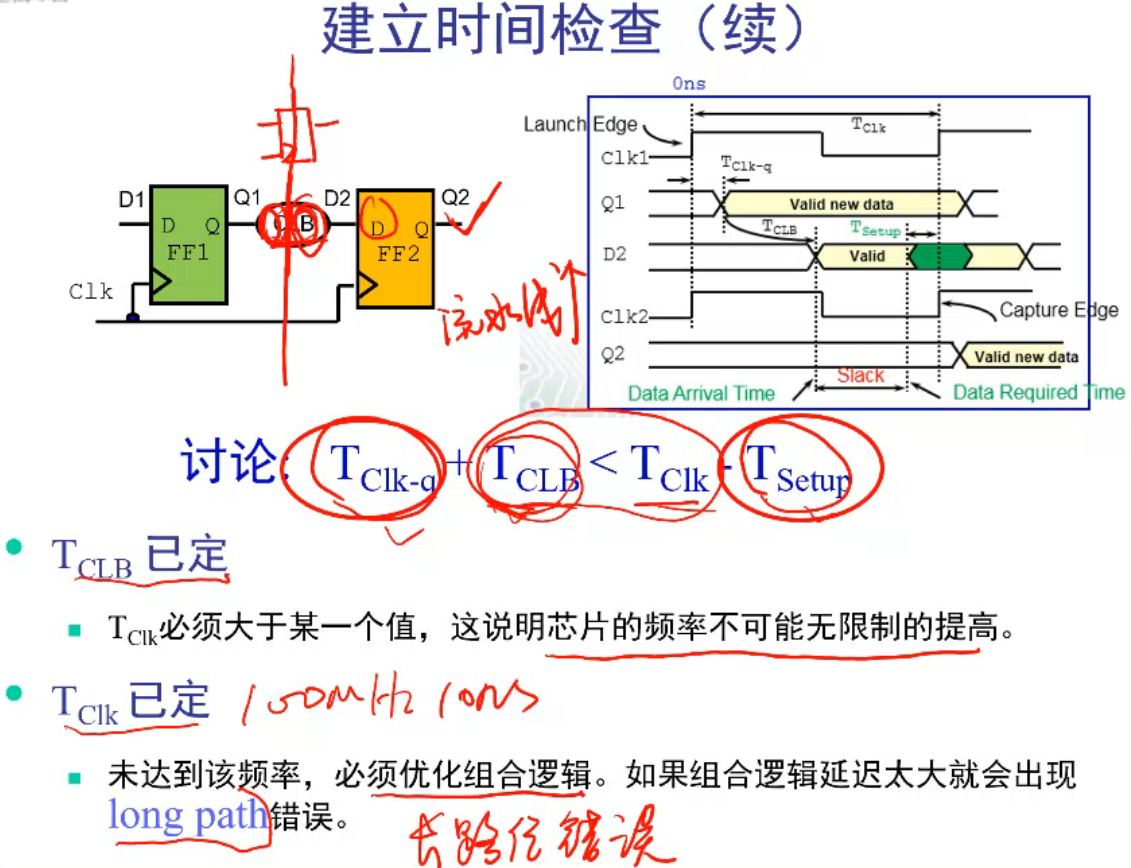
# 建立时间

保持时间

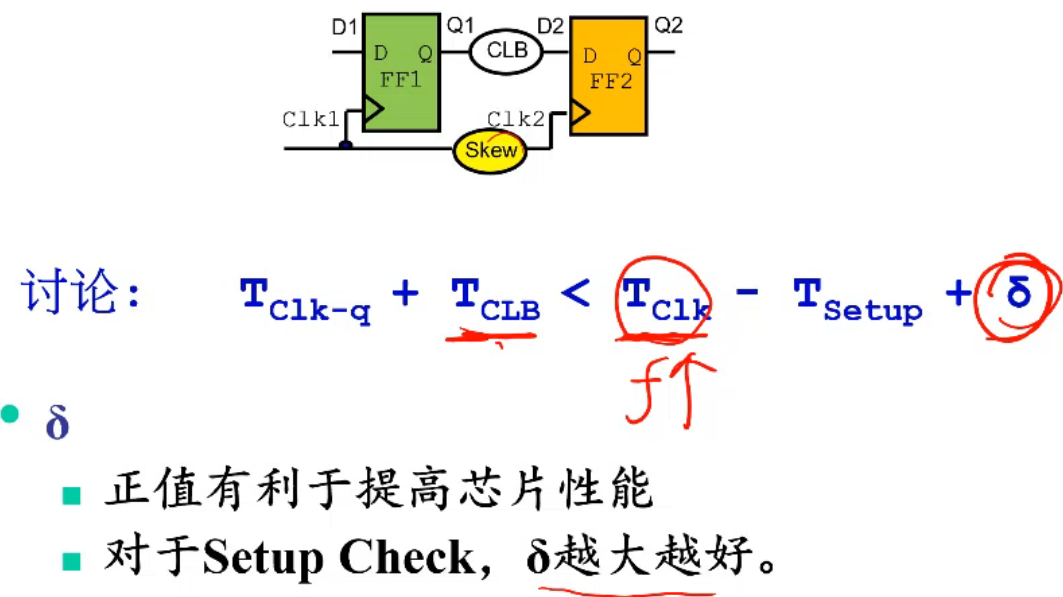
触发器Q端存在的延迟

建立时间

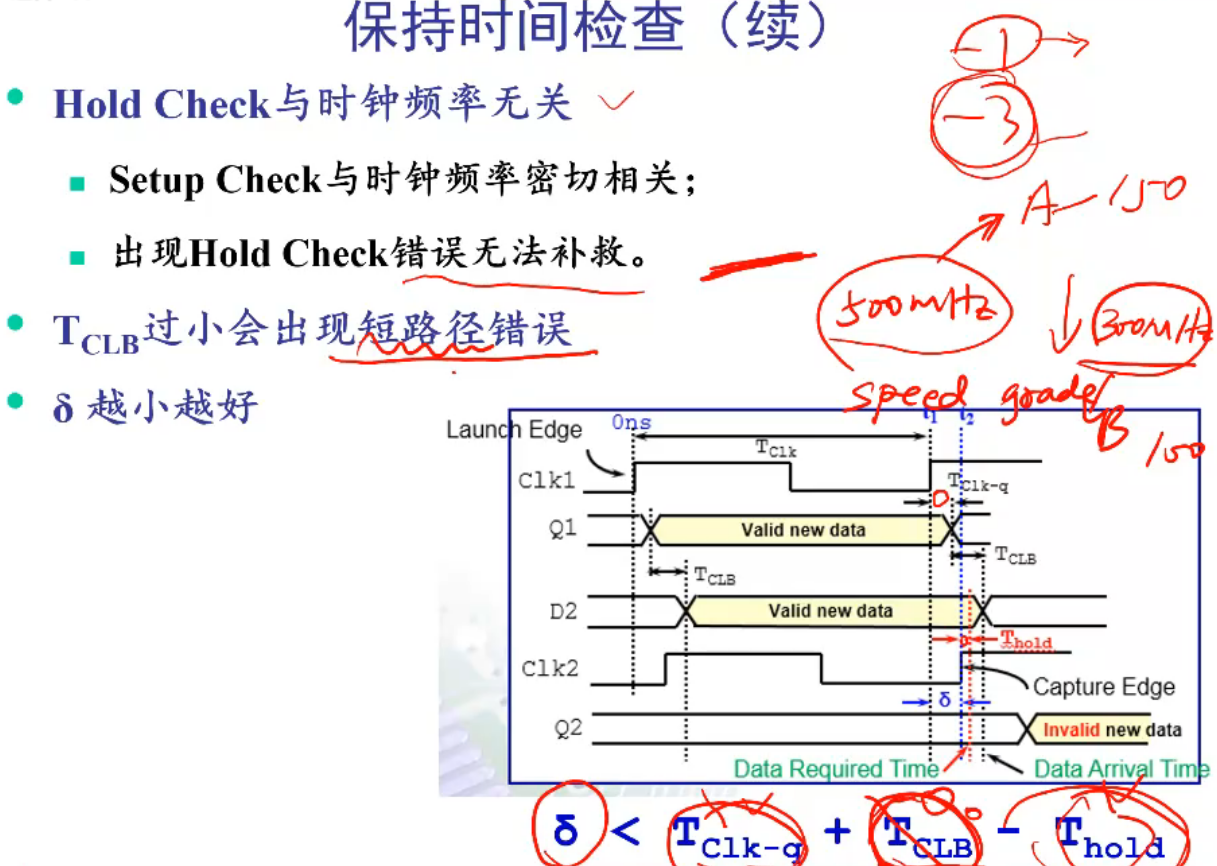
组合逻辑延迟



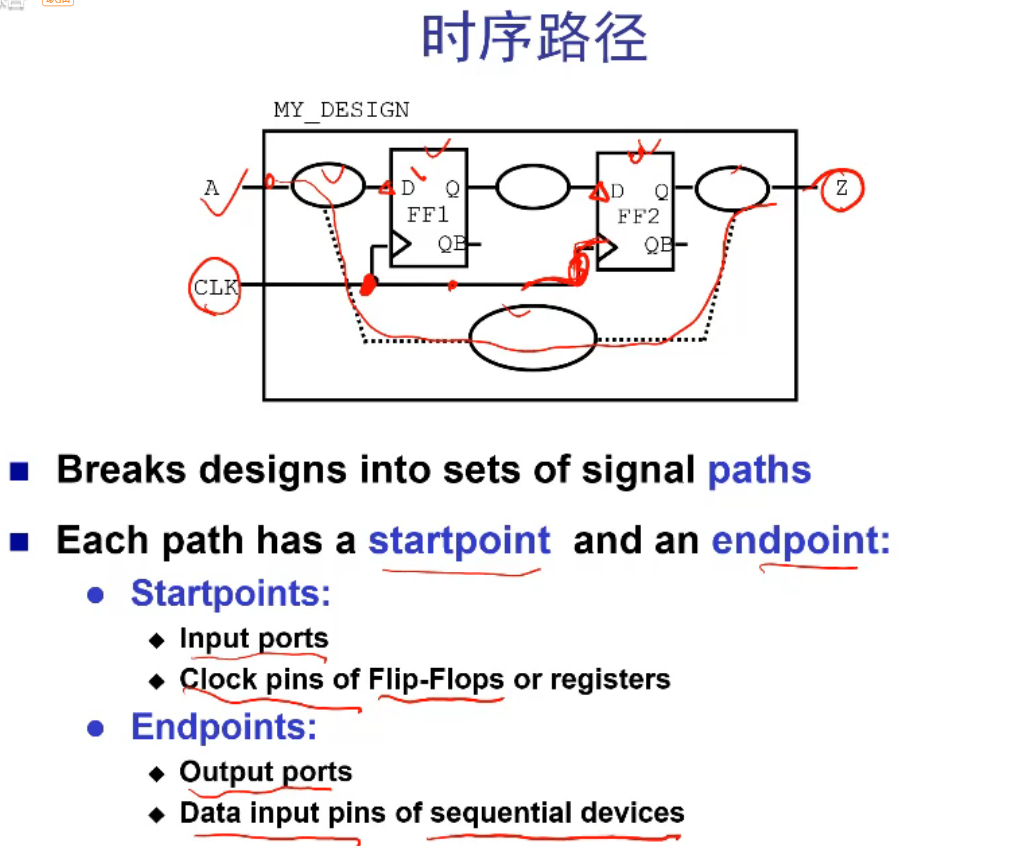
这就要求 Slack为一个正值， 和 基本上由工艺库决定，几十ps

假设clk有一定的时钟偏移Skew（可能正也可能负），偏移值为δ。

# 保持时间



# 时序路径



时序路径起点： ① 输入端口

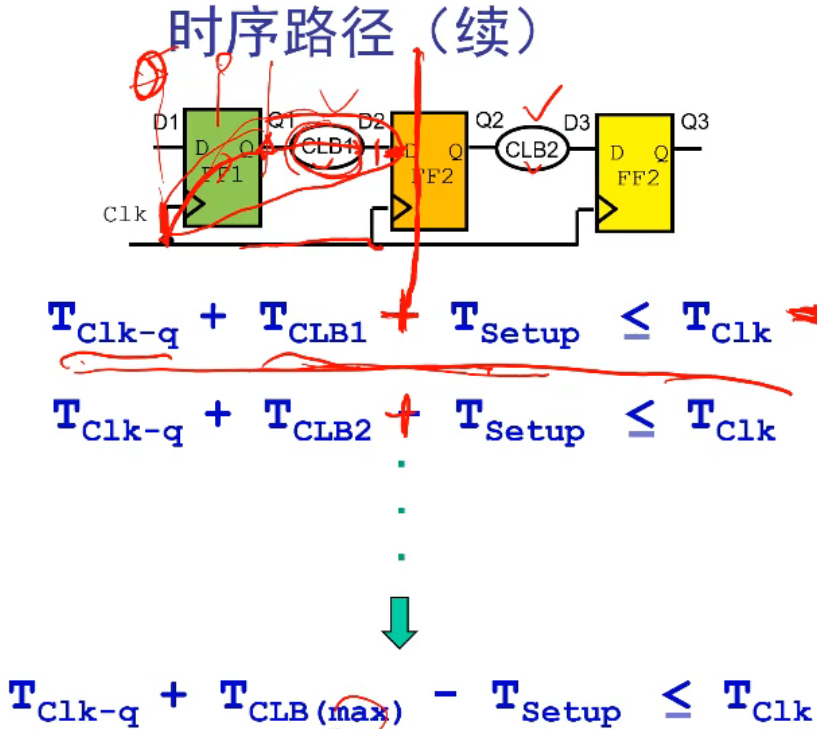
② 触发器或寄存器的时钟管脚

时序路径终点： ① 输出端口

② 时序器件的数据输入管脚

所以能得出有4种不同的时序路径

以clk为起点，D2为终点可以得到：



可以得出是影响频率的关键因素，对应的路径为关键路径。

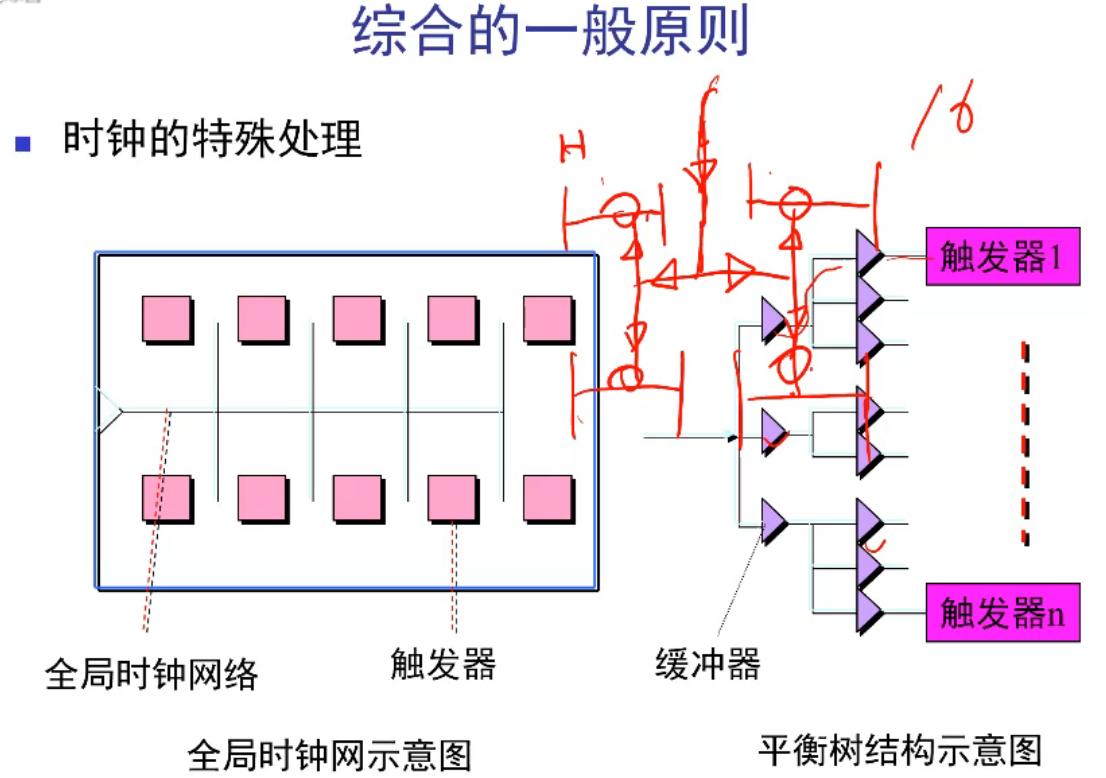
寻找时序路径时，不要忘记反馈路径

# 小结

1. 总共有4种不同的时序路径
2. 时钟频率受制于 检查以及关键路径（的路径）
3. clock skew对时钟频率有很大的影响
4. 检查独立于时钟频率，这个错误无法补救

# 5. 逻辑综合

## 5.1 时钟的特殊处理



为了能让一个clk驱动成百上千个触发器，需要平衡clk到各个触发器的延迟时间相等(利用buf)，常见的方法有平衡树法，H树法.

## 5.2 组合逻辑的注意事项

1. 组合逻辑的电路敏感表要完整

2. 在电平敏感的always块中，条件语句的分支语句、赋值语句要完整（否则系统自动判定为q = q），并避免出现 q = q; 避免产生锁存器

☆防止锁存器产生：

* 1. 对于case语句，添加分支使之完整。
  2. 加入default分支项：对于case语句，加入default分支项；对于if语句，加入else分支项
  3. 在if，case语句前加入缺省赋值语句( q = a; … )(推荐)
  4. 对于沿敏感的always块，只会生成触发器，赋值语句分支可以不完整，不会产生锁存器（触发器自带保持功能）

3. 避免在多个always块中对同一个变量赋值，防止输出冲突的产生

4. 通过改变RTL级代码风格控制资源共享（资源共享是RTL级模块中两个或两个以上的部分共享一组逻辑）

