0/BU2/U0/grf.rf/gcx.clkx/rd_p
    0.054  From  char_fifo_i0/BU2/U0/grf.rf/gcx.clkx/rd_p
  Component switching limits
    7.778  char_fifo_i0/BU2/U0/grf.rf/mem/gbm.gbmg.gbm
    7.778  char_fifo_i0/BU2/U0/grf.rf/mem/gbm.gbmg.gbm
    7.778  samp_ram_i0/BU2/U0/blk_mem_generator/valid.
```

Report Example

- Constraint summary
  - Number of paths analyzed
  - Number of timing errors
  - Length of critical path
- Total delay
  - Clock and data breakdown
- Clock jitter analysis
- Detailed path description
  - Delay types are described i n the data sheet
  - Worst-case conditions are a ssumed, unless pro-rated



```
Timing constraint: TS clk gen i0 clk core i0 clk0 = PERIOD TIMEGRP "clk gen i0 clk core i0 clk0" TS clk pin HIGH 50%;
7062 paths analyzed, 1234 endpoints analyzed, 0 failing endpoints
0 timing errors detected. (0 setup errors, 0 hold errors, 0 component switching limit errors)
Minimum period is 11.110ns.
```



```
Slack (setup path):25.890ns (requirement - (data path - clock path skew + uncertainty))
Source: cmd parse i0/send resp data 10 (FF)
Destination: char fifo i0/BU2/U0/grf.rf/mem/gbm.gbmg.gbmga.ngecc.bmg/blk mem generator/valid.cs
```

Requirement	Data Path Delay	Clock Path Skew:	Clock Uncertainty
37.000ns	10.852ns (Levels of Logic = 4)	-0.198ns (1.104 - 1.302)	0.060ns

Clock Uncertainty: 0.060ns ((TSJ^2 + TIJ^2)^1/2 + DJ) / 2 + PE	
Total System Jitter (TSJ):	0.000ns
Total Input Jitter (TIJ):	0.000ns
Discrete Jitter (DJ):	0.120ns
Phase Error (PE):	0.000ns

```
Maximum Data Path: cmd parse i0/send resp data 10 to char fifo i0/BU2/U0/grf.rf/mem/gbm.gbmg.gb

                                             Physical Resource
Location            Delay type       Delay(ns) Logical Resource(s)
SLICE_X14Y33.DQ     Tcko                 0.628 send_resp_data<10>
                                               cmd parse i0/send resp data 10
SLICE_X31Y42.A3     net (fanout=35)      2.954 send resp data<10>

SLICE_X31Y42.A      Tilo                 0.487 resp_gen_i0/Madd_n0129_cy<3>
                                               resp gen i0/Madd n0129 cy<3>11
SLICE_X32Y44.A4     net (fanout=2)       0.847 resp gen i0/Madd n0129 cy<3>

SLICE_X32Y44.A      Tilo                 0.440 resp_gen_i0/Mmux_char_fifo_din643
                                               resp gen i0/Mmux char fifo din643
SLICE_X35Y44.A3     net (fanout=1)       0.881 resp gen i0/Mmux char fifo din643

SLICE_X35Y44.A      Tilo                 0.487 char_fifo_din<5>
                                               resp gen i0/Mmux char fifo din6881
SLICE_X35Y44.B6     net (fanout=1)       0.039 resp gen i0/Mmux char fifo din688

SLICE_X35Y44.B      Tilo                 0.487 char_fifo_din<5>
                                               resp gen i0/Mmux char fifo din6159
RAMB16_X2Y32.DIA5 net (fanout=1)         3.302 char fifo din<5>

RAMB16_X2Y32.CLKA Trdck DIA              0.300 char_fifo_i0/BU2/U0/grf.rf/mem/gbm.gbmg.gbmga.ng
                                               char fifo i0/BU2/U0/grf.rf/mem/gbm.gbmg.gbmga.nc
                                   Total  10.852ns (2.829ns logic, 8.023ns route)
                                               (26.1% logic, 73.9% route)
```

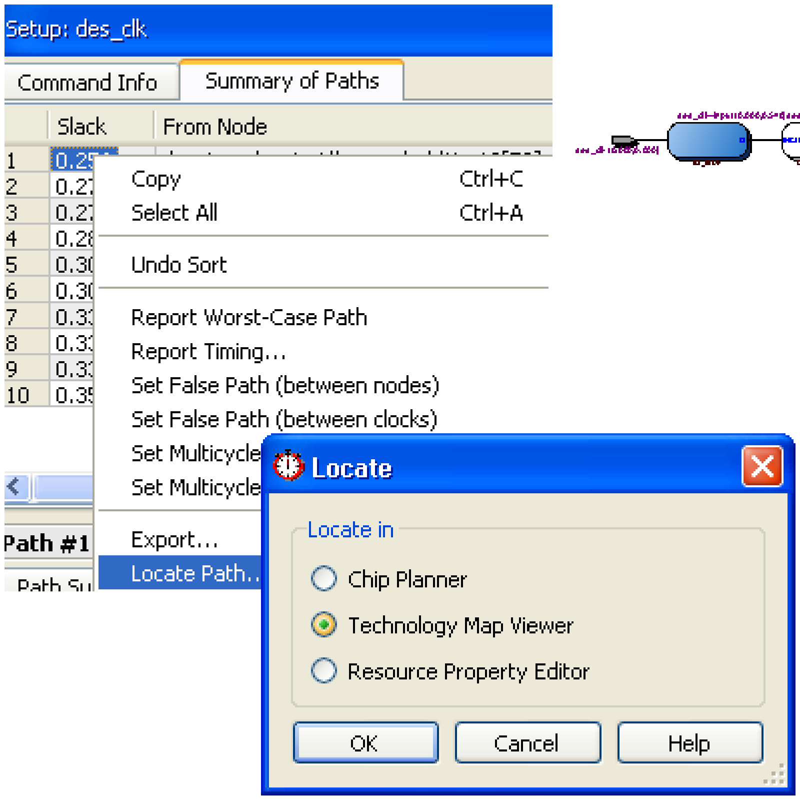
Quartus II 软件寻找时序不满足

- Technology Map Viewer
  - Graphically shows number of logic levels

- Chip Planner
  - Graphically shows placement

- TimeQuest path analysis
  - Highlights clock/path delays
  - Highlights fan-out
  - Highlights number of logic levels
  - And just about everything else

Technology Map Viewer



Setup: des_clk

Command Info | Summary of Paths

	Slack	From Node
1	0.25	
2	0.27	
3	0.27	
4	0.28	
5	0.30	
6	0.30	
7	0.33	
8	0.33	
9	0.33	
10	0.35	

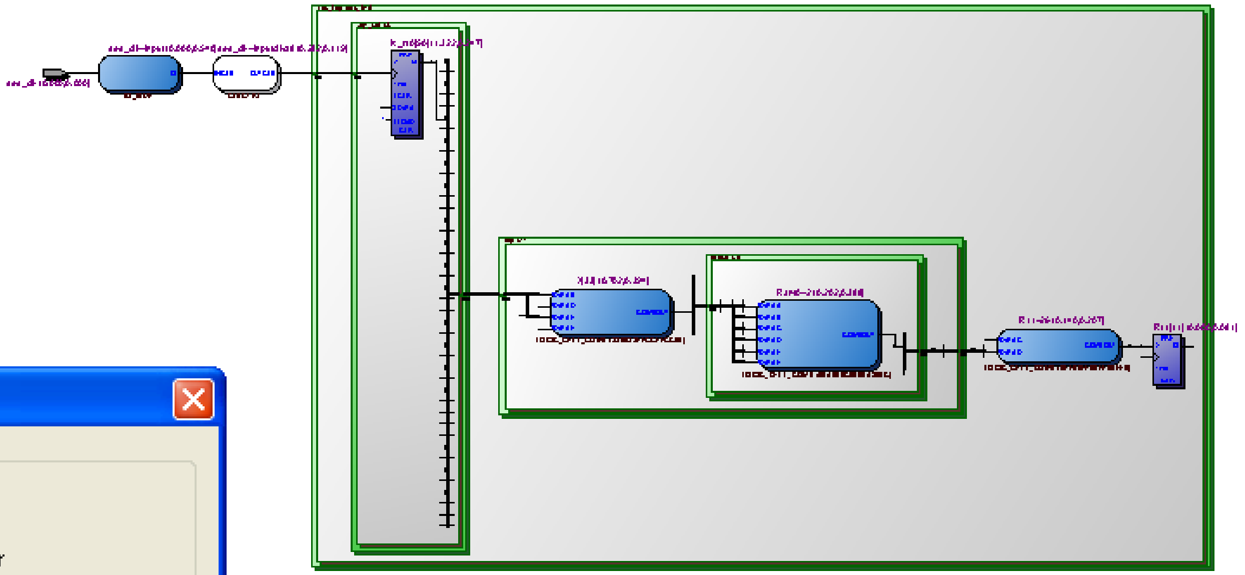
Path #1

Path Su

Locate

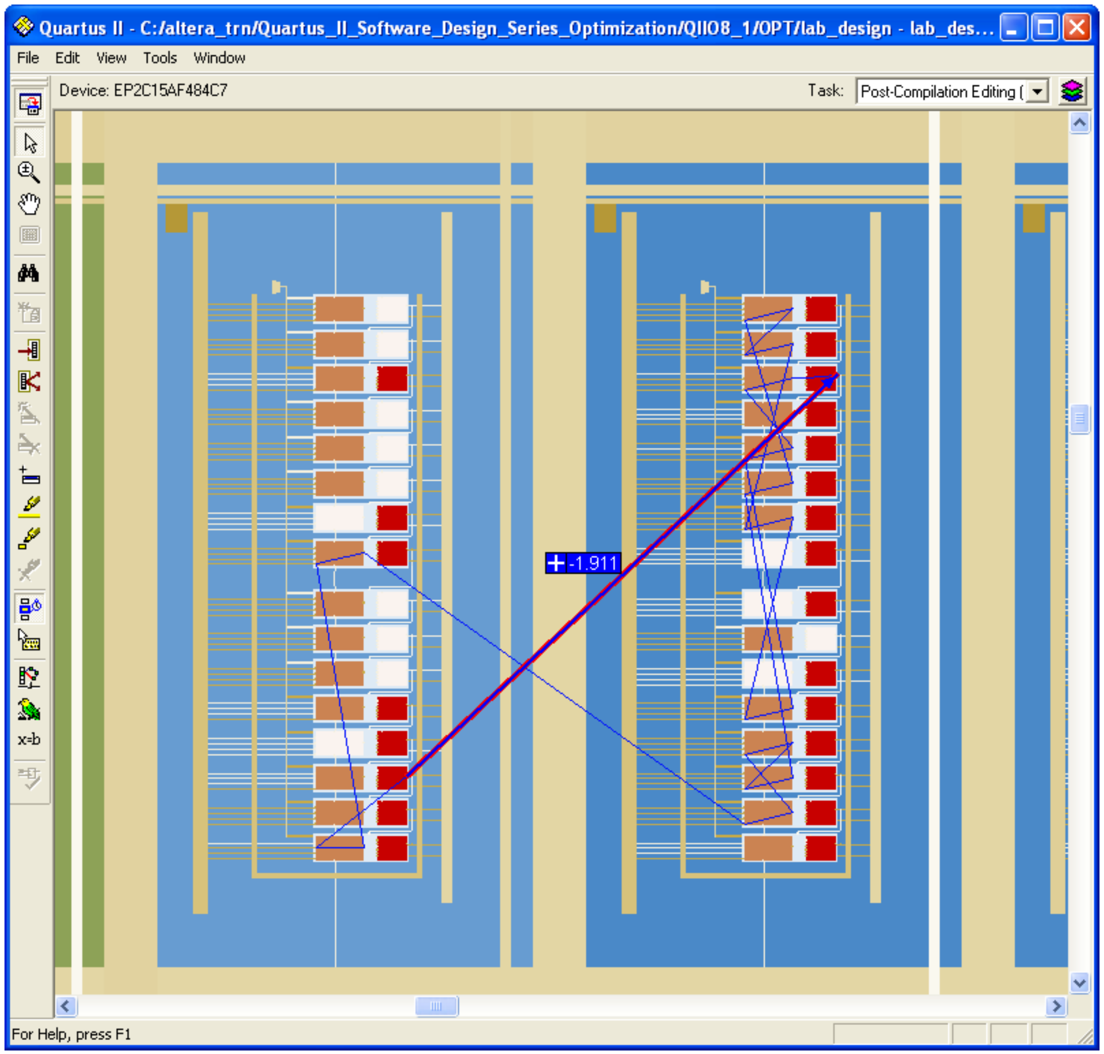
Locate in

( ) Chip Planner
[✓ Technology Map Viewer]
( ) Resource Property Editor

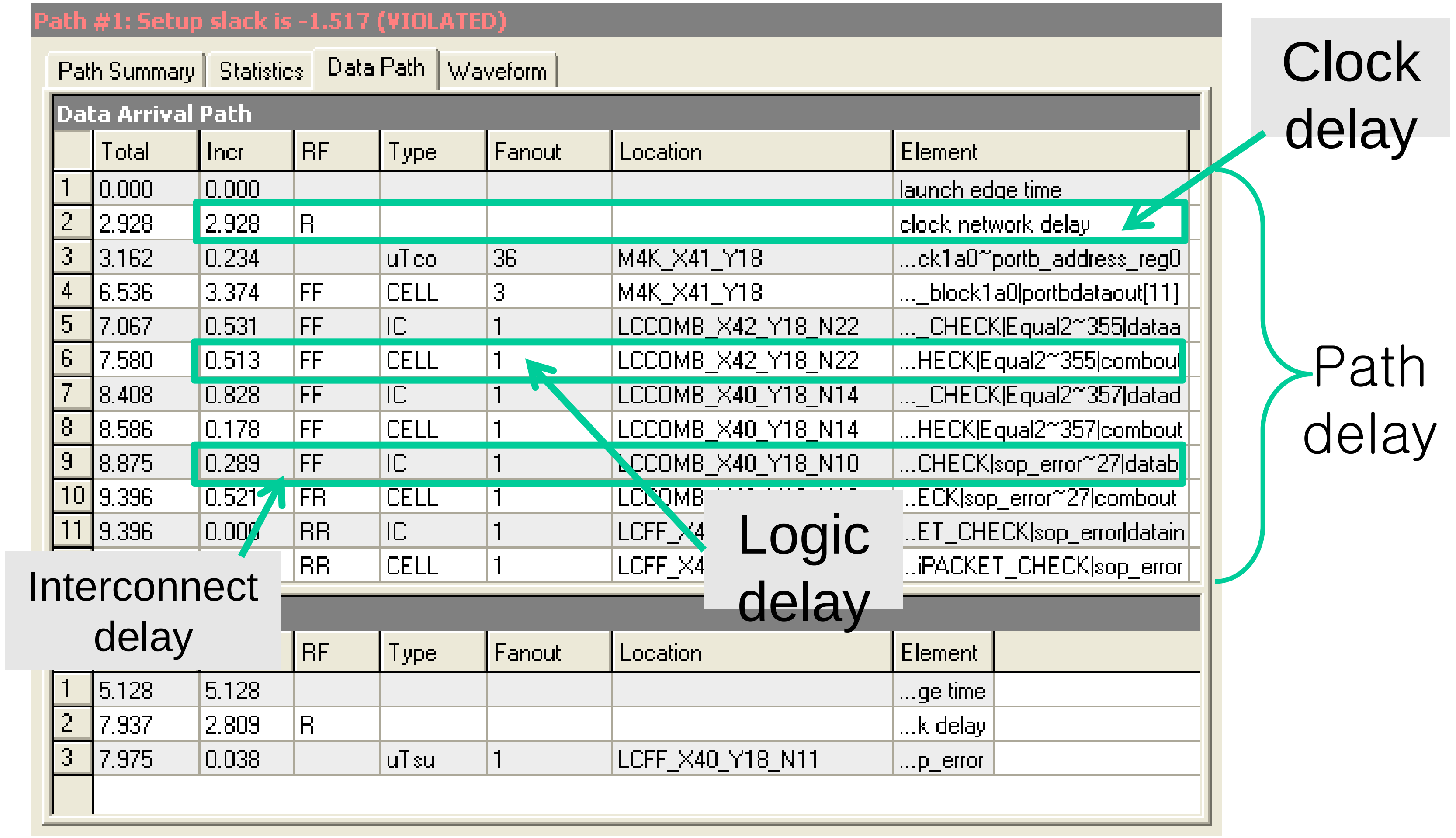


Chip Planner

Displays placement, routing & timing information for selected path


Quartus II - C:/altera_trn/Quartus_II_Software_Design_Series_Optimization/QII08_1/OPT/lab_design - lab_des...
File Edit View Tools Window
Device: EP2C15AF484C7
Task: Post-Compilation Editing (
-1.911
For Help, press F1

TimeQuest Path Analysis



Path #1: Setup slack is -1.517 (VIOLATED)

Path Summary | Statistics | Data Path | Waveform

Data Arrival Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	2.928	2.928	R				clock network delay
3	3.162	0.234		uTco	36	M4K_X41_Y18	...ck1a0~portb_address_reg0
4	6.536	3.374	FF	CELL	3	M4K_X41_Y18	..._block1a0|portbdataout[11]
5	7.067	0.531	FF	IC	1	LCCOMB_X42_Y18_N22	..._CHECK|Equal2~355|dataa
6	7.580	0.513	FF	CELL	1	LCCOMB_X42_Y18_N22	...HECK|Equal2~355|combout
7	8.408	0.828	FF	IC	1	LCCOMB_X40_Y18_N14	..._CHECK|Equal2~357|datad
8	8.586	0.178	FF	CELL	1	LCCOMB_X40_Y18_N14	...HECK|Equal2~357|combout
9	8.875	0.289	FF	IC	1	LCCOMB_X40_Y18_N10	...CHECK|sop_error~27|datab
10	9.396	0.521	FR	CELL	1	LCCOMB [illegible]	..ECK|sop_error~27|combout
11	9.396	0.00[illegible]	RR	IC	1	LCFF_X4[illegible]	..ET_CHECK|sop_error|datain
[illegible]	[illegible]	[illegible]	RR	CELL	1	LCFF_X4[illegible]	..iPACKET_CHECK|sop_error

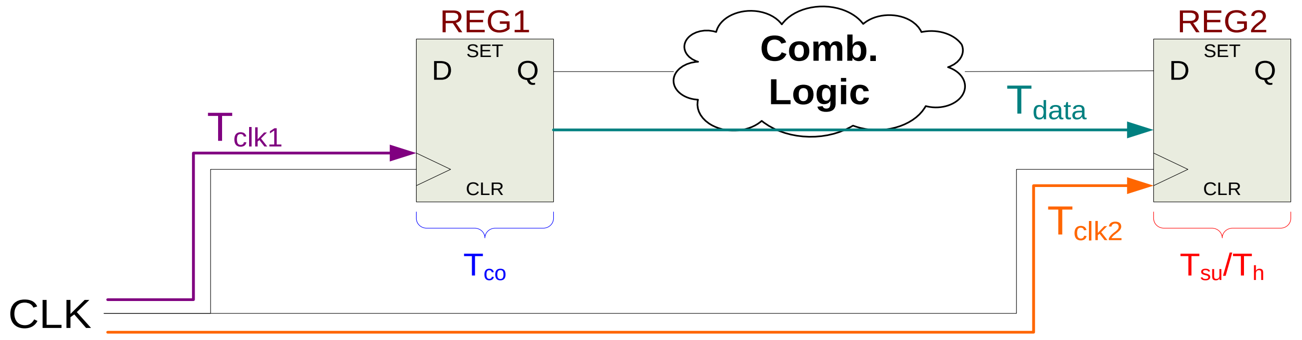
	[illegible]	[illegible]	RF	Type	Fanout	Location	Element
1	5.128	5.128					...ge time
2	7.937	2.809	R				...k delay
3	7.975	0.038		uTsu	1	LCFF_X40_Y18_N11	...p_error

课程安排

- 时序收敛流程
- **如何解决FPGA中存在的时序问题**
- 通过FPGA设计工具进行时序优化

分析时序不满足

- Typical synchronous path
  - Registers can be internal or external to FPGA



Failure type	REG1	REG2
Input	External	Internal
Output	Internal	External
Internal	Internal	Internal

Slack

Setup Slack Equation:

$$\underbrace{(\text{latch edge} + T_{clk2} - \boxed{T_{su}})}_{\text{Data Required}} - \underbrace{(\text{launch edge} + T_{clk1} + \boxed{T_{co}} + T_{data})}_{\text{Data Arrival}}$$

T_{su}, T_h, T_{co} 为固定值

Hold Slack Equation:

$$\underbrace{(\text{launch edge} + T_{clk1} + \boxed{T_{co}} + T_{data})}_{\text{Data Arrival}} - \underbrace{(\text{latch edge} + T_{clk2} + \boxed{T_{h}})}_{\text{Data Required}}$$

Slack

Setup Slack Equation:

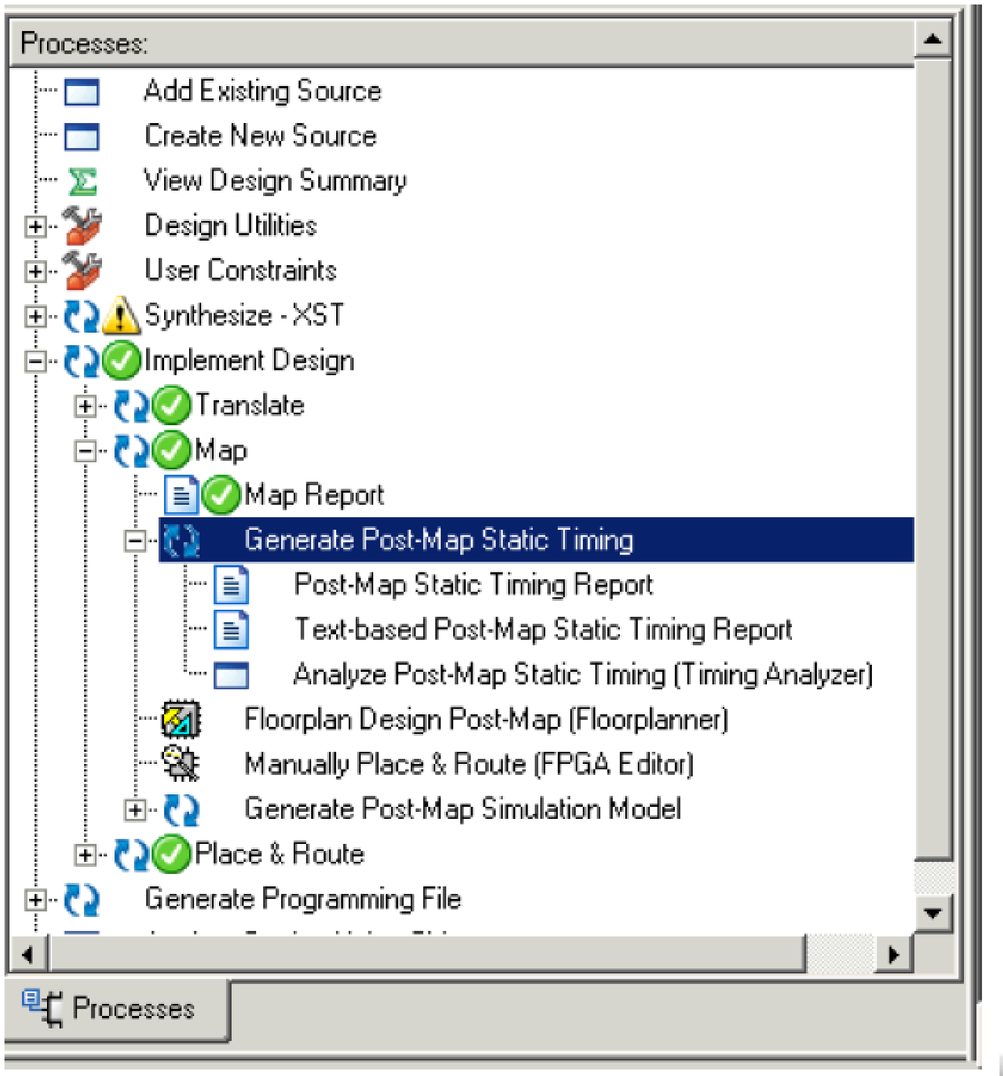
$$\underbrace{(\text{latch edge} + T_{clk2} - T_{su})}_{\text{Data Required}} - \underbrace{(\text{launch edge} + T_{clk1} + T_{co} + T_{data})}_{\text{Data Arrival}}$$

Timing issues show up here

Hold Slack Equation:

$$\underbrace{(\text{launch edge} + T_{clk1} + T_{co} + T_{data})}_{\text{Data Arrival}} - \underbrace{(\text{latch edge} + T_{clk2} + T_{h})}_{\text{Data Required}}$$

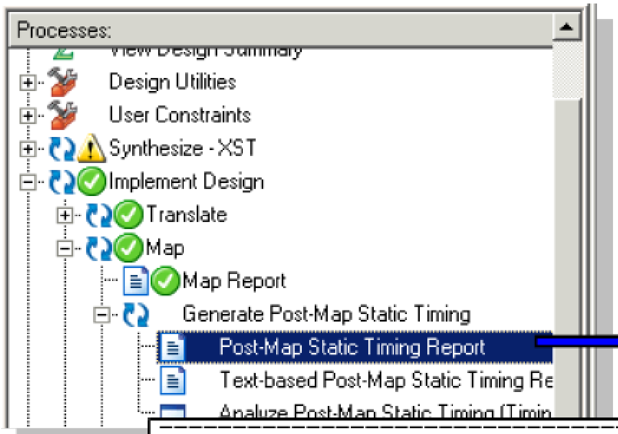
判断设计是否满足性能要求？



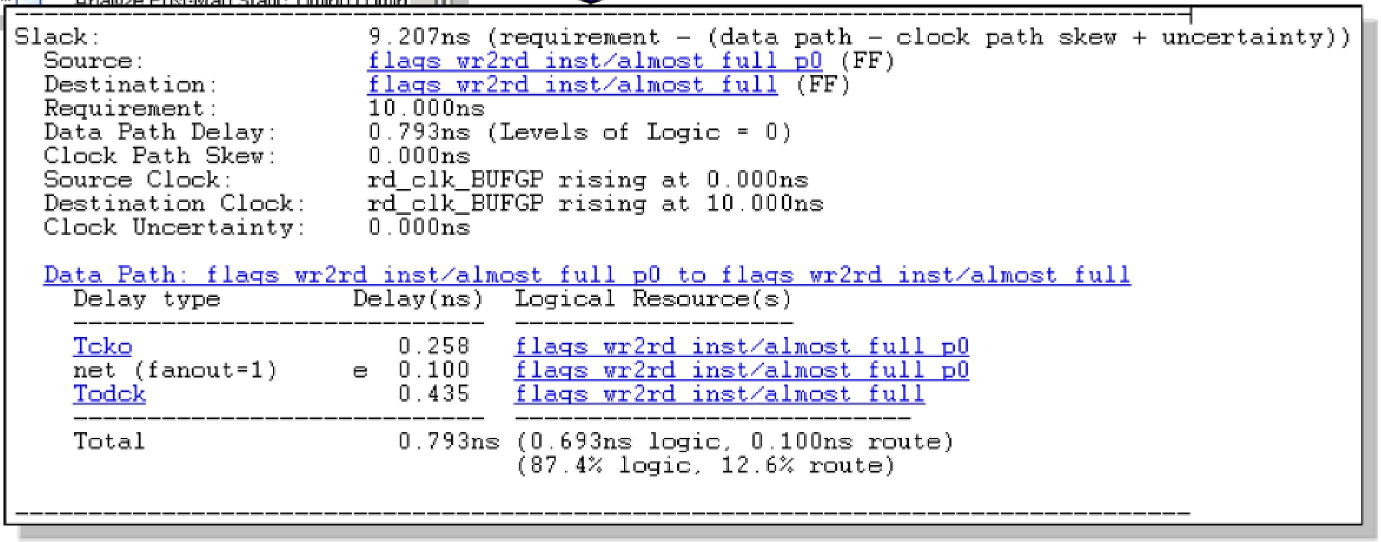
Processes:

- Add Existing Source
- Create New Source
- View Design Summary
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design
  - Translate
  - Map
    - Map Report
    - Generate Post-Map Static Timing
      - Post-Map Static Timing Report
      - Text-based Post-Map Static Timing Report
      - Analyze Post-Map Static Timing (Timing Analyzer)
    - Floorplan Design Post-Map (Floorplanner)
    - Manually Place & Route (FPGA Editor)
    - Generate Post-Map Simulation Model
  - Place & Route
- Generate Programming File

Post-Map Static Timing Report
给出合理的约束参考：60/40 准则



Processes:
View Design Summary
Design Utilities
User Constraints
Synthesize - XST
Implement Design
Translate
Map
Map Report
Generate Post-Map Static Timing
Post-Map Static Timing Report
Text-based Post-Map Static Timing Re
Analyze Post-Map Static Timing (Timin



```
Slack:                    9.207ns (requirement - (data path - clock path skew + uncertainty))
  Source:                 flags_wr2rd_inst/almost_full_p0 (FF)
  Destination:            flags_wr2rd_inst/almost_full (FF)
  Requirement:            10.000ns
  Data Path Delay:        0.793ns (Levels of Logic = 0)
  Clock Path Skew:        0.000ns
  Source Clock:           rd_clk_BUFGP rising at 0.000ns
  Destination Clock:      rd_clk_BUFGP rising at 10.000ns
  Clock Uncertainty:      0.000ns

  Data Path: flags_wr2rd_inst/almost_full_p0 to flags_wr2rd_inst/almost_full
    Delay type         Delay(ns)  Logical Resource(s)
    ----------------------------  -------------------
    Tcko                  0.258   flags_wr2rd_inst/almost_full_p0
    net (fanout=1)     e  0.100   flags_wr2rd_inst/almost_full_p0
    Todck                 0.435   flags_wr2rd_inst/almost_full
    ----------------------------  ---------------------------
    Total                 0.793ns (0.693ns logic, 0.100ns route)
                                  (87.4% logic, 12.6% route)

--------------------------------------------------------------------------------
```

评估、处理不合理的约束

```
Slack:              0.828ns (requirement - (data path - clock path skew + uncertainty))
  Source:             pn_correlator_inst/pn_correlation_fsm_inst/pn_addr_cntr_1 (FF)
  Destination:        pn_correlator_inst/pn_correlation_fsm_inst/pn_addr_cntr_9 (FF)
  Requirement:        3.000ns
  Data Path Delay:    2.172ns (Levels of Logic = 3)
  Clock Path Skew:    0.000ns
  Source Clock:       clk_bufr_BUFGP rising at 0.000ns
  Destination Clock:  clk_bufr_BUFGP rising at 20.000ns
  Clock Uncertainty:  0.000ns
  Data Path: pn_correlator_inst/pn_correlation_fsm_inst/pn_addr_cntr_1 to pn_correlator_inst/pn_correlation_fsm_inst/pn_addr_cntr_9
    Delay type         Delay(ns)  Logical Resource(s)
    -----------------  ---------  -------------------
    Tcko                  0.258   pn_correlator_inst/pn_correlation_fsm_inst/pn_addr_cntr_1
    net (fanout=8)     e  0.100   pn_correlator_inst/pn_correlation_fsm_inst/pn_addr_cntr<1>
    Tilo                  0.146   pn_correlator_inst/pn_correlation_fsm_inst/_n002312
    net (fanout=3)     e  0.100   pn_correlator_inst/pn_correlation_fsm_inst/_n0023_map1085
    Tif5x                 0.428   pn_correlator_inst/pn_correlation_fsm_inst/_n0027_1_G
                                  pn_correlator_inst/pn_correlation_fsm_inst/_n0027_1
    net (fanout=2)     e  0.100   pn_correlator_inst/pn_correlation_fsm_inst/_n00271
    Tilo                  0.146   pn_correlator_inst/pn_correlation_fsm_inst/_n0005<9>76
    net (fanout=1)     e  0.100   pn_correlator_inst/pn_correlation_fsm_inst/_n0005<9>_map1059
    Tsrck                 0.794   pn_correlator_inst/pn_correlation_fsm_inst/pn_addr_cntr_9
    ---------------   ----------  --------------------------
    Total                 2.172ns (1.772ns logic, 0.400ns route)
                                  (81.6% logic, 18.4% route)
```

分析Post-Place & Route Timing

- Timing Error有很多原因
  - Poor micro-architecture
  - Neglecting synchronous design rules or using incorrect HDL codin g style
  - Poor synthesis results (too many logic levels in the path)
  - Inaccurate or incomplete timing constraints
  - Poor logic mapping or placement
- Each root cause has a different solution
  - Rewrite HDL code
  - Ensure that synthesis constraints are correct and use proper synthes is options
  - Add path-specific timing constraints
  - Resynthesize or reimplement with different software options

Case 1

```
Data Path: source to dest
    Delay type         Delay(ns)  Logical Resource(s)
    ----------------------------  -------------------
    Tcko                  0.290   source
    net (fanout=7)        0.325   net_1
    Tilo                  0.060   lut_1
    net (fanout=1)        1.500   net_2
    Tilo                  0.060   lut_2
    net (fanout=1)        0.245   net_3
    Tilo                  0.060   lut_3
    net (fanout=1)        0.204   net_4
    Tdick                 0.300   dest
    ----------------------------  ------------------------------
    Total                 3.044ns (0.770ns logic, 2.274ns route)
                                  (25.3% logic, 74.7% route)
```

- This path is constrained to 3 ns
- What is the primary cause of the timing failure?

Case 1 Answer

```
Data Path: source to dest
    Delay type              Delay(ns)   Logical Resource(s)
    ----------------------------        -------------------
    Tcko                        0.290   source
    net (fanout=7)              0.325   net_1
    Tilo                        0.060   lut_1
    net (fanout=1)              1.500   net_2
    Tilo                        0.060   lut_2
    net (fanout=1)              0.245   net_3
    Tilo                        0.060   lut_3
    net (fanout=1)              0.204   net_4
    Tdick                       0.300   dest
    ----------------------------        ------------------------------
    Total                       3.044ns (0.770ns logic, 2.274ns route)
                                        (25.3% logic, 74.7% route)
```

- What is the primary cause of the timing failure?
  - The net_2 signal has a long delay and low fanout
  - Most likely cause **is poor placement**

Poor Placement: Solutions

- Increase placement effort level (or overall effort level)
- PAR extra effort or SmartXplorer
- Area constraints with the PlanAhead™ tool

Case 2

```
Data Path: source to dest
    Delay type            Delay(ns)  Logical Resource(s)
    ----------------------------  -------------------
    Tcko                     0.290   source
    net (fanout=7)           0.125   net_1
    Tilo                     0.060   lut_1
    net (fanout=187)         2.500   net_2
    Tilo                     0.060   lut_2
    net (fanout=1)           0.174   net_3
    Tilo                     0.060   lut_3
    net (fanout=1)           0.204   net_4
    Tdick                    0.300   dest
    ----------------------------  ------------------------------
    Total                    3.773ns (0.770ns logic, 3.003ns route)
                                     (20.0% logic, 80.0% route)
```

- This path is also constrained to 3 ns
- What is the primary cause of the timing failure?

Case 2 Answer

```
Data Path: source to dest
    Delay type            Delay(ns)  Logical Resource(s)
    ----------------------------  -------------------
    Tcko                     0.290  source
    net (fanout=7)           0.125  net_1
    Tilo                     0.060  lut_1
    net (fanout=187)         2.500  net_2
    Tilo                     0.060  lut_2
    net (fanout=1)           0.174  net_3
    Tilo                     0.060  lut_3
    net (fanout=1)           0.204  net_4
    Tdick                    0.300  dest
    ----------------------------  ------------------------------
    Total                    3.773ns (0.770ns logic, 3.003ns route)
                                     (20.0% logic, 80.0% route)
```

- What is the primary cause of the timing failure?
  - The signal net_2 has a long delay, but the fanout is not low
  - Most likely cause is high fanout

High Fanout: Solutions

- Most likely solution is to duplicate the source of the high-fan out net
  - If the net is the output of a flip-flop, the solution is to dup licate the flip-flop
    - Use manual duplication (recommended) or synthesis options
  - If the net is driven by combinatorial logic, locating the so urce of the net in the HDL code can be more difficult
    - Use synthesis options to duplicate the source
    - Duplicate one or more flip-flops upstream from the net

Case 3

```
Data Path: source to dest
    Delay type              Delay(ns)  Logical Resource(s)
    ----------------------------  -------------------
    Tcko                       0.290   source
    net (fanout=7)             0.521   net_1
    Tilo                       0.060   lut_1
    net (fanout=1)             0.280   net_2
    Tilo                       0.060   lut_2
    net (fanout=1)             0.223   net_3
    Tilo                       0.060   lut_3
    net (fanout=1)             0.223   net_4
    Tilo                       0.060   lut_4
    net (fanout=1)             0.310   net_5
    Tilo                       0.060   lut_5
    net (fanout=1)             0.233   net_6
    Tilo                       0.060   lut_6
    net (fanout=1)             0.308   net_7
    Tdick                      0.300   dest
    ----------------------------  ---------------------------------------
    Total                      3.048ns (0.950ns logic, 2.098ns route)
                                       (31.2% logic, 68.8% route)
```

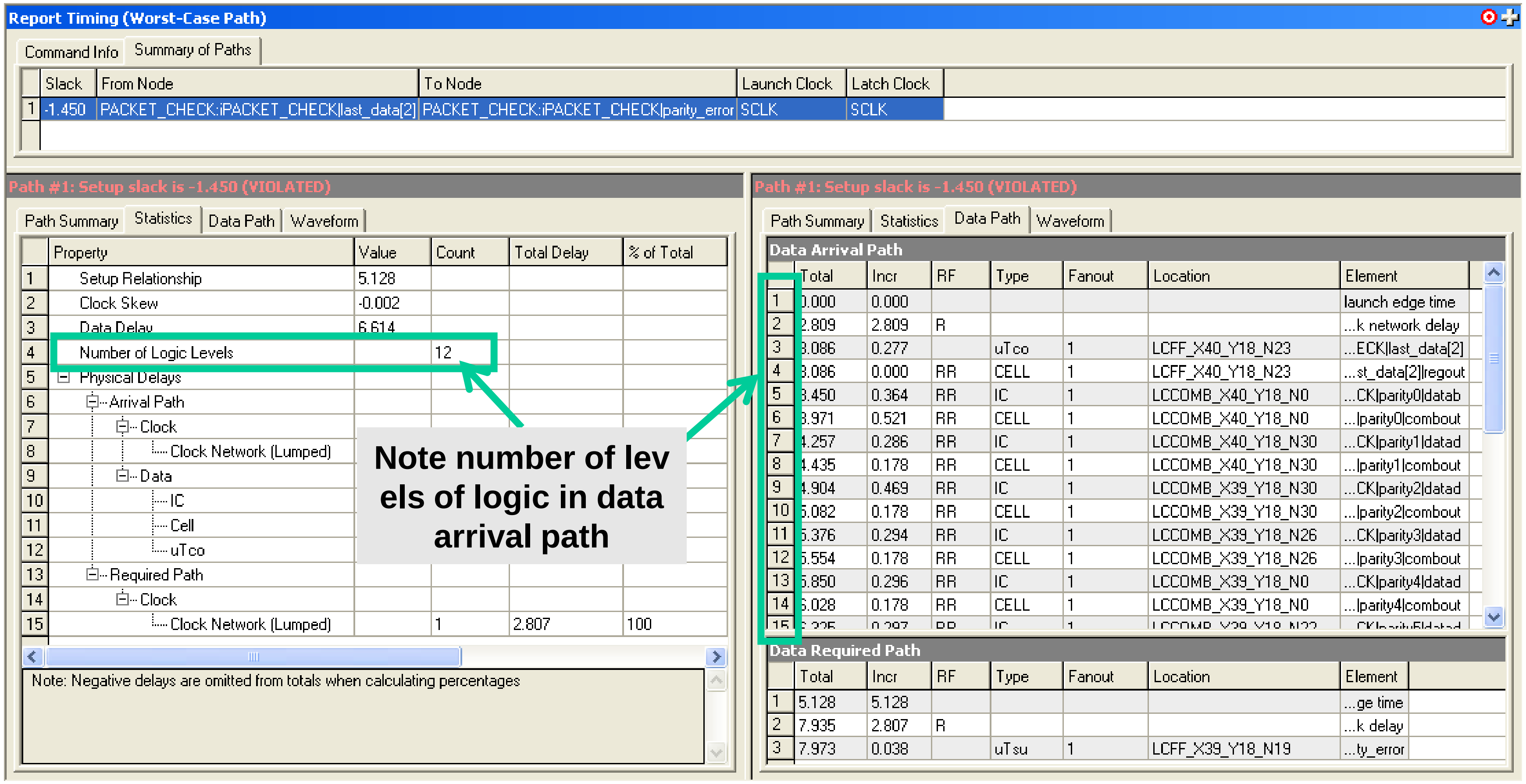
- This path is also constrained to 3 ns
- What is the primary cause of the timing failure?

Case 3 Answer

```
Data Path: source to dest
    Delay type            Delay(ns)  Logical Resource(s)
    ----------------------------  -------------------
    Tcko                     0.290   source
    net (fanout=7)           0.521   net_1
    Tilo                     0.060   lut_1
    net (fanout=1)           0.180   net_2
    Tilo                     0.060   lut_2
    net (fanout=1)           0.223   net_3
    Tilo                     0.060   lut_3
    net (fanout=1)           0.123   net_4
    Tilo                     0.060   lut_4
    net (fanout=1)           0.310   net_5
    Tilo                     0.060   lut_5
    net (fanout=1)           0.233   net_6
    Tilo                     0.060   lut_6
    net (fanout=1)           0.308   net_7
    Tdick                    0.300   dest
    ----------------------------  ---------------------------------------
    Total                    3.048ns (0.950ns logic, 2.098ns route)
                                     (31.2% logic, 68.8% route)
```

- What is the primary cause of the timing failure?
  - There are no really long delays, but there are a lot of logic levels

TimeQuest



Report Timing (Worst-Case Path)

Command Info | Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock
1	-1.450	PACKET_CHECK:iPACKET_CHECK|last_data[2]	PACKET_CHECK:iPACKET_CHECK|parity_error	SCLK	SCLK

Path #1: Setup slack is -1.450 (VIOLATED)

Path Summary | Statistics | Data Path | Waveform

	Property	Value	Count	Total Delay	% of Total
1	Setup Relationship	5.128			
2	Clock Skew	-0.002			
3	Data Delay	6.614			
4	Number of Logic Levels		12		
5	Physical Delays				
6	Arrival Path				
7	Clock				
8	Clock Network (Lumped)				
9	Data				
10	IC				
11	Cell				
12	uTco				
13	Required Path				
14	Clock				
15	Clock Network (Lumped)		1	2.807	100

Note: Negative delays are omitted from totals when calculating percentages

Note number of levels of logic in data arrival path

Path #1: Setup slack is -1.450 (VIOLATED)

Path Summary | Statistics | Data Path | Waveform

Data Arrival Path

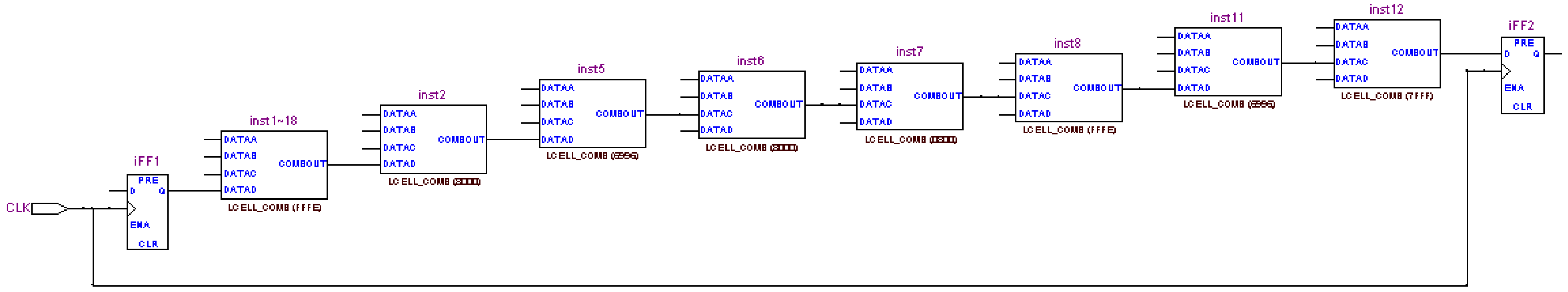
	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	2.809	2.809	R				...k network delay
3	3.086	0.277		uTco	1	LCFF_X40_Y18_N23	...ECK|last_data[2]
4	3.086	0.000	RR	CELL	1	LCFF_X40_Y18_N23	...st_data[2]|regout
5	3.450	0.364	RR	IC	1	LCCOMB_X40_Y18_N0	...CK|parity0|datab
6	3.971	0.521	RR	CELL	1	LCCOMB_X40_Y18_N0	...|parity0|combout
7	4.257	0.286	RR	IC	1	LCCOMB_X40_Y18_N30	...CK|parity1|datad
8	4.435	0.178	RR	CELL	1	LCCOMB_X40_Y18_N30	...|parity1|combout
9	4.904	0.469	RR	IC	1	LCCOMB_X39_Y18_N30	...CK|parity2|datad
10	5.082	0.178	RR	CELL	1	LCCOMB_X39_Y18_N30	...|parity2|combout
11	5.376	0.294	RR	IC	1	LCCOMB_X39_Y18_N26	...CK|parity3|datad
12	5.554	0.178	RR	CELL	1	LCCOMB_X39_Y18_N26	...|parity3|combout
13	5.850	0.296	RR	IC	1	LCCOMB_X39_Y18_N0	...CK|parity4|datad
14	6.028	0.178	RR	CELL	1	LCCOMB_X39_Y18_N0	...|parity4|combout
15	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	5.128	5.128					...ge time
2	7.935	2.807	R				...k delay
3	7.973	0.038		uTsu	1	LCFF_X39_Y18_N19	...ty_error

Technology Map Viewer

Right-click on failing path and select **Locate Endpoints** or **Locate Path**

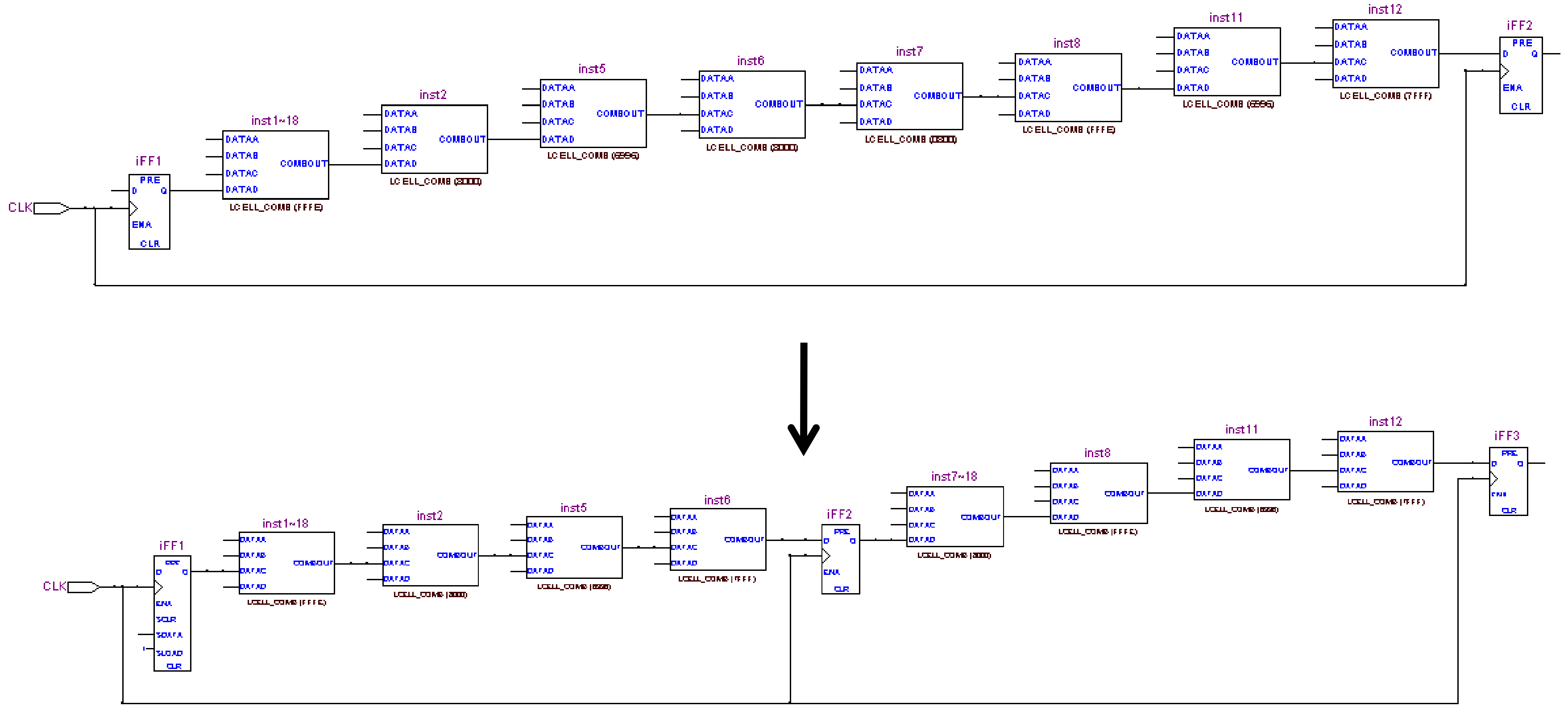


This path has 8 levels of logic

Too Many Logic Levels: Solutions

- The implementation tools cannot do much to improve performance
- The netlist must be altered to reduce the amount of logic between flip-f lops
- Possible solutions
  - Check whether the path is a multicycle path
    - If yes, add a multicycle path constraint
  - Ensure that proper constraints were used during synthesis
  - Use the retiming option during synthesis to distribute logic more evenly a mong flip-flops
  - Confirm that good coding techniques were used to build this logic (no nest ed if or case statements)
  - Change the micro-architecture of this path
    - Add a pipeline stage, manually re-pipeline...

Pipeline Registers



举例

- assign dat_act = ((hcount_r >= hdat_begin) && (hcount_r < hdat _end))&& ((vcount_r >= vdat_begin) && (vcount_r < vdat_end)) ;
- assign tft_r = (dat_act) ? {rgb16_dat[15:11], 3'b111} : 8'h00;
- assign tft_g = (dat_act) ? {rgb16_dat[10:5], 2'b11} : 8'h00;
- assign tft_b = (dat_act) ? {rgb16_dat[4:0], 3'b111} : 8'h00;

亚稳态(Metastability)

- 亚稳态(Metastability)
  - **Flip-flop输出进入到一个过渡状态：即不是一个有效的‘0’，也不是一个有效的‘1’　。**
  - 在稳定在一个有效的值前，保持在亚稳态的时间不可预测。
  - 从统计角度看，亚稳态的发生只能努力减少，不能完全消除。
  - 使用同步器电路解决亚稳态问题，

Violation（1）

- 建立时间不满足要求往往是由于逻辑团的延时过大造成的
- 保持时间不满足要求往往是由于逻辑团的延时过小造成的
- 建立时间和保持时间不满足的情况往往出现在异步时钟域中

Violation（2）

解决建立时间不满足要求的方法：

- 采用over constraint重新综合，对于violation path进一步优化
- 降低时钟频率
- 拆分组合逻辑，增加流水线
- 减小传输延时

Violation（3）

解决保持时间不满足要求的方法：

- 增加逻辑团延时
- 减小时钟延时
- 大多数P&R工具都有“fix hold”选项

课程安排

- 时序收敛流程
- 如何解决FPGA中存在的时序问题
- **通过FPGA设计工具进行时序优化**

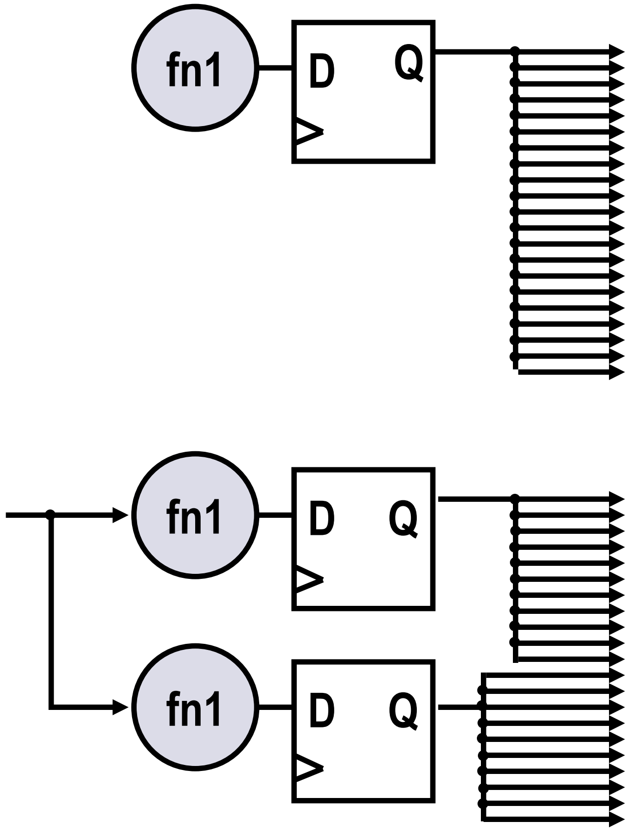
综合技术

- 综合工具提供许多优化选择，以获得期望的系统性能和面积要求

  - ☐ Register Duplication
  - ☐ Timing-Driven Synthesis
  - ☐ Timing Constraint Editor
  - ☐ FSM Extraction
  - ☐ Retiming
  - ☐ Hierarchy Management
  - ☐ Schematic Viewer
  - ☐ Error Navigation
  - ☐ Cross-Probing
  - ☐ Physical Optimization

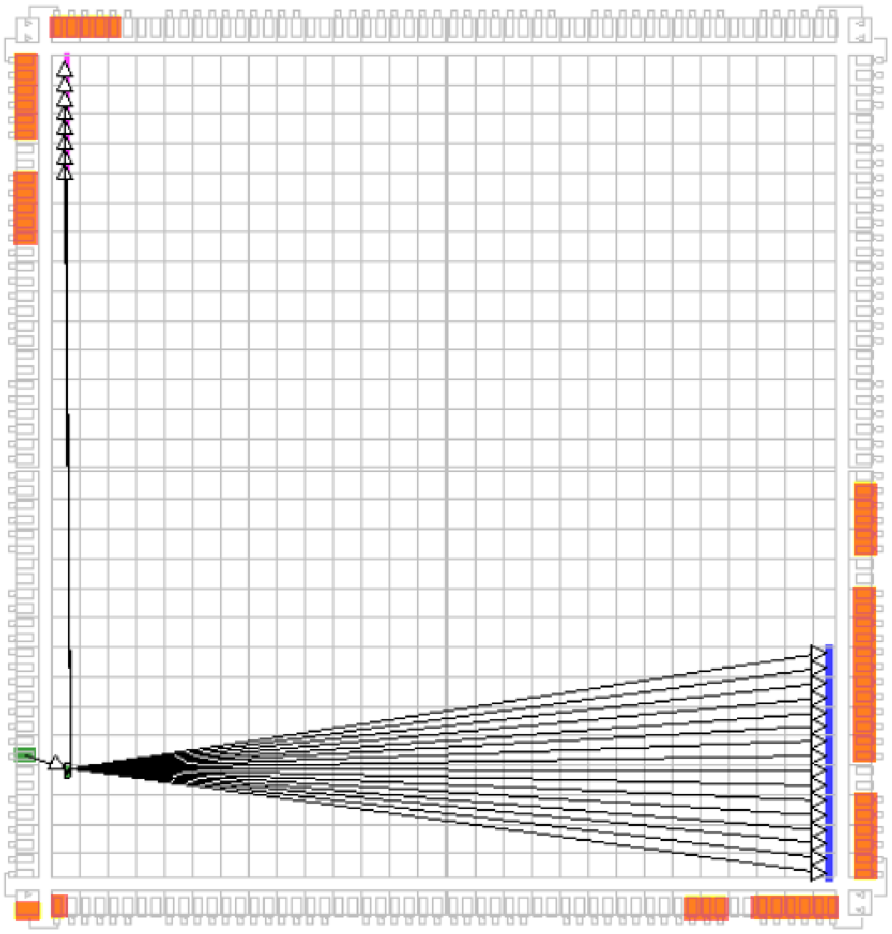
Duplicating Flip-Flops

- 高扇出(High-fanout nets)会很慢，并且难以布线
- Flip-Flop复制可以解决这两个问题
  - 减小扇出可以缩短网络延迟
  - 每个flip-flop向一个不同的芯片物理区域扇出，从而减小布 线拥挤
- Design trade-offs
  - 增加布线可行性和性能
  - 增加面积
  - 增加了其他网络的扇出



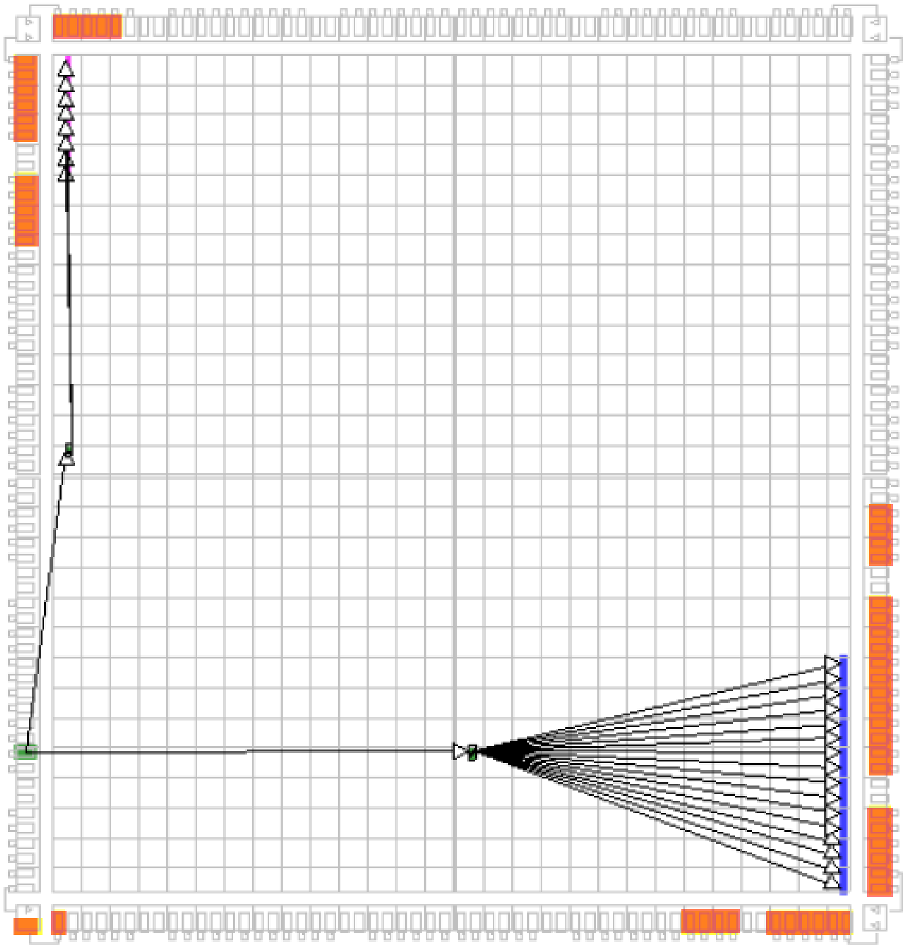
Flip-Flop复制的例子

- 源flip-flop驱动两个flip-flop组(bank)，它们分别位于芯片不同的区域
- 源flip-flop和pad并没有被约束
- 周期约束，PERIOD = 5 ns
- 使用不周的属性进行设计implementation
- 最长路径= 6.806 ns
  - 没有满足时序要求



Flip-Flop复制的例子

- 如果对源 flip-flop 进行复制
- 每个flip-flop 分别驱动芯片上的一个区域
  - 每个flip-flop 可以更加靠近它要驱动的flip-flop组
  - 理短的路延迟
- 最长路径 = 4.666 ns
  - 满足设计时序要求



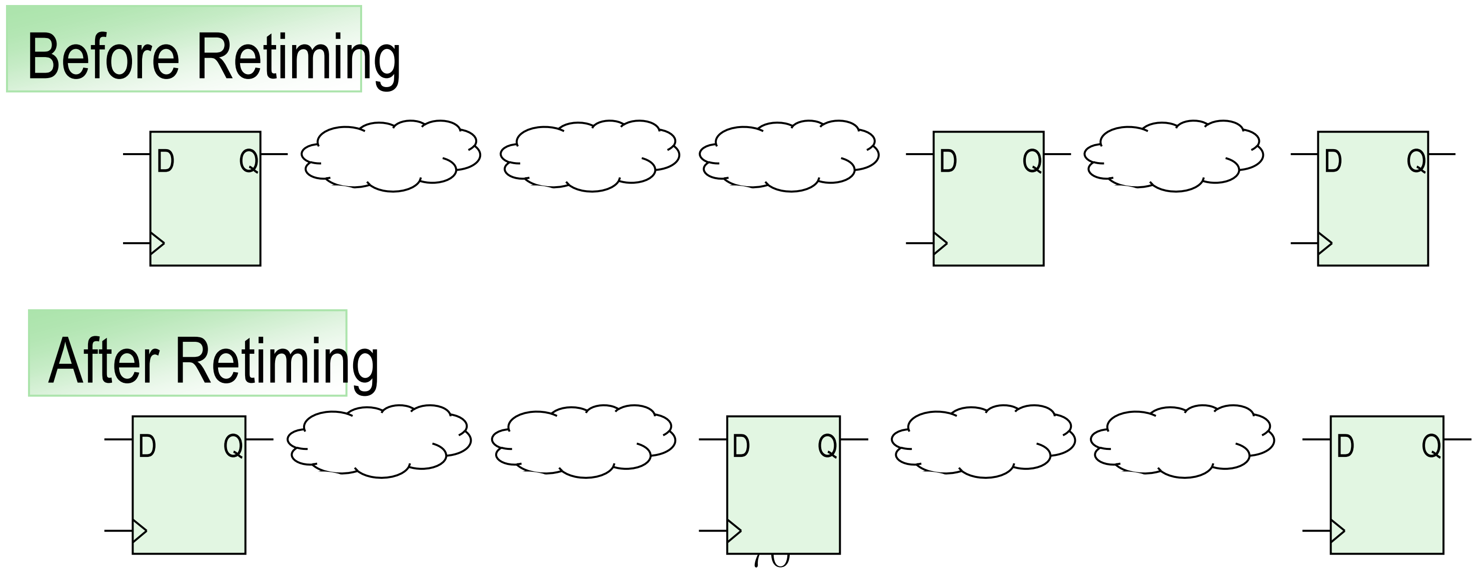
综合技术

Timing-Driven Synthesis

- 实施period约束和input/output约束
  - 通常，根据期望的性能目标进行1.5X－2X的过约束，综合工具会提高工作级别，有利于在实现中更容易满足时序目标
- 使用Multi-cycle和false paths约束
- 使用Critical path约束，对Critical path进行优化

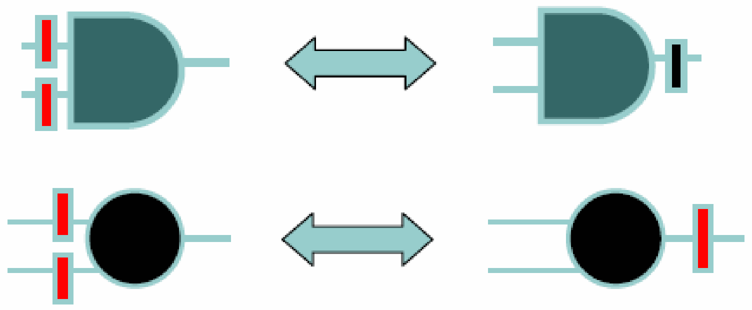
Retiming

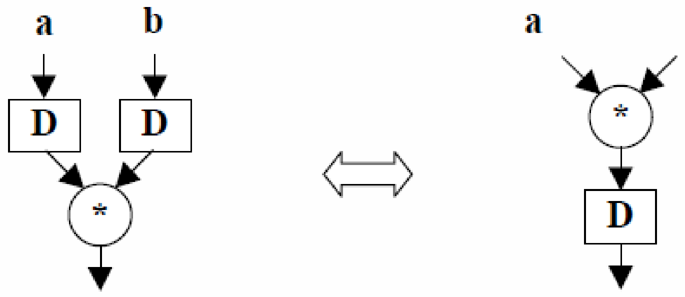
- Synplify, and XST software
- Retiming: The synthesis tool automatically tries to move register stag es to balance combinatorial delay on each side of the registers

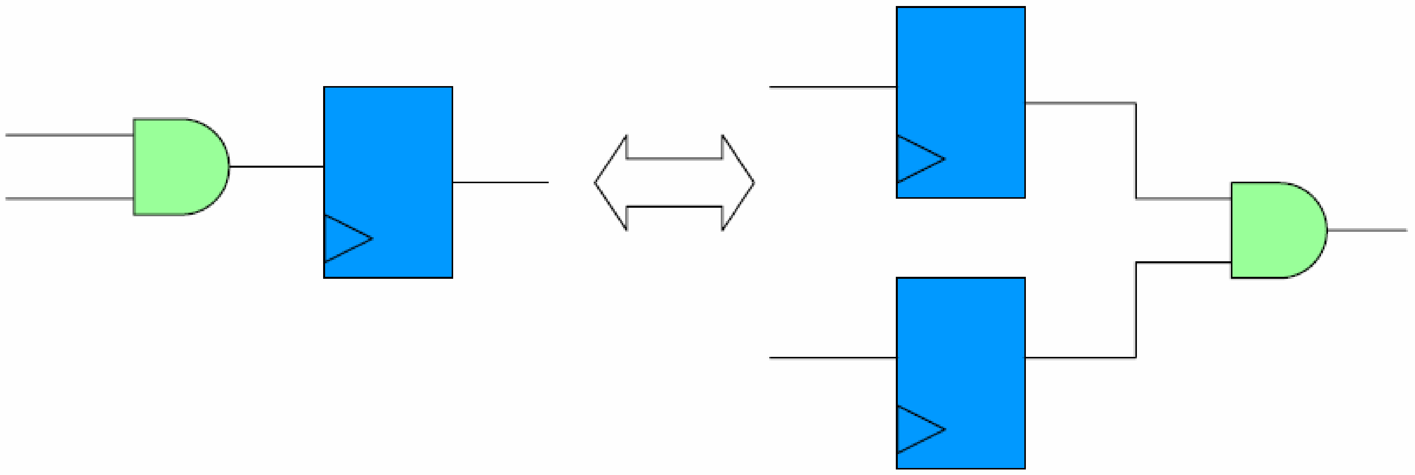


最简单的Retiming

- 在组合逻辑中移动寄存器







Hierarchy Management

- Synplify and XST software
- The basic settings are:
  - Flatten the design: Allows total combinatorial optimization across all bou ndaries
  - Maintain hierarchy: Preserves hierarchy without allowing optimization of combinatorial logic across boundaries

- If you have followed the synchronous design guidelines, use the setting

  -*maintain hierarchy*
- If you have not followed the synchronous design guidelines, use the setting

  -*flatten the design*

管脚约束

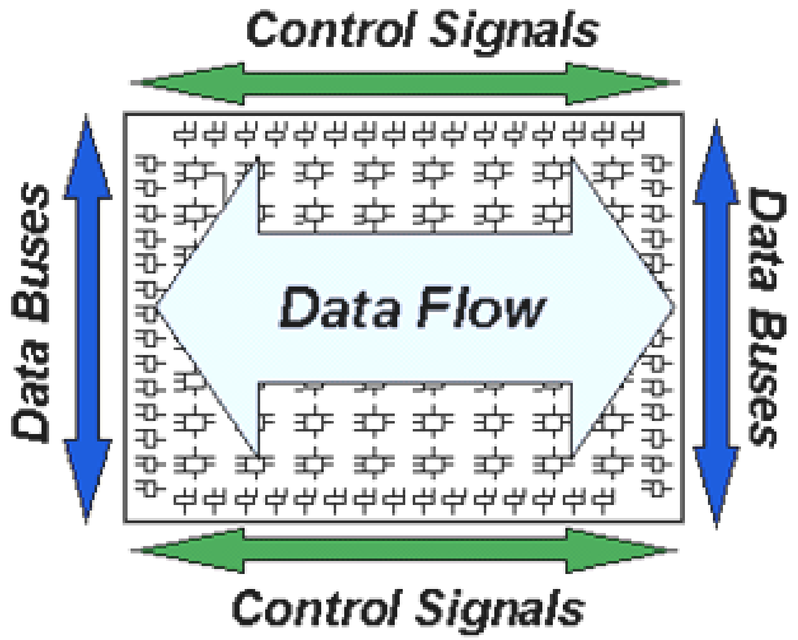
- 管脚约束通常在设计早期就要确定下来，以保证电路板的设计同步进行
- 对高速设计、复杂设计和具有大量I/O管脚的设计，Xilinx推荐手工进行管脚约束
  - 实现工具可以自动布局逻辑和管脚，但是一般来说不会是最优的
  - 管脚约束可以指导内部数据流向,不合理的管脚布局很容易降低系统性能
  - 合理的管脚布局需要对所设计系统和Xilinx器件结构的详细了解，如要考虑I/O bank、I/O电气标准等
  - 时钟(单端或差分)必须约束在专用时钟管脚

    注意：时钟资源数量的限制
  - 最后使用dual-purpose管脚(如配置管脚)

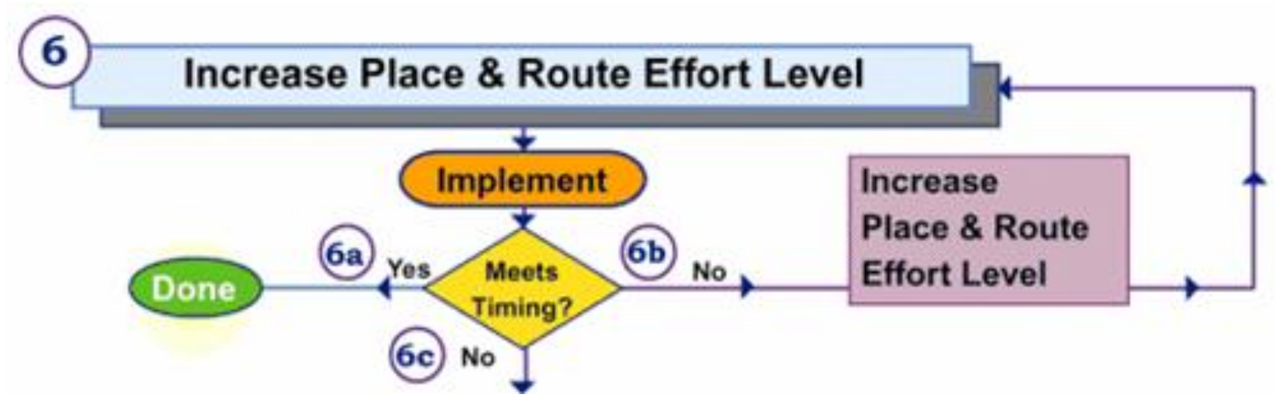
根据数据流指导管脚约束

- 用于控制信号的I/O置于器件的顶部或底部
  - 控制信号垂直布置
- 用于数据总线的I/O置于器件的左部和右部
  - 数据流水平布置。

- 以上布局方法可以充分利用Xilinx器件的资源布局方式
  - 进位链排列方式
  - 块RAM， 乘法器位置



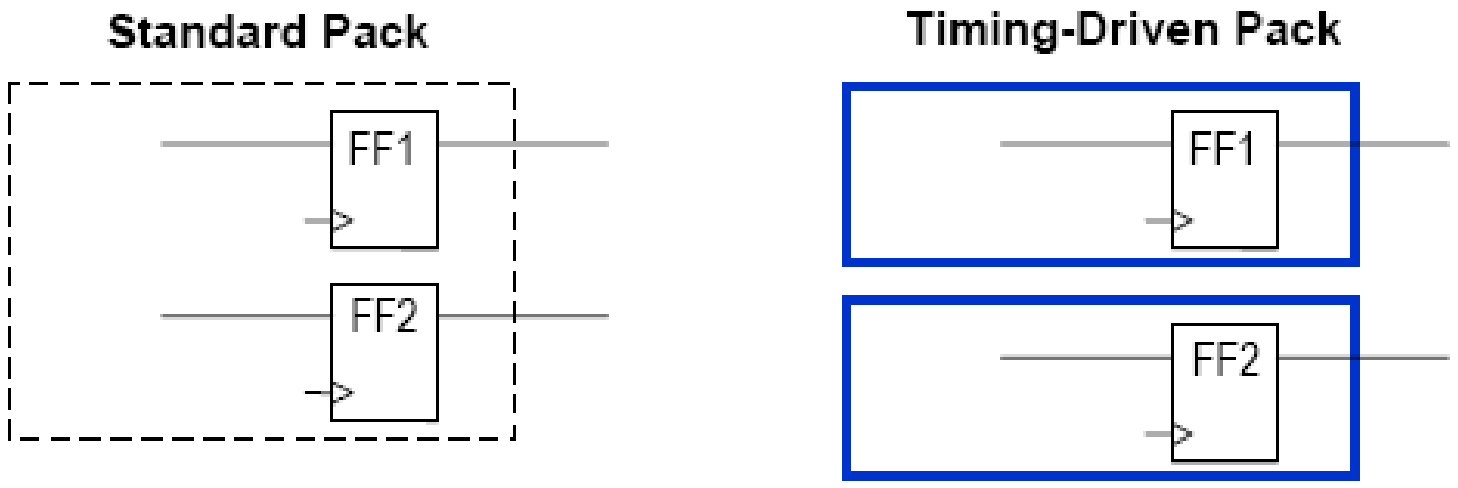
R&R参数选项：Effort Level


6
Increase Place & Route Effort Level
Implement
6a
Yes
Meets Timing?
6b
No
Increase Place & Route Effort Level
Done
6c
No

- 使用更高级别的Effort Level：可以提高时序性能
- Xilinx推荐：第一遍实现时，使用全局时序约束和缺省的实现参数选项。如果不能满足时序要求：
  - 尝试修改代码，如使用合适的代码风格，增加流水线等
  - 修改综合参数选项，如Optimization Effort , Keep Hierarchy , Register Duplication, Register Balancing 等
  - 增加PAR Effort Level
  - Apply path-specific timing constraints for synthesis and implementation

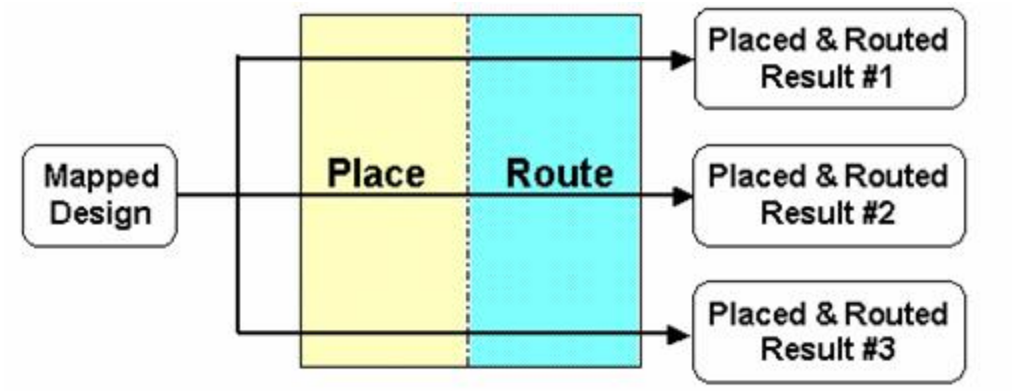
Timing Driven Packing

- Originally, the flip-flops were packed together into a slice.
- After placement and timing analysis, the flip-flops are packed into differ ent slices to allow independent movement



MPPR和PAR Extra Effort

- MPPR：对同一个设计运行PAR多次，试图找到最可能满足设计要求的结果，保留作为设计结果


Mapped Design
Place
Route
Placed & Routed Result #1
Placed & Routed Result #2
Placed & Routed Result #3

- 当最高级别的PAR Effort Level被选择时，PAR Extra Effort可选三种选择：None,Normal和Continue on impossible
- 典型情况下，大约可以提高4%的性能
- 通常PAR消耗更多的时间(增加200%以上)

实例：初始代码

```
module tst_ori_asynrst(
   clk,
   rst_n,
   fst,
   out
   );
   input          clk;
   input          rst_n;
   input          fst;
   output[2:0]    out;
   reg[2:0]       out;
   reg[9:0]       cnt;

   //counter 640
   always@(posedge clk or negedge rst_n)
      if(!rst_n)                  // reset expression "
         cnt <= 10'b0;
      else
         if(!fst)
            cnt <= 10'b0;
         else
            if(cnt >= 10'd639)
               cnt <= 10'b0;
            else
               cnt <= cnt + 1'b1;

   //output
   always@(posedge clk or negedge rst_n)
      if(!rst_n)
         out <= 3'b0;
      else
         if(cnt == 10'd608)
            out <= 3'b001;
         else if(cnt == 10'd618)
            out <= 3'b010;
         else if(cnt == 10'd628)
            out <= 3'b100;
         else
            out <= 3'b0;

endmodule
```

初始代码

```
always@(posedge clk or negedge
rst_n)
if (!rst_n)
      cnt <= 10'b0;
else
     if(!fst)
            cnt <= 10'b0;
     else if (cnt >= 10'd639)
            cnt <= 10'b0;
     else
            cnt <= cnt + 1'b1;
```

优化：修改计数器归零条件

```
module tst_ori_opti_comparor_asynrst(
    clk,
    rst_n,
    fst,
    out
    );
    input           clk;
    input           rst_n;
    input           fst;
    output[2:0]     out;
    reg[2:0]        out;
    reg[9:0]        cnt;

    //counter 640
    always@(posedge clk or negedge rst_n)      // asynchronous reset
        if(!rst_n)                         // reset expression "(!rst_n || !fst)" modify to "!rst_n"
           cnt <= 10'b0;
        else
           if(!fst)
              cnt <= 10'b0;
           else
              if(cnt == 10'd639)    //if condition "cnt >= 10'd639" modify to "cnt == 10'd639"
                 cnt <= 10'b0;
              else
                 cnt <= cnt + 1'b1;

    //output
    always@(posedge clk or negedge rst_n)      // asynchronous reset
        if(!rst_n)
           out <= 3'b0;
        else
           if(cnt == 10'd608)
              out <= 3'b001;
           else if(cnt == 10'd618)
              out <= 3'b010;
           else if(cnt == 10'd628)
              out <= 3'b100;
           else
              out <= 3'b0;

endmodule
```

if (cnt >= 10’d639)

↓

if (cnt == 10’d639)

优化：修改if-else嵌套为case结构

```
module tst_ori_opti_case_asynrst(
    clk,
    rst_n,
    fst,
    out
    );
    input           clk;
    input           rst_n;
    input           fst;
    output[2:0]     out;
    reg[2:0]        out;
    reg[9:0]        cnt;

    //counter 640
    always@(posedge clk or negedge rst_n)
        if(!rst_n)                  // reset expression "(!rst_n
            cnt <= 10'b0;
        else
            if(!fst)
                cnt <= 10'b0;
            else
                if(cnt == 10'd639)  //if condition "cnt >= 10'd6
                    cnt <= 10'b0;
                else
                    cnt <= cnt + 1'b1;

    //output
    always@(posedge clk or negedge rst_n)
        if(!rst_n)
            out <= 3'b0;
        else
            case(cnt)               // "if-elseif-else" modify to case
                10'd608: out <= 3'b001;
                10'd618: out <= 3'b010;
                10'd628: out <= 3'b100;
                default: out <= 3'b0;
            endcase

endmodule
```

```
if (cnt == 10’d608)
  … else if (cnt == 10’
d618)
  … else if (cnt == 10’
d628)
  …
        ↓
case (cnt)
    10’d608: …
    10’d618: …
    10’d628: …
    default:  …
endcase
```

优化：拆分大计数器成几个小计数器

```
//counter mid-3bit-8,         "cnt"(10-bit) cut to high,mid,low three part
always@(posedge clk or negedge rst_n)      // asynchronous reset
    if(!rst_n)
        cnt_mid <= 3'b0;
    else
        if(!fst)
            cnt_mid <= 3'b0;
        else if(cnt_low == 4'd9)
            cnt_mid <= cnt_mid + 1'b1;
        else
            cnt_mid <= cnt_mid;

//counter low-4bit-10,        "cnt"(10-bit) cut to high,mid,low three part
always@(posedge clk or negedge rst_n)      // asynchronous reset
    if(!rst_n)
        cnt_low <= 4'b0;
    else
        if(!fst)
            cnt_low <= 4'b0;
        else
            if(cnt_low == 4'd9)
                cnt_low <= 4'b0;
            else
                cnt_low <= cnt_low + 1'b1;

//output
always@(posedge clk or negedge rst_n)      // asynchronous reset
    if(!rst_n)
        out <= 3'b0;
    else
        if(cnt_high == 3'd7 && cnt_mid == 3'd4 && cnt_low == 4'd8)        // 608
            out <= 3'b001;
        else if(cnt_high == 3'd7 && cnt_mid == 3'd5 && cnt_low == 4'd8)   // 618
            out <= 3'b010;
        else if(cnt_high == 3'd7 && cnt_mid == 3'd6 && cnt_low == 4'd8)   // 628
            out <= 3'b100;
        else
            out <= 3'b0;

endmodule
```

```
if (cnt == 10'd639)
        ↓
if (cnt_high == 3'd7
    && cnt_mid == 3'd7
    && cnt_low == 4'd9)
```

优化：简化if条件表达式中逻辑

```
//output
//============v=================v================
always@(posedge clk or negedge rst_n)      // asynchronous rese
   if(!rst_n)
      out <= 3'b0;
   else
      if(fa_high & fa_mid & fa_low)        // 608
         out <= 3'b001;
      else if(fb_high & fb_mid & fb_low)   // 618
         out <= 3'b010;
      else if(fc_high & fc_mid & fc_low)   // 628
         out <= 3'b100;
      else
         out <= 3'b0;
//==============^=================^===============

//get flag "fa_high"
always@(posedge clk or negedge rst_n)      // asynchronous rese
   if(!rst_n)
      fa_high <= 1'b0;
   else
      if(cnt_high == 3'd7)
         fa_high <= 1'b1;
      else
         fa_high <= 1'b0;

//get flag "fa_mid"
always@(posedge clk or negedge rst_n)      // asynchronous reset
   if(!rst_n)
      fa_mid <= 1'b0;
   else
      if(cnt_mid == 3'd4)
         fa_mid <= 1'b1;
      else
         fa_mid <= 1'b0;

//get flag "fa_low"
always@(posedge clk or negedge rst_n)      // asynchronous reset
   if(!rst_n)
      fa_low <= 1'b0;
```

```
if (cnt_high == 3'd7
    && cnt_mid == 3'd7
    && cnt_low == 4'd9)
              ↓
if (flaga_high  == 1'b1
    && flaga_mid == 1'b
1
    && flaga_low == 1'b
1)
```