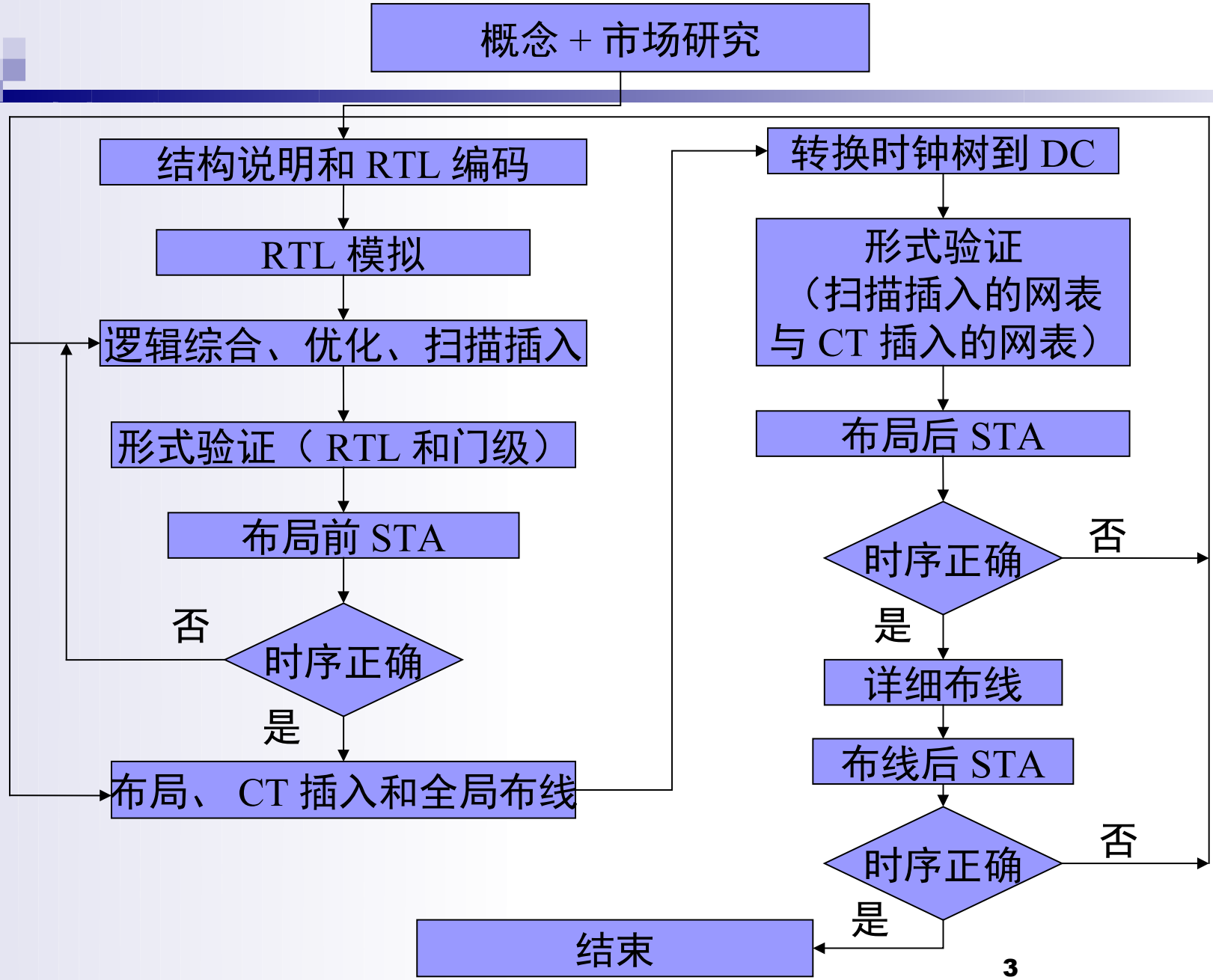
静态时序分析
Static Timing Analysis

-- 于斌

概要

- 时序分析概述
- 时序分析中的基本概念
- 常用工具简介

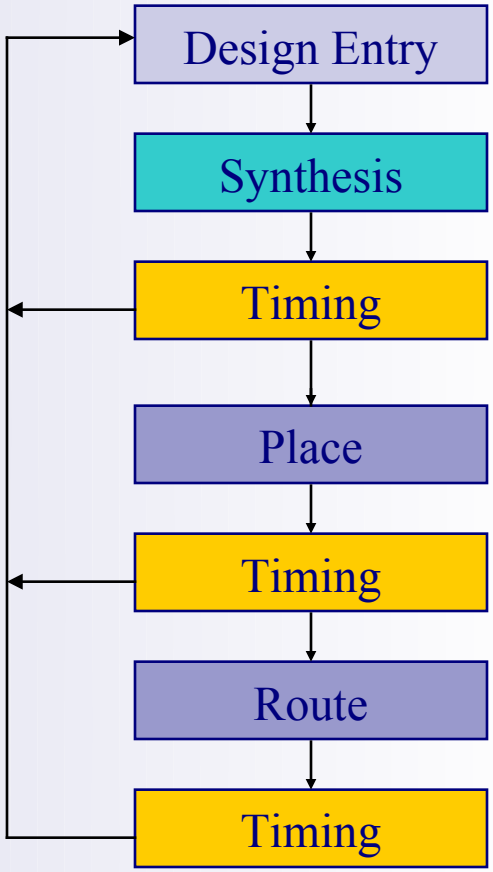

概念 + 市场研究
结构说明和 RTL 编码
RTL 模拟
逻辑综合、优化、扫描插入
形式验证（ RTL 和门级）
布局前 STA
否
时序正确
是
布局、 CT 插入和全局布线
转换时钟树到 DC
形式验证
（扫描插入的网表
与 CT 插入的网表）
布局后 STA
时序正确
否
是
详细布线
布线后 STA
时序正确
否
是
结束

1. 结构及电学特性规范
2.HDL 中的 RTL 编码
3. 为包含存储单元的设计插入 DFT memory BIST
4. 为验证设计功能，进行详尽的动态仿真
5. 设计环境设置，包括将使用的工艺库和其他环境属性
6. 使用 DC 对具有扫描插入（和可选 JTAG ）的设计进行约束和综合设计
7. 使用 DC 的内建静态时序分析机进行模块级的静态时序分析
8. 设计的形式验证，使用 Formality 将 RTL 和综合后的网表进行对比
9. 使用 PT 进行整个设计布局前的静态时序分析
10. 对布局工具进行时序约束前的前标注
11. 具有时序驱动单元布局、时钟树插入和全局布线的初始布局划分
12. 将时钟树转换到驻留在ＤＣ中的原始设计

13. 在 DC 中进行设计的布局优化
14. 使用 Formality 在综合网表和时钟树插入的网表之间进行进行形式验证
15. 在全局布线后（ 11 步）
16. 从全局布线得到的估计时间数据反标注到 PT
17. 使用全局布线后提取的估计延时数据在 PT 中进行静态时序分析
18. 设计的详细布局
19. 提取来自详细布局设计的实际时间延迟
20. 实际提取时间数据反标注到 PT
21. 使用 PT 进行布局后的静态时序分析
22. 布局后的门级功能仿真（如果需要的话）
23. 在 LVS 和 DRC 验证之后交货

时序分析概述

- 与时序相关的流程



动态时序仿真
静态时序分析
形式验证

动态时序仿真与静态时序分析

- 动态时序仿真是针对给定的仿真输入信号波形，模拟设计在器件实际工作时的功能和延时情况，给出相应的仿真输出信号波形。**它主要用于验证设计在器件实际延时情况下的逻辑功能。**由动态时序仿真报告无法得到设计的各项时序性能指标，如最高时钟频率等。

- 静态时序分析则是通过分析每个时序路径的延时，计算出设计的各项时序性能指标，如最高时钟频率、建立保持时间等，发现时序违规。它仅仅聚焦于时序性能的分析，并不涉及设计的逻辑功能，**逻辑功能验证仍需通过仿真或其他手段（如形式验证等）进行。**静态时序分析是最常用的分析、调试时序性能的方法和工具。

静态时序分析 -Static Timing Analysis

- STA 是一种**验证方法**
- STA 的前提是**同步逻辑设计**
- STA 是使用工具通过路径计算延迟的综合，并比较相对预定义时钟的延迟
- STA 仅**关注时序**间的相对关系而**不是评估逻辑功能**
- 无需用向量去激活某个路径，而是对所有的时序路径进行错误分析，能处理百万门级的设计，分析速度比时序仿真工具快几个数量级，在同步逻辑情况下，可以达到 100% 的时序路径覆盖
- STA 的目的是找出隐藏的时序问题，根据时序分析结果优化逻辑或约束条件，使设计**达到时序闭合**（timing closure）

STA 的作用

- **确定芯片最高工作频率**

  通过时序分析可以控制工程的综合、映射、布局布线等环节，减少延迟，从而尽可能提高工作频率

- **检查时序约束是否满足**

  可以通过时序分析来查看目标模块是否满足约束，如不满足，可以定位到不满足约束的部分，并给出具体原因，进一步修改程序直至满足时序要求

- **分析时钟质量**

  时钟存在抖动、偏移、占空比失真等不可避免的缺陷。通过时序分析可以验证其对目标模块的影响

STA 的过程

- STA 分三步走：

1 、将设计打散成一个一个的 timing path

2 、计算每条 path 的延迟

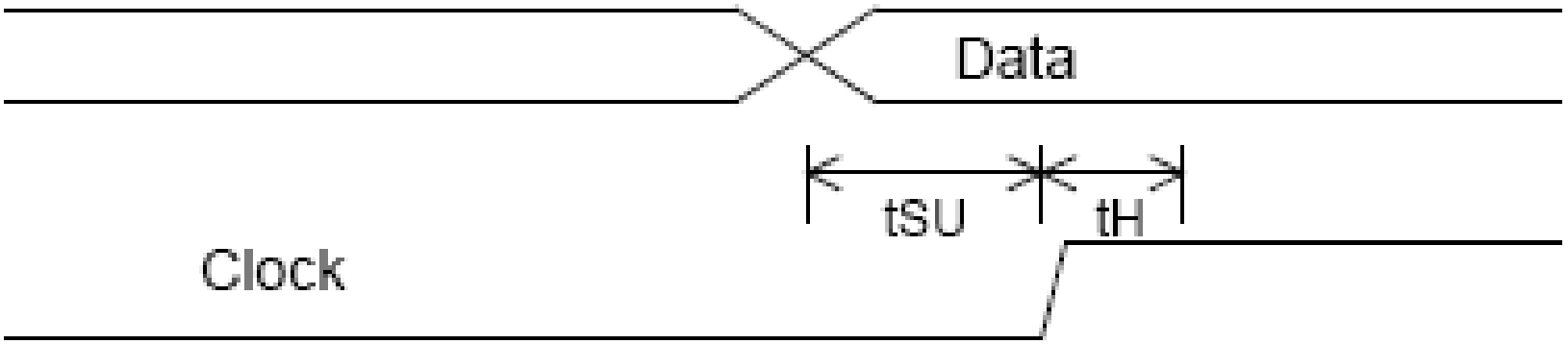
3 、检验延迟是否满足设计约束的要求。

时序分析基本概念

- 建立时间（ setup time ）
- 保持时间（ hold time ）
- 时钟到输出延迟（ clock to output time ）
- 时钟偏斜（ clock skew ）
- 时钟抖动（ jitter ）

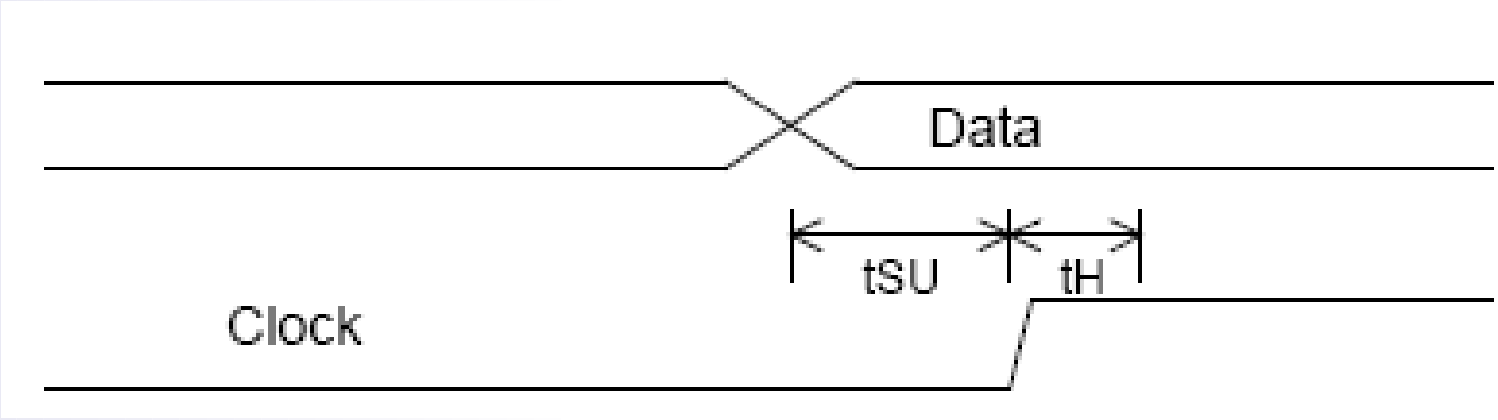
建立时间 t_{SU} （ setup time ）

- **触发器的时钟信号上升沿到来以前，数据稳定不变的时间**。输入信号应提前时钟上升沿 （假设上升沿有效） T 时间到达芯片，这个 T 就是建立时间 Setup time. 如不满足 setup time, 这个数据就不能被这一时钟打入触发器，只有在下一个时钟上升沿，数据才能被打入触发器。



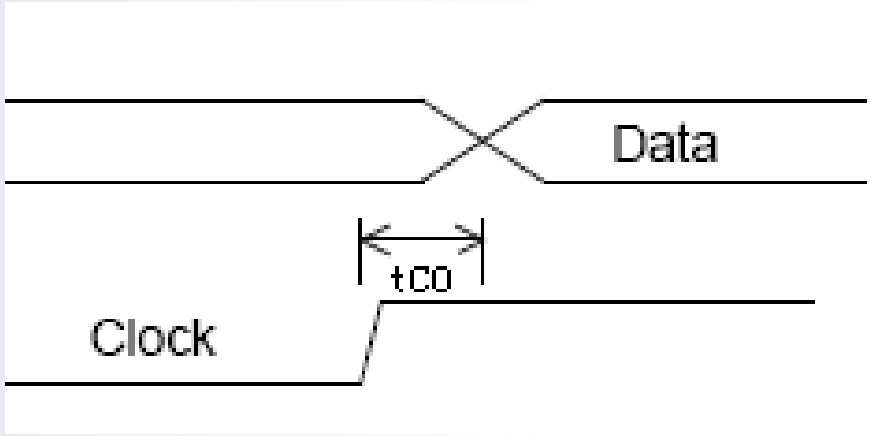
保持时间 t_H （ hold time ）

- **保持时间是指触发器的时钟信号上升沿到来以后，数据稳定不变的时间。**如果 hold time 不够，数据同样不能被打入触发器 。

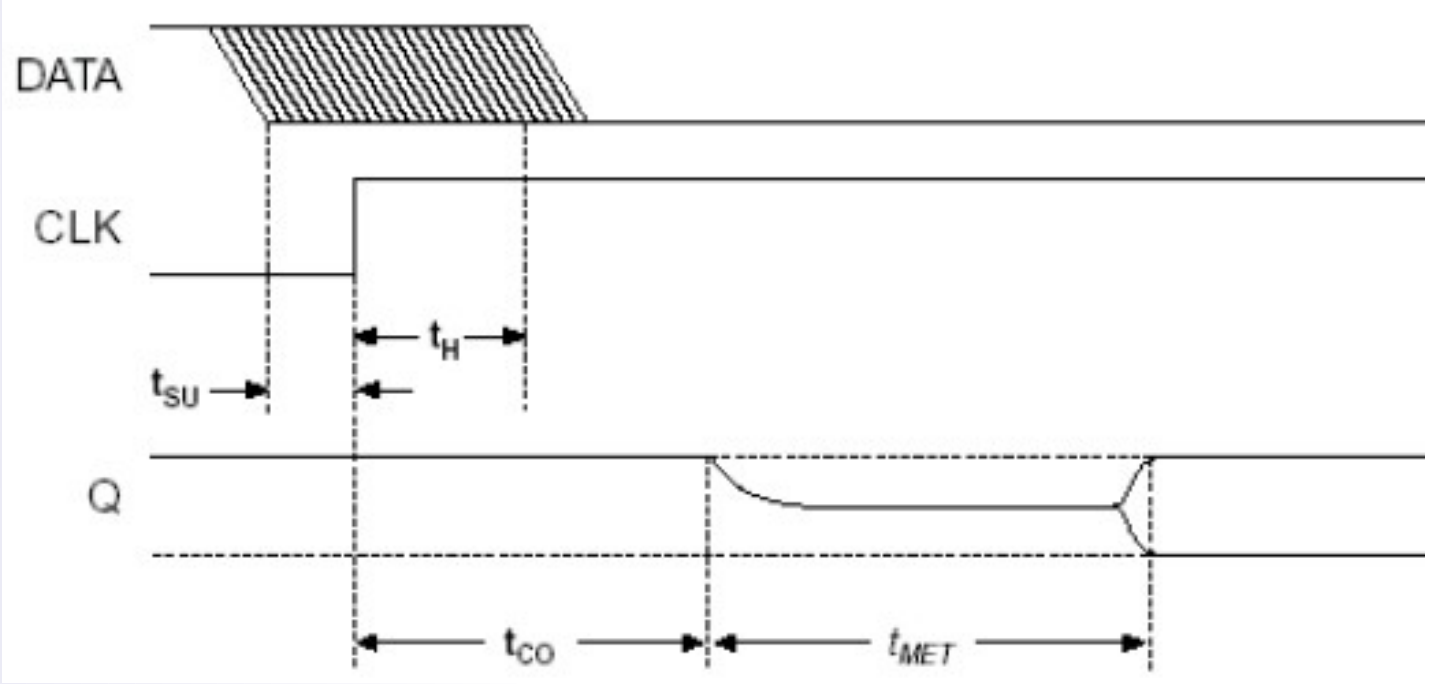


时钟到输出延迟 t_{CO} （clock to output time）

- 从时钟信号有效沿到数据有效的时间间隔

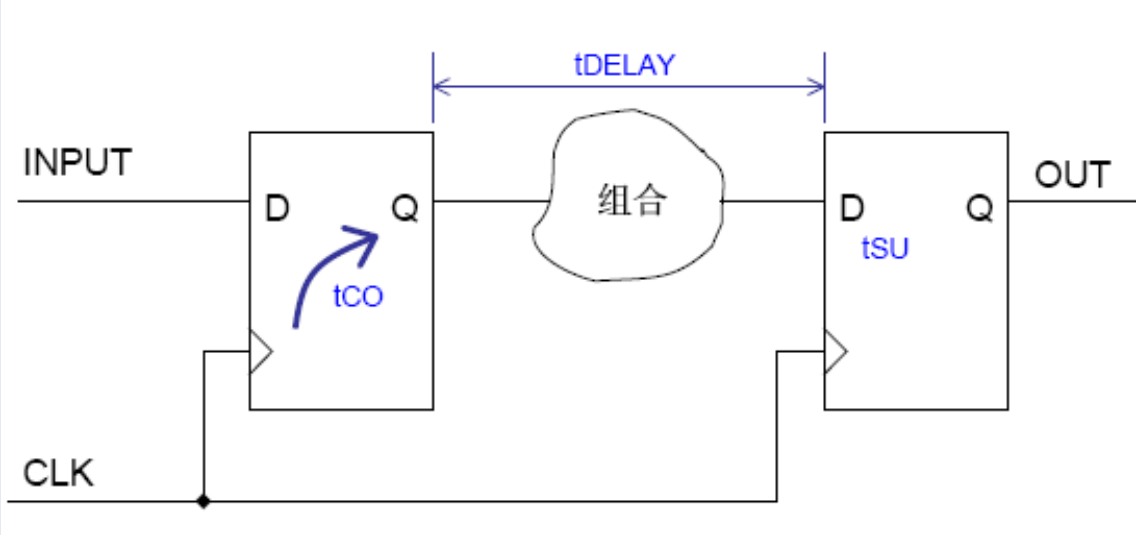


- 不满足建立 / 保持时间，可能出现亚稳态



t_{MET}—setting time，亚稳态到稳态的时间，与工艺无关

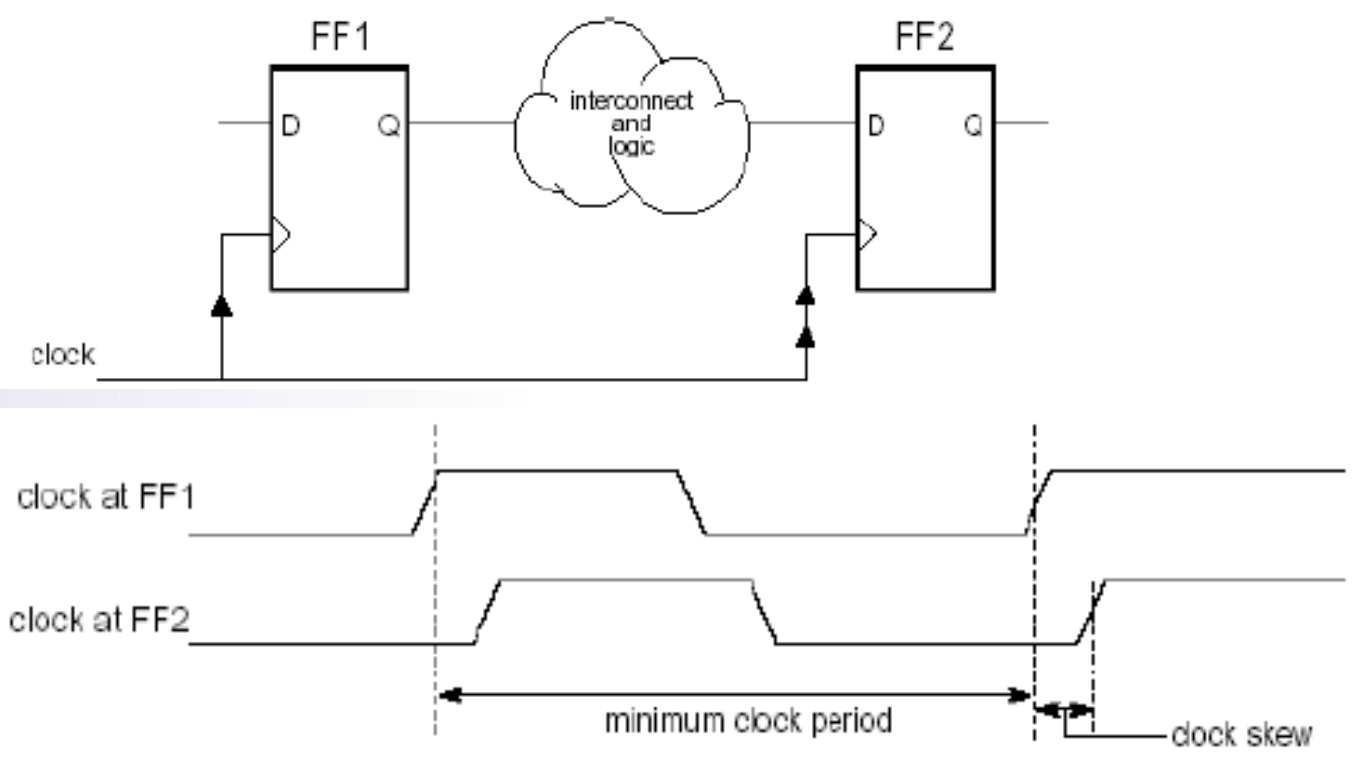
最小周期 T



$$T=t_{CO}+t_{DELAY}+t_{SU}$$

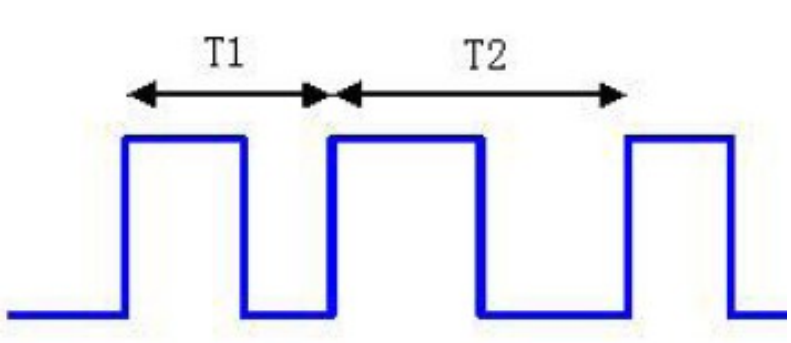
时钟偏斜（ clock skew ）

- 时钟偏斜指的是同一个时钟信号到达两个不同寄存器之间的时间差值
- 时钟偏斜永远存在，到一定程度就会严重影响电路的时序



时钟抖动（ jitter ）

- 所谓抖动，就是指两个时钟周期之间存在的差值，这个误差是在时钟发生器内部产生的，和晶振或者 PLL 内部电路有关，布线对其没有影响



jitter=T2-T1

STA 的过程

- STA 分三步走：

1、将设计打散成一个一个的 timing path

2、计算每条 path 的延迟

3、检验延迟是否满足设计约束的要求。

时序分析基本概念

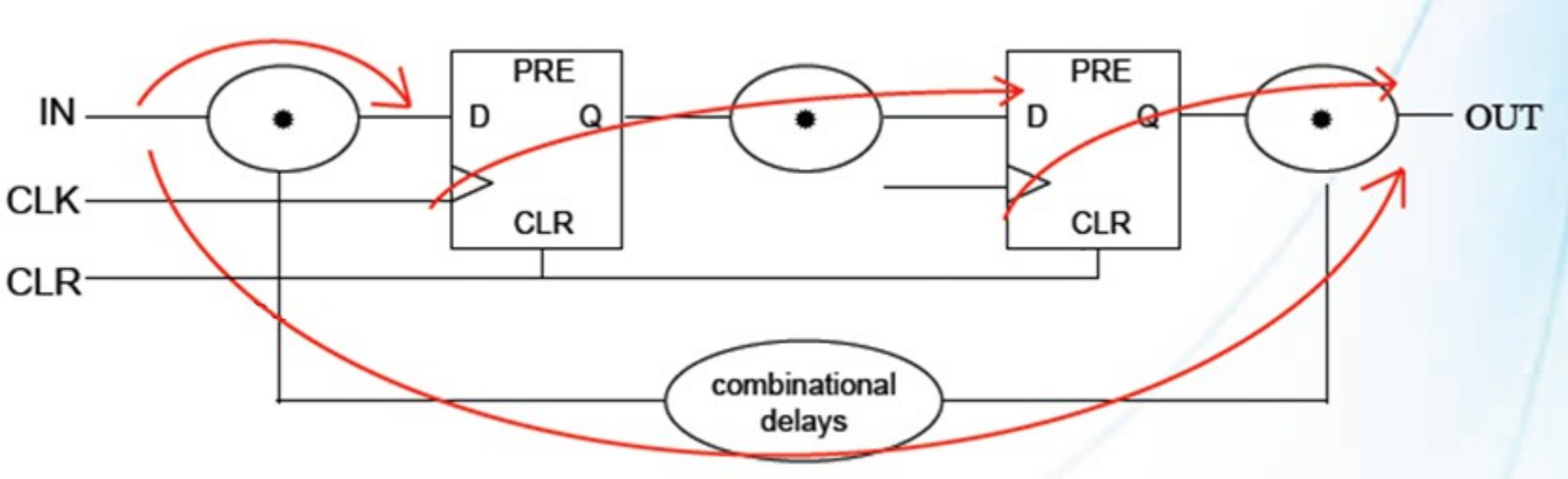
- 时序路径

① 从输入端口到触发器的数据 D 端

② 从触发器的时钟 clk 端到触发器的数据 D 端

③ 从触发器的时钟 clk 端到输出端口

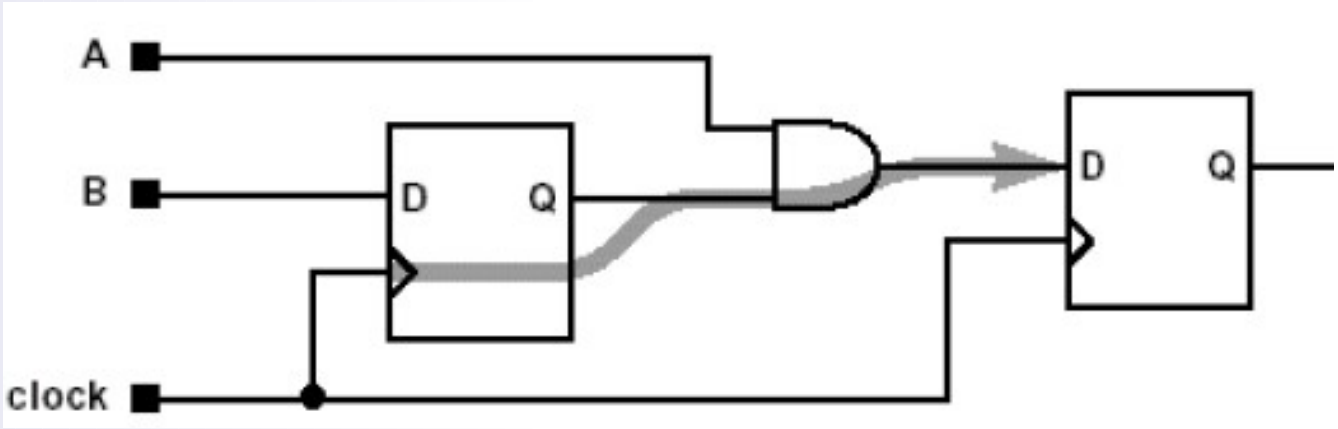
④ 从输入端口到输出端口

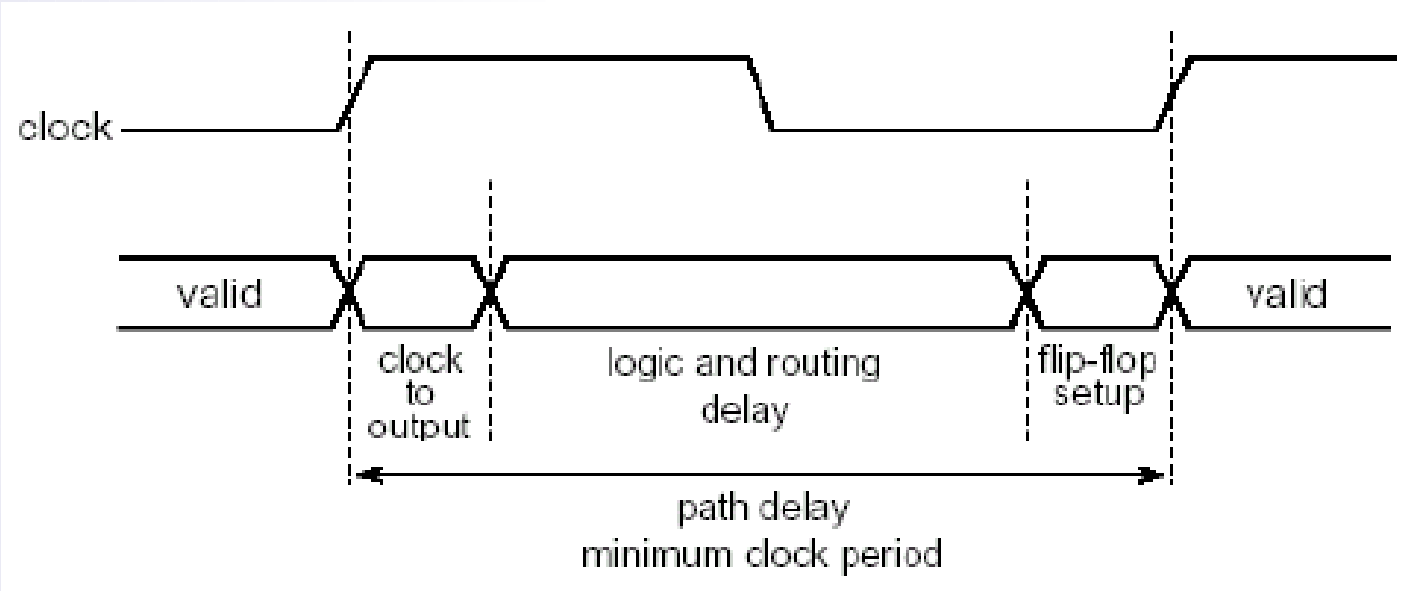


时序分析常用路径

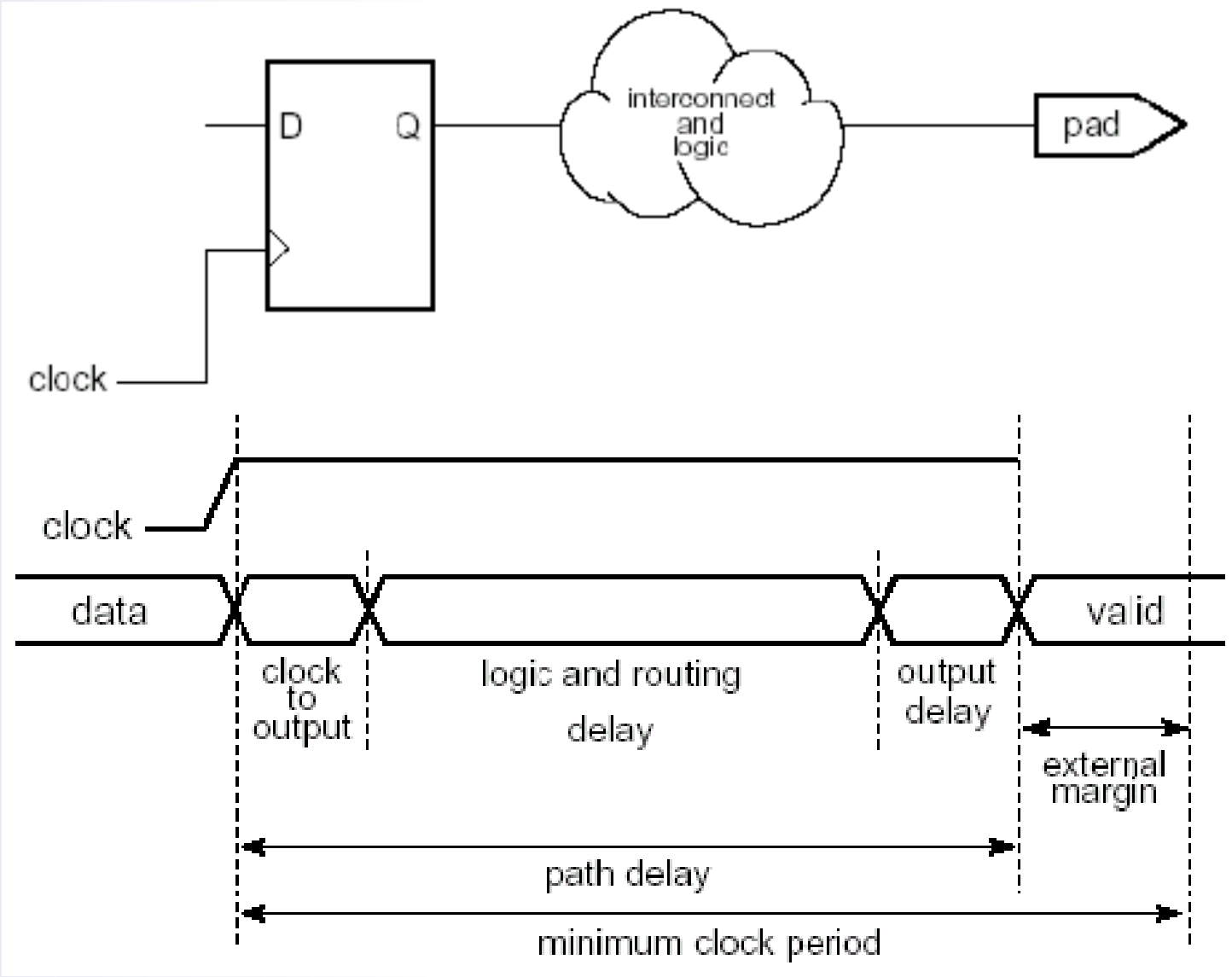
- 时钟到建立　clock to setup path
- 时钟到管脚　clock to pad path
- 结束于时钟引脚　paths ending at clock pin of flip-flops
- 管脚到管脚　pad to pad
- 管脚到建立　pad to setup

时钟到建立　clock to setup path

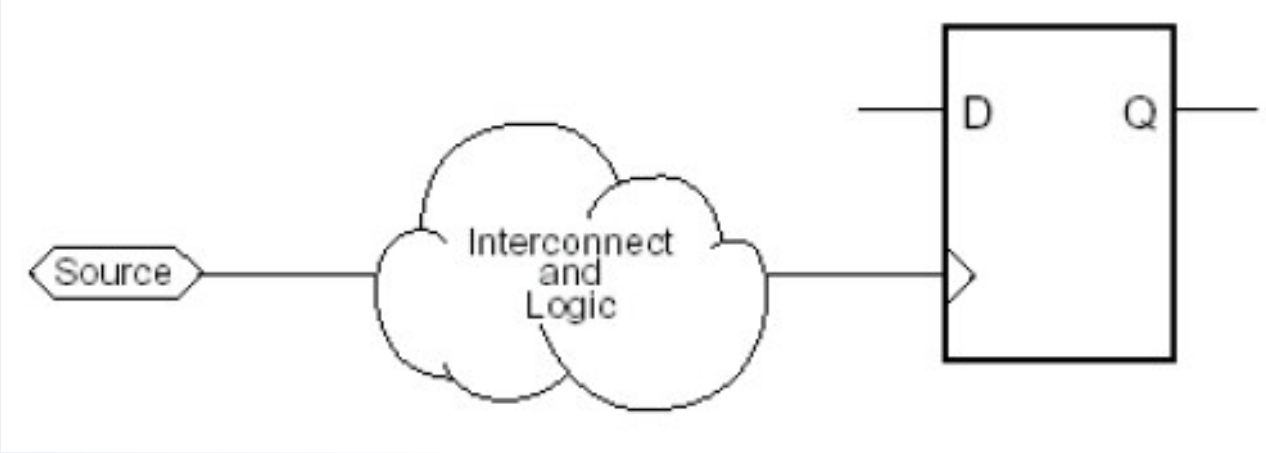


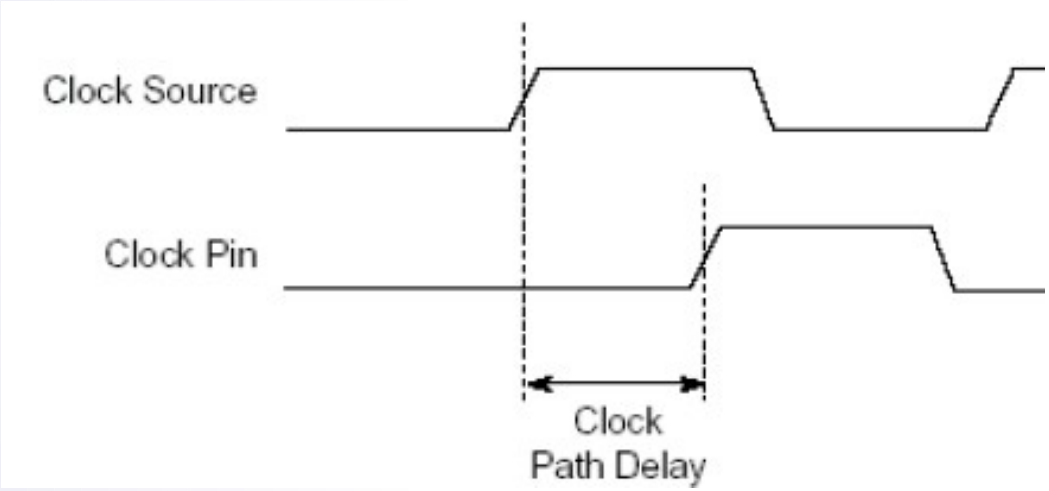


时钟到管脚　clock to pad path

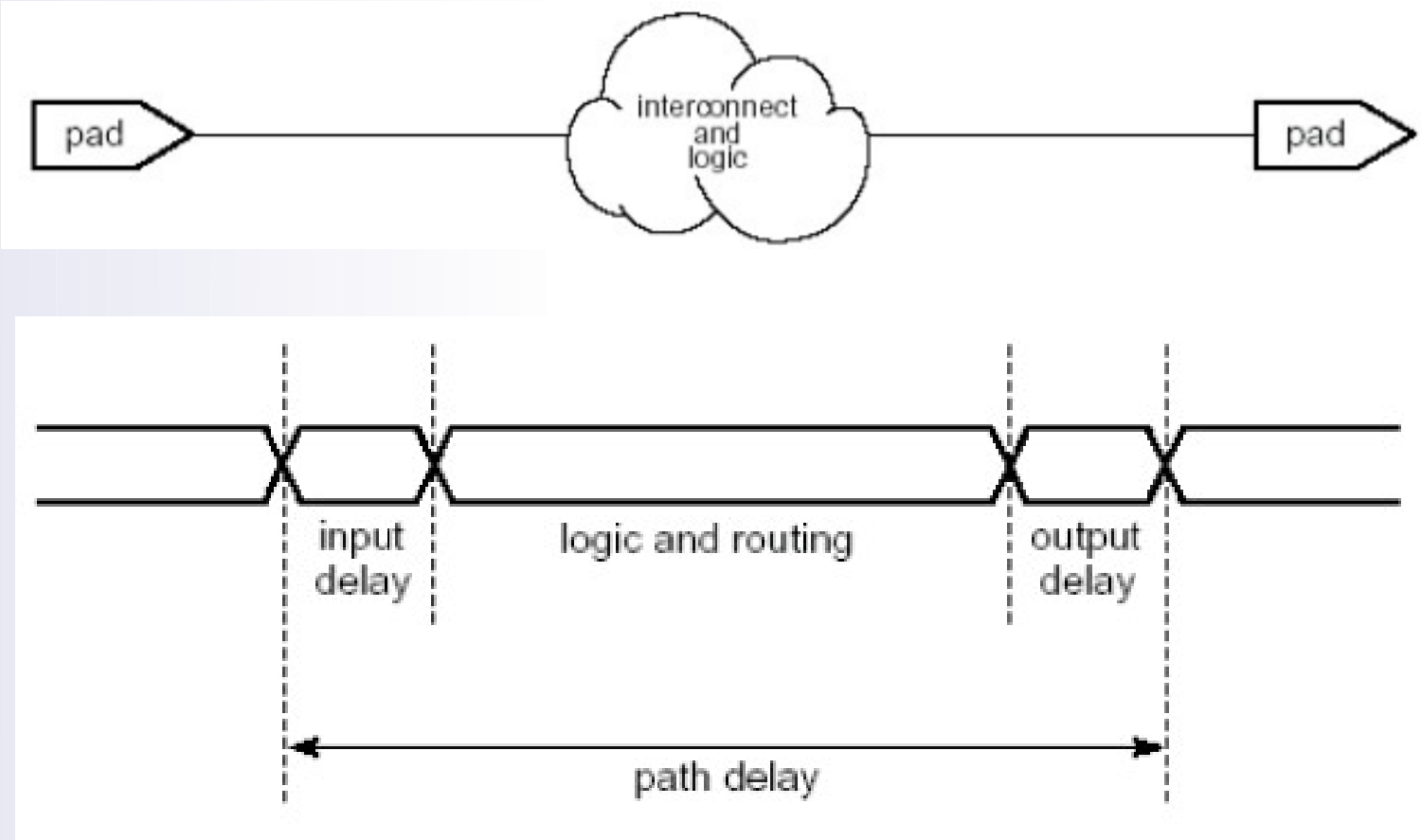


结束于时钟引脚 ending at clock pin of F-F

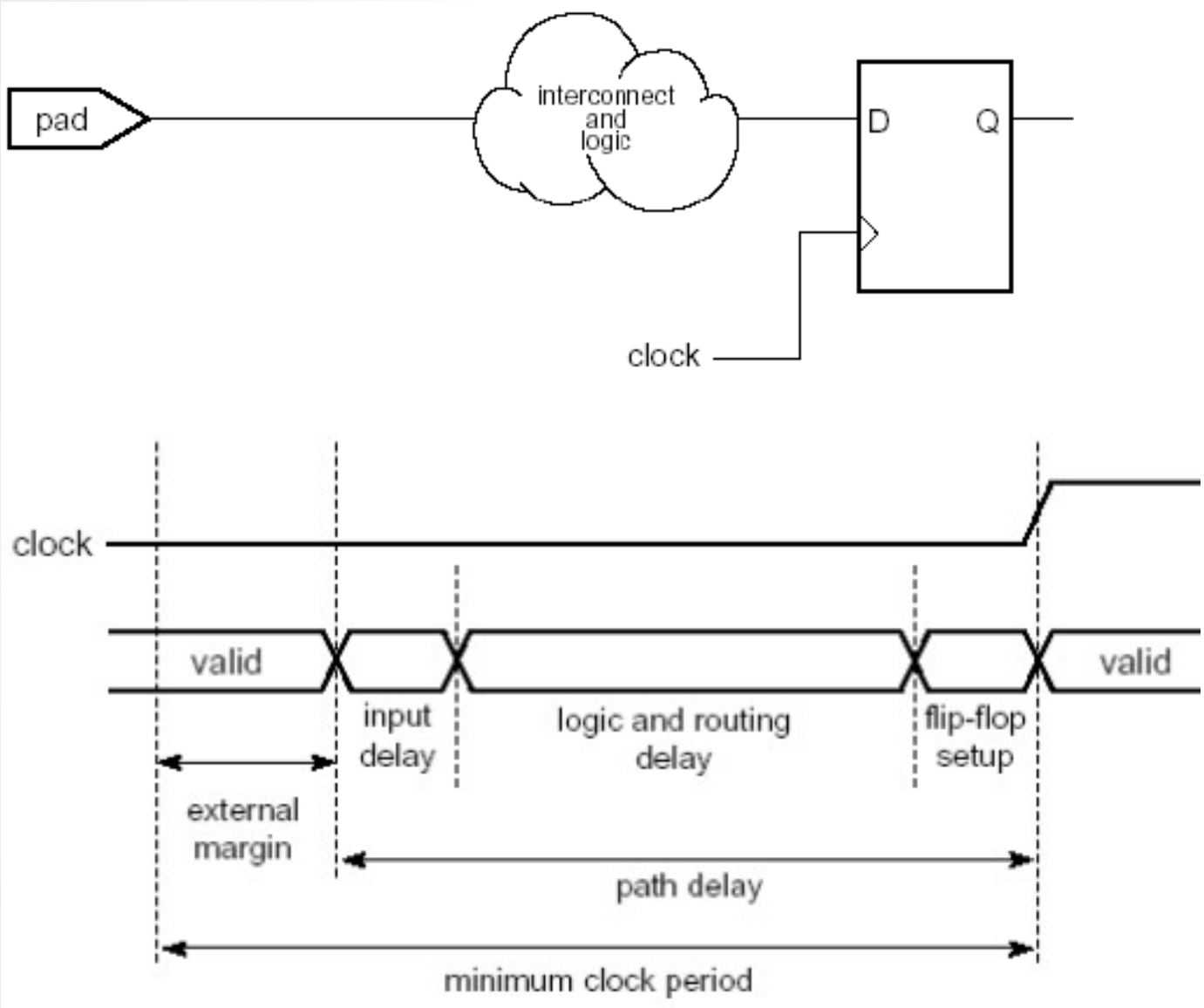




管脚到管脚　pad to pad



管脚到建立　pad to setup



时序分析基本概念

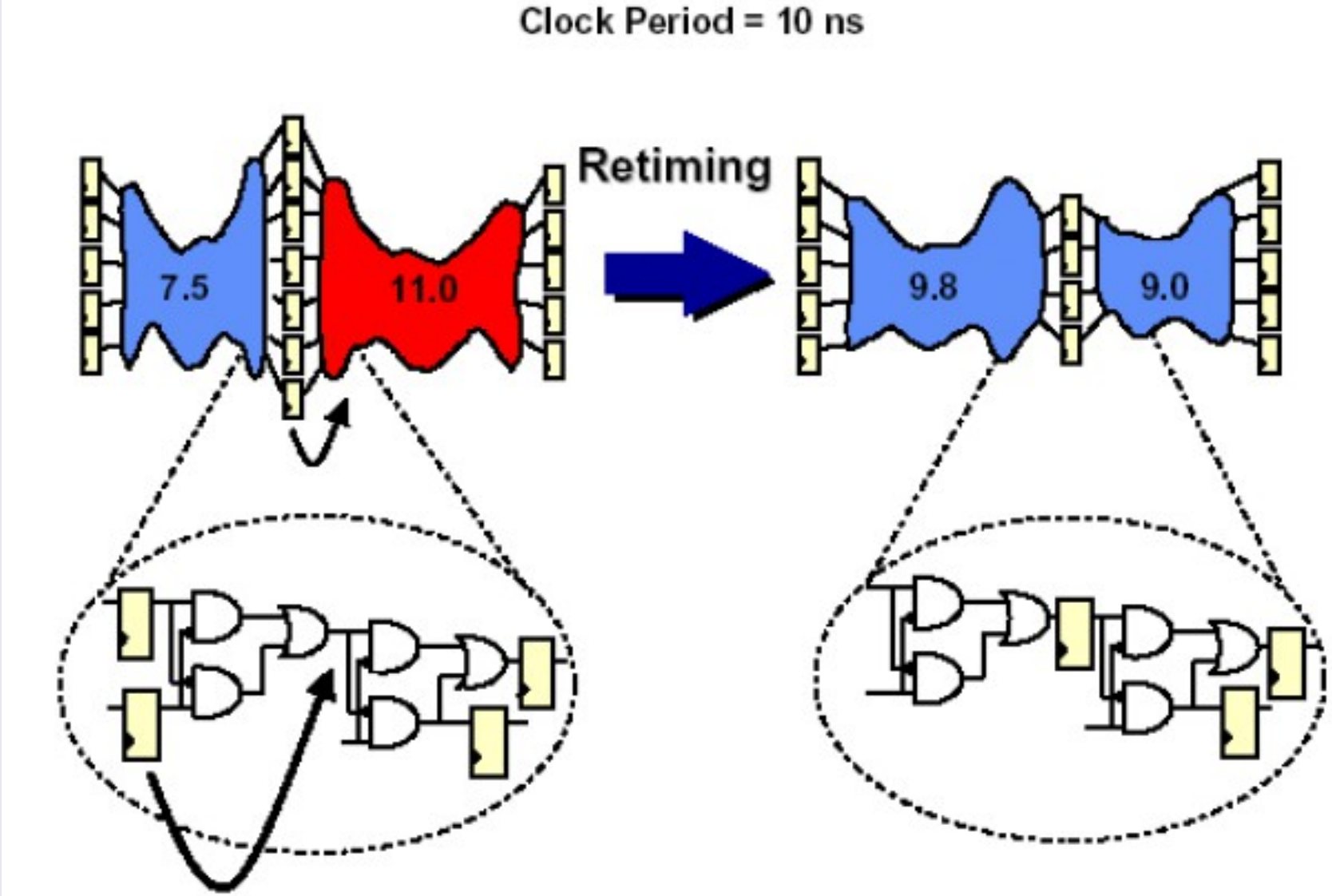
- 关键路径

  关键路径通常是指同步逻辑电路中，组合逻辑时延最大的路径。也就是说**关键路径是对设计能起决定性影响的时序路径**。

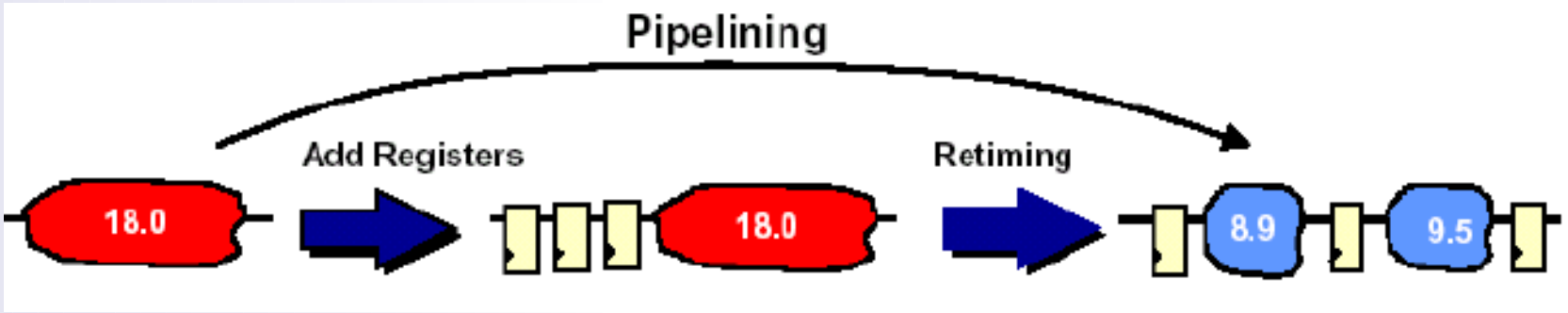
  静态时序分析可以找出逻辑电路的关键路径，通过查看时序分析报告，可以确定关键路径

  常用优化方法： Retiming 、 Pipeline

时序优化方法 -Ritiming



时序优化方法 -Pipeline



主流工具

- Synopsys 公司的 PrimeTime 主要用于全芯片的 IC 设计， PrimeTime 是业界最流行的分析工具
- 各 FPGA 厂商的工具均提供静态时序分析功能， FPGA 的静态时序分析比 IC 简单

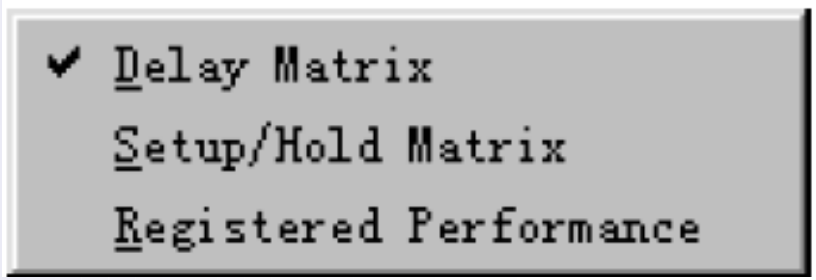
Timing Analyzer

- Altera 公司的 QuartusII 自带的静态时序分析工具，可以进行：

  时序路径的时延分析（ Delay Matrix ）

  建立 / 保持时间分析（ Setup/Hold Matrix ）

  同步逻辑性能（ Registered Performance ）

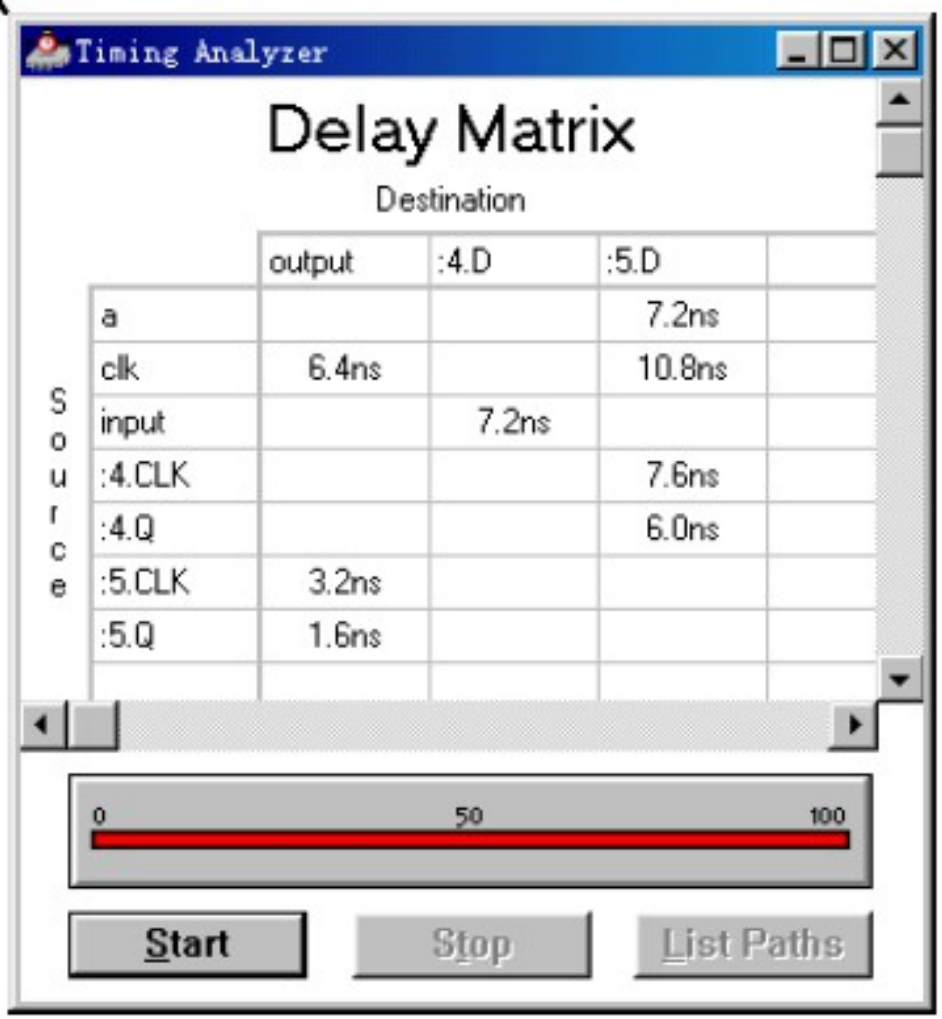


Timing Analyzer

Timing Analyzer – Delay Matrix

路径时延分析

可分析多个源和多个目标结点（node）之间的路径传播时延，

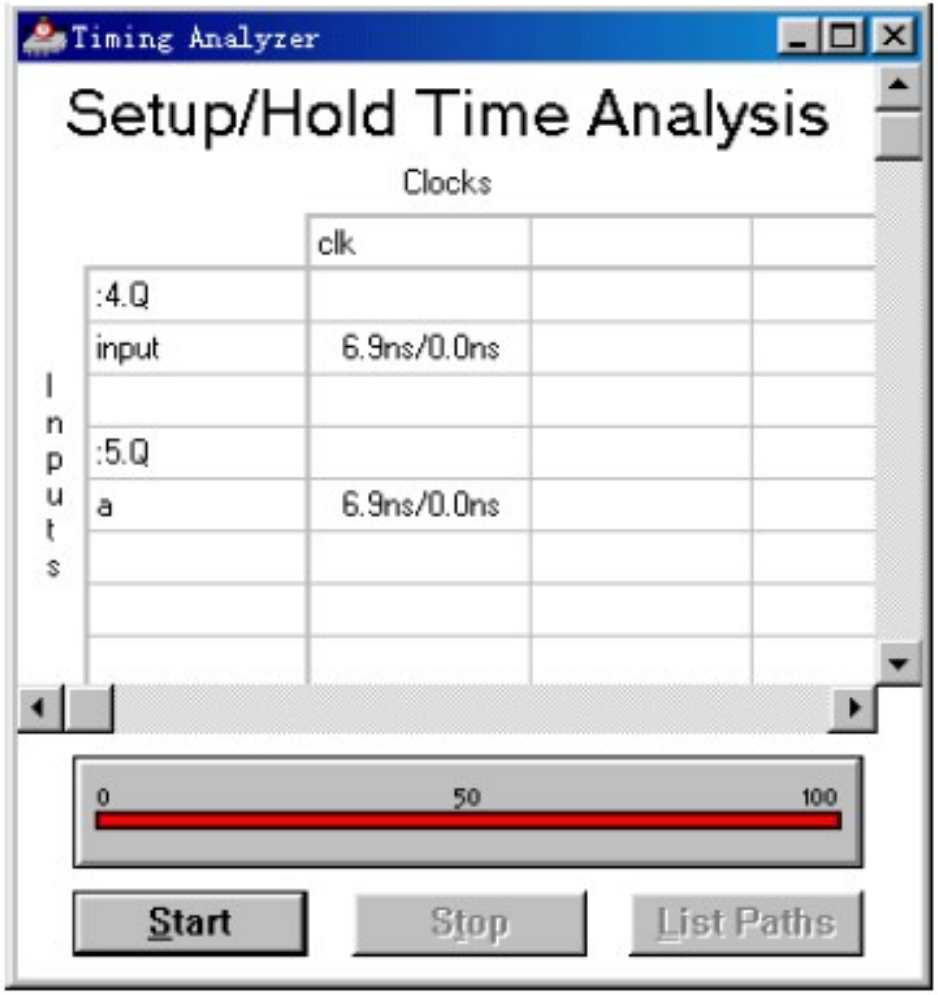


Timing Analyzer

Timing Analyzer – Setup/Hold

建立/保持时间分析

计算输入引脚到DFF的数据、时钟、时钟使能输入端的最小要求建立、保持时间



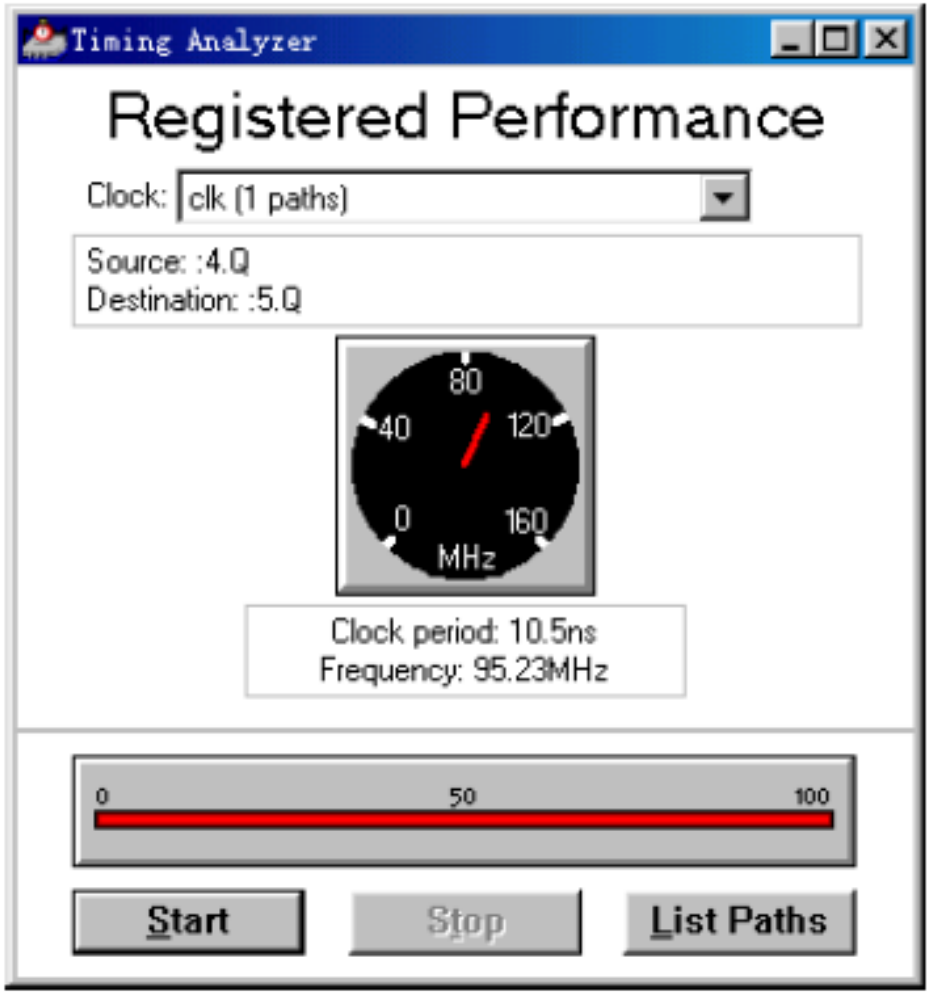
Timing Analyzer

Timing Analyzer – Registered Performance

同步逻辑性能分析

分析同步逻辑，确定限制性能的时延、最小时钟周期、最大时钟频率。

实际上包含了各内部寄存器的建立、保持时间分析。



Timing Analyzer

Timing Analyzer Summary				
	Type	Actual Time	From	To
1	Worst-case tsu	4.790 ns	reset	floating_cordic:f_core|atan_ro
2	Worst-case tco	9.526 ns	ctl_cordic:ctl_core|x_out[11]	x_out[11]
3	Worst-case th	-0.257 ns	reset	ctl_cordic:ctl_core|bs
4	Clock Setup: 'clk'	22.49 MHz (period = 44.468 ns)	floating_cordic:f_core|shift_right:shift_2|tmp[4]	ctl_cordic:ctl_core|xo[29]

PrimeTime

- PrimeTime 是 Synopsys 的静态时序分析工具，为业界标准，占据最大的市场份额
- PrimeTime 是数字 ASIC 设计的 sign-off 必选工具，受到所有 EDA 工具和 IC 厂家的支持
- FPGA 逻辑静态时序分析，仅用到 PrimeTime 的一小部分功能

Report 术语

- Arrival Time- 信号到达时间

  表示实际计算所得的信号到达逻辑电路中某一点的绝对时间，等于信号到达某条路径起点的时间加上信号在该条路径上的逻辑单元间传递延时的总和

- Required Arrival Time- 要求到达时间

  简称 RAT，表示要求信号在逻辑电路的某一特定点处的到达时间

- Slack- 余量

  表示在逻辑电路的某一特定点处要求到达时间与实际到达时间之间的差。 Slack 值表示该信号到达的太早或太晚

PT 过程

- PrimeTime 做 STA 分四步流程：

1、读入设计及库

2、约束设计

3、指定延迟计算信息

4、静态时序分析和报告

1、 建立设计环境
- 建立搜索路径（ search path ）和链接路径（ link path ）
- 读入设计和库
- 链接顶层设计
- 建立运作条件、连线负载模型、端口负载、驱动和传输时间

2、 说明时序声明（约束）
- 定义时钟周期、波形、不确定性（ uncertainty) 和滞后时间 (latency)
- 说明输入、输出端口的延时

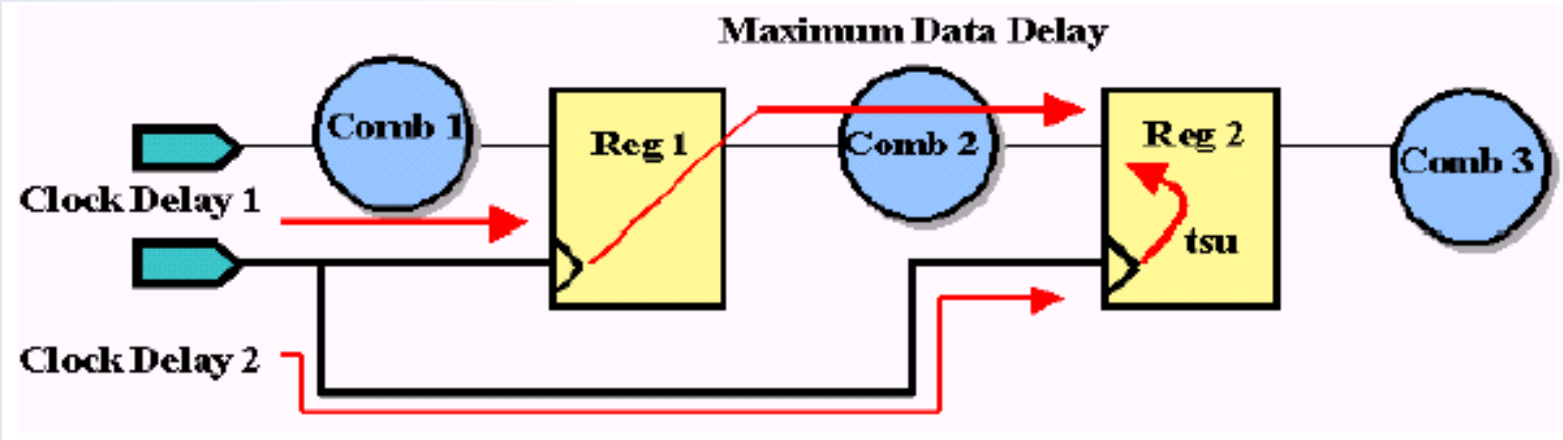
3、 说明时序例外情况（ timing exceptions ）
- 多周期路径（ multicycle paths ）
- 不合法路径（ false paths ）
- 说明最大和最小延时、路径分割（ path segmentation ）和失效弧（ disabled arcs ）

4、 进行分析和生成报告
- 检查时序
- 生成约束报告
- 生成路径时序报告

```
set search_path ". $QUARTUS_ROOTDIR/eda/synopsys/primetime/lib"

set link_path {* alt_vtl.db apex20ke_asynch_mem_lib.db
apex20ke_lvds_receiver_lib.db

apex20ke_cam_lib.db apex20ke_lvds_transmitter_lib.db apex20ke_io_lib.db
apex20ke_pll_lib.db

apex20ke_lcell_lib.db apex20ke_pterm_lib.db}

read_verilog
{ $QUARTUS_ROOTDIR/eda/synopsys/primetime/lib/apex20ke_camslice_pt.v }

read_verilog
{ $QUARTUS_ROOTDIR/eda/synopsys/primetime/lib/apex20ke_ramslice_pt.v }

read_verilog snug_pt.vo

current_design snug

link_design snug

read_sdf snug_v.sdo
create_clock “CLK” -period 4 –waveform {0 2}


check_timing

report_analysis_coverage

report_timing
```

PrimeTime

- 建立时间检查



clock delay1- clock delay2+max data path+t_{SU} clock period

Max data path 是寄存器的 t_{CO} 加上寄存器间的组合逻辑延迟

建立时间检查

- clock delay1=0ns
- clock delay2=0ns
- max data path=tco+path delay=1.449ns+0.258ns=1.707ns
- 若 T=4ns, 则 slack=4ns-1.707ns=2.293ns

建立时间检查

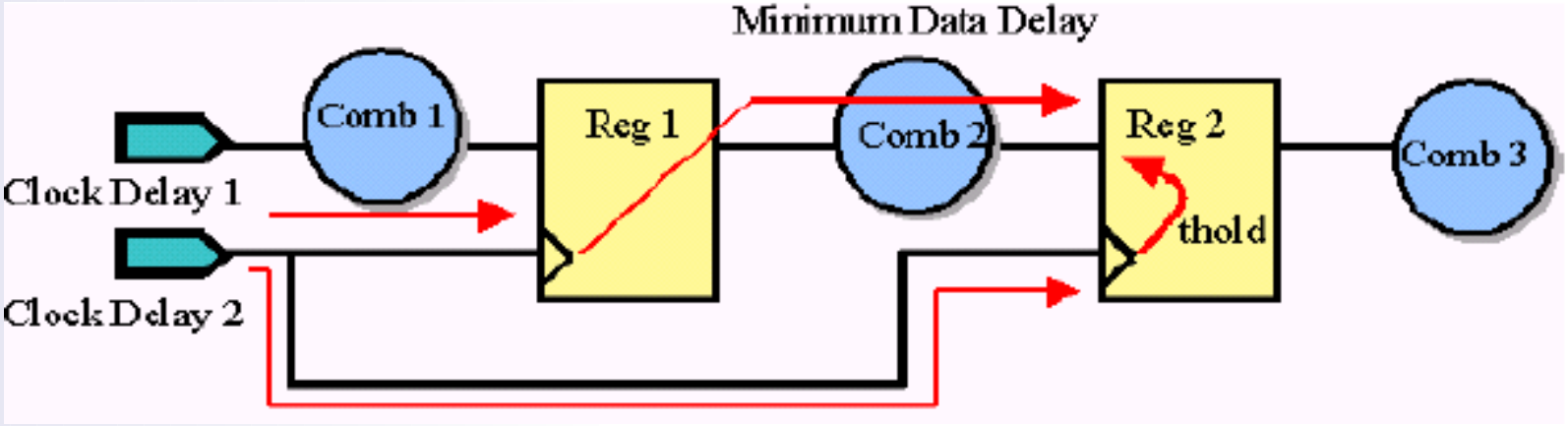
```
Startpoint: i_inst (rising edge-triggered flip-flop clocked by clk)
Endpoint: i_inst2 (rising edge-triggered flip-flop clocked by clk)
Path Group: clk
Path Type: max

Point                                          Incr       Path
-----------------------------------------------------------------
clock clk (rise edge)                        0.000      0.000
clock network delay (ideal)                  0.000      0.000
i_inst/clk (apex20ke_lcell)                  0.000      0.000 r
i_inst/regout (apex20ke_lcell)               1.449 *    1.449 r
i_inst2/datad (apex20ke_lcell)               0.258 *    1.707 r
data arrival time                                       1.707

clock clk (rise edge)                        4.000      4.000
clock network delay (ideal)                  0.000      4.000
i_inst2/clk (apex20ke_lcell)                            4.000 r
library setup time                           0.000      4.000
data required time                                      4.000
-----------------------------------------------------------------
data required time                                      4.000
data arrival time                                      -1.707
-----------------------------------------------------------------
slack (MET)                                             2.293
```

MET：满足建立时间

VIOLATED：不满足

PrimeTime

- 保持时间检查



clock delay1-clock delay2+min data path -t_H 0

保持时间检查

- clock delay1=0ns
- clock delay2=0ns
- min data path=tco+path delay=1.449ns+0.258ns=1.707ns
- intrinsic hold time=1.284ns
- 则 slack=1.707ns-1.284ns=0.493ns

保持时间检查

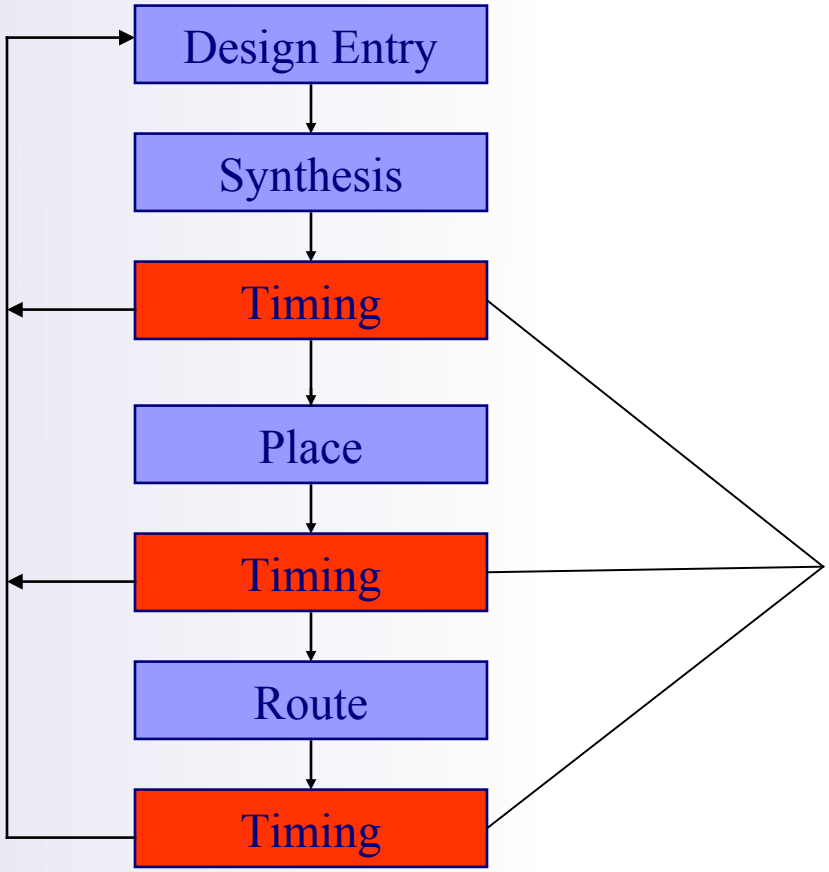
```
Startpoint: i_inst (rising edge-triggered flip-flop clocked by clk)
Endpoint: i_inst2 (rising edge-triggered flip-flop clocked by clk)
Path Group: clk
Path Type: min

Point                                          Incr       Path
-----------------------------------------------------------------
clock clk (rise edge)                        0.000      0.000
clock network delay (ideal)                  0.000      0.000
i_inst/clk (apex20ke_lcell)                  0.000      0.000 r
i_inst/regout (apex20ke_lcell)               1.449 *    1.449 r
i_inst2/datad (apex20ke_lcell)               0.258 *    1.707 r
data arrival time                                       1.707

clock clk (rise edge)                        0.000      0.000
clock network delay (ideal)                  0.000      0.000
i_inst2/clk (apex20ke_lcell)                            0.000 r
library hold time                            1.284 *    1.284
data required time                                      1.284
-----------------------------------------------------------------
data required time                                      1.284
data arrival time                                      -1.707
-----------------------------------------------------------------
slack (MET)                                             0.423
```

MET：满足建立时间

VIOLATED：不满足

问题

- 三个阶段时序分析有何不同？



综合后 STA

- 建立时间不符合 - 重新设计
- 保持时间不符合 - 此处修改或布局后修改（根据大小）
- 采用的统计线载模型
- 时钟扇出和时钟翻转固定

布局后 STA

- 布局工具将关键单元彼此靠近放置用以最小化路径延迟
- 修改保持时间违例（或根据违例程度选择布线后修改）
- 插入了时钟树（ clock tree， CT ），改变了原有设计

布线后 STA

- 加入寄生电容和 RC 连线延迟
- 修正保持时间（插入缓冲器）
- 最接近实际情况

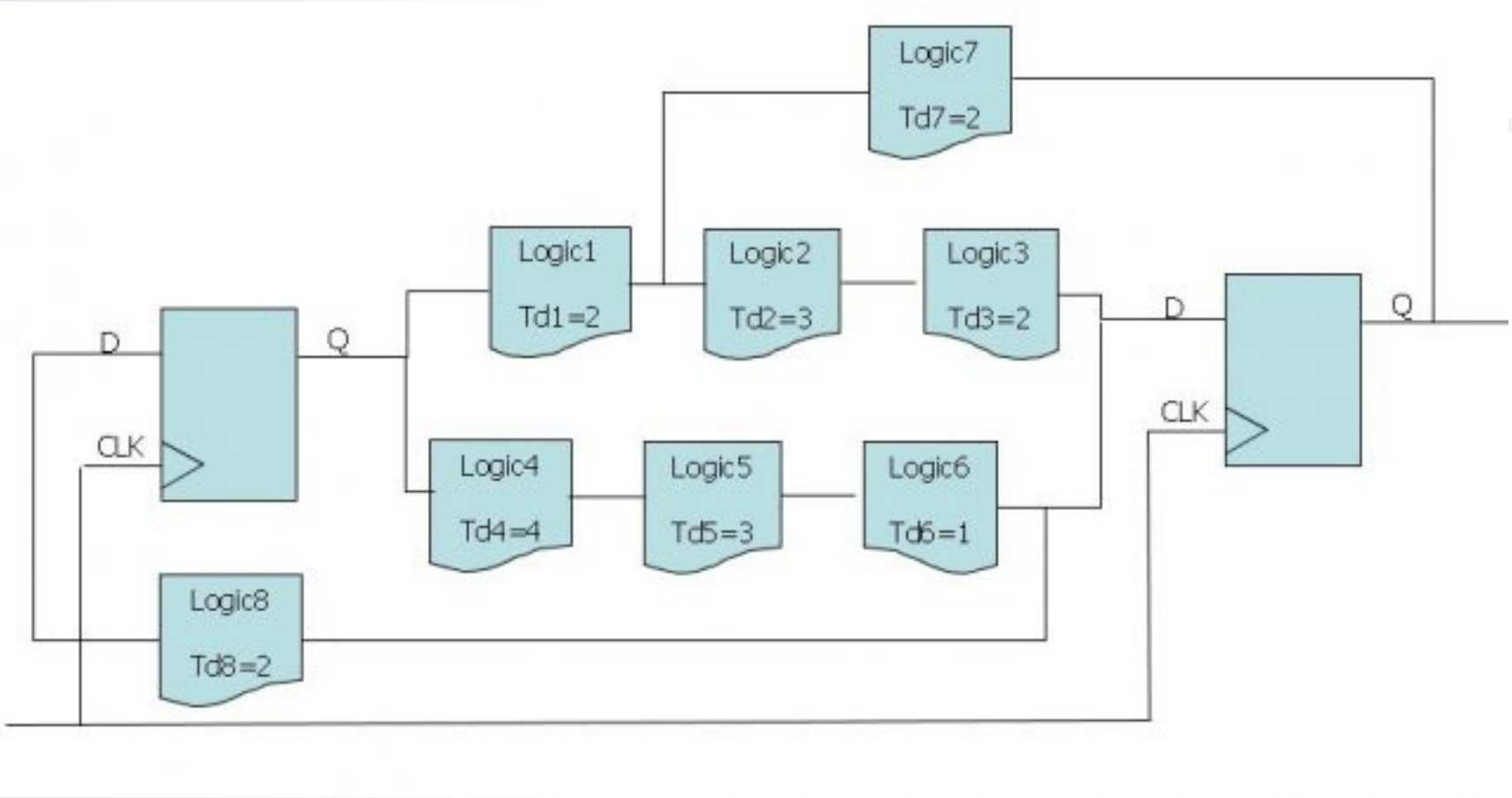
需要掌握的部分

- 流程图和相对应的文字说明
- 静态时序分析的概念、目的和作用
- 建立 / 保持时间的概念和约束条件的计算
- PrimeTime 的基本过程

补充题

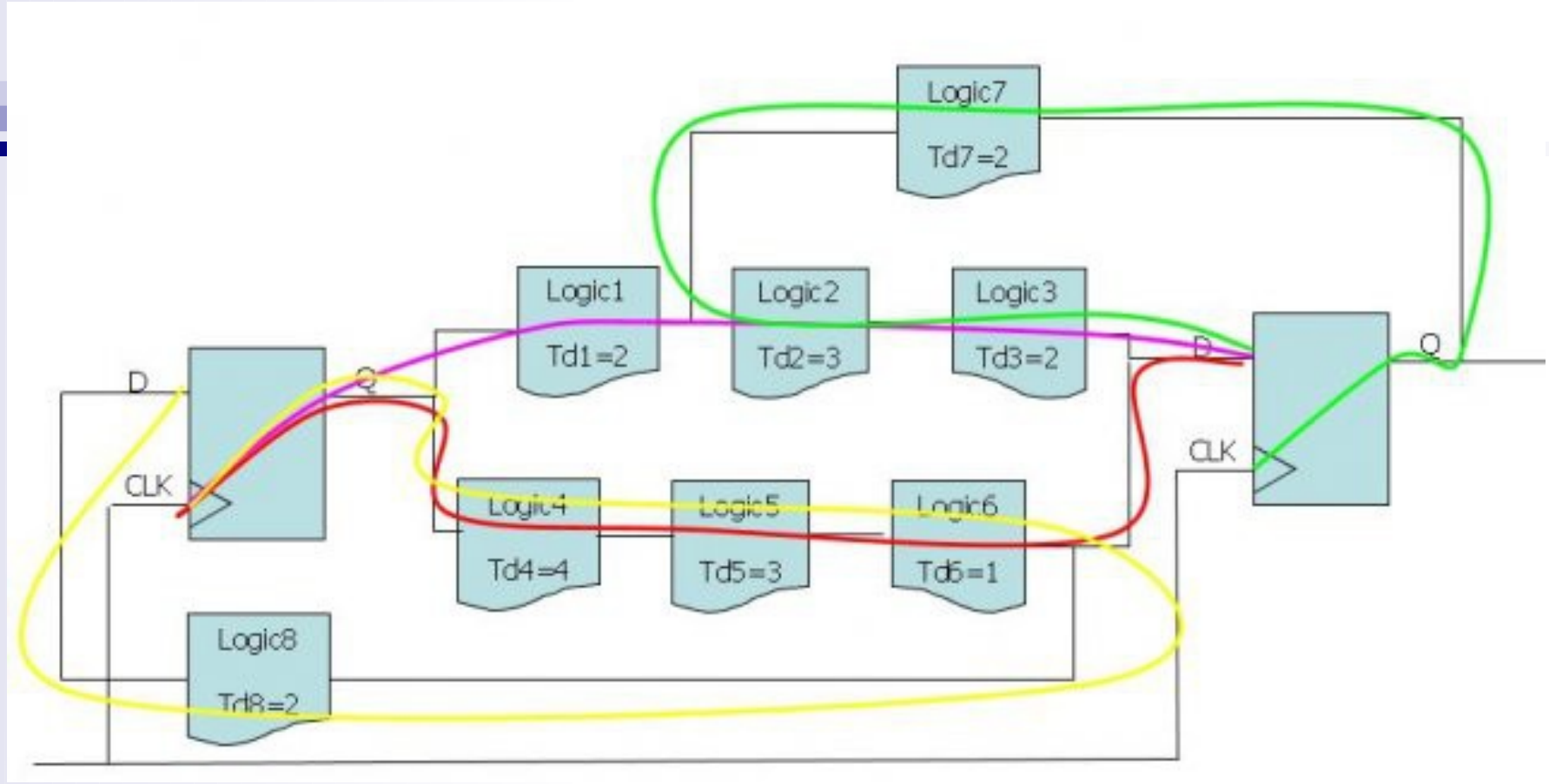
- 给定 setup time /hold time 的案例，要求算出最小时钟周期。
- 也可以给定周期和 setup time 和 hold time，计算时间裕度。

- 我们假设时钟周期是 20, 每个触发器的 cell 延迟是 1, 触发器的建立时间是 1, 保持时间是 0.5, 分析下列图中的建立时间和保持时间的 slack 。



设时钟周期是 20, 每个触发器的 cell 延迟是 1, 触发器的建立时间是 1, 保持时间是 0.5, 分析图中的建立时间和保持时间的 slack 。

- 看到设计，首先要分析路径，找出最长和最短路径，因为dc 的综合都是根据约束而得到最短和最长路径来进行器件选择的。
- 接下来将图中的所有路径标出。
- 因为没有前级（ input_delay) 和后级电路 (output_delay)，我们只分析图中给出的 路径



对于红色路径： Td=Tcell+Td4+Td5+Td6=1+4+3+1=9

对于黄色路径：

Td=Tcell+Td4+Td5+Td6+Td8=1+4+3+1+2=11

对于紫色路径： Td=Tcell+Td1+Td2+Td3=1+2+3+2=8

对于绿色路径： Td=Tcell+Td7+Td2+Td3=1+2+3+2=8

所以 Tlongest=11,Tshortest=8

对于 setup time 的 slack： Tclk-Tlongest-Tsetup=20-11-1=8

对于 hold time 的 slack ： Tshortest-Thold=8-0.5=7.5