时序检查

- 建立时间检查

  要保证Launch路径延时 <u>小于等于</u> Capture路径

$$T_{launch} + T_{ck2q} + T_{dp} < T_{capture} + T_{cycle} - T_{setup}$$

- 保持时间检查

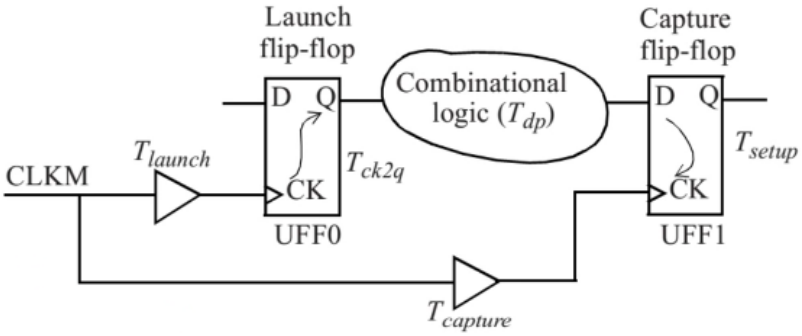
  要保证Launch路径时间 <u>超过</u> Capture路径

$$T_{launch} + T_{ck2q} + T_{dp} > T_{capture} + T_{hold}$$

> 静态时序分析的路径有两种起点和两种终点
>
> - 路径起点
>   - Cell/Pin_CLK
>   - Design/Input_Port
> - 路径终点
>   - Cell/Pin_D
>   - Design/Output_Port

1. **建立时间检查**（Setup Timing Check）
   - Reg-to-Reg模型

     > 两条网络
     >
     > - launch网络
     >
     > ***Launch 网络时序分析的计算结果为Data Arrival Time***
     >
     > - capture网络
     >
     >   ***Capture 网络时序分析的计算结果为 Slack***



The setup check can be mathematically expressed as:

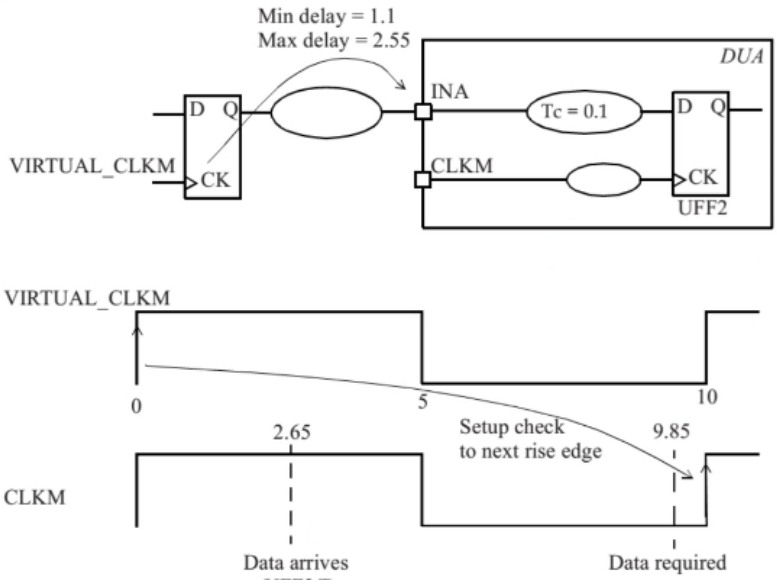
$$T_{launch} + T_{ck2q} + T_{dp} < T_{capture} + T_{cycle} - T_{setup}$$

要求数据路径延时要小于时钟偏斜与建立时间之和 —— Launch端延迟要小于Capture端采样时间点

对Tdp取最大值，若能满足公式，则其他Tdp都能满足时序，不会发生时序违例

- **InputPort-to-reg模型**

Input to Flip-flop Path



> 约束：
>
> ```
> # 约束一个虚拟时钟
> create_clock -name VIRTUAL_CLKM -period 10 -waveform {0 5}
> # 对虚拟时钟域约束Input Port Delay
> set_input_delay -clock VIRTUAL_CLKM -max 2.55 [get_ports INA]
> ```

Input Port Delay算在Launch网络中，用于计算Data Arrival Time

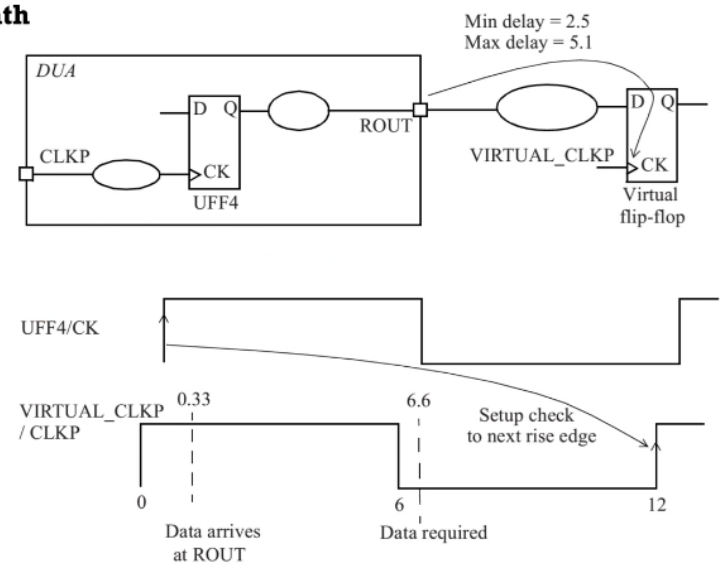
- **Reg-to-OutputPort**

Similar to the input port constraint described above, an output port can be constrained either with respect to a virtual clock, or an internal clock of the design, or an input clock port, or an output clock port.

To determine the delay of the last cell connected to the output port correctly, **one needs to specify the load on this port**. The output load is specified above using the **set_load** command.

> 除了约束OutputDelay，还需要对负载进行约束
>
> ```
> # 约束OutputDelay
> set_output_delay -clock VIRTUAL_CLKP -max 5.1 [get_ports ROUT]
> # 约束负载 set_load
> set_load 0.02 [get_ports ROUT]
> ```

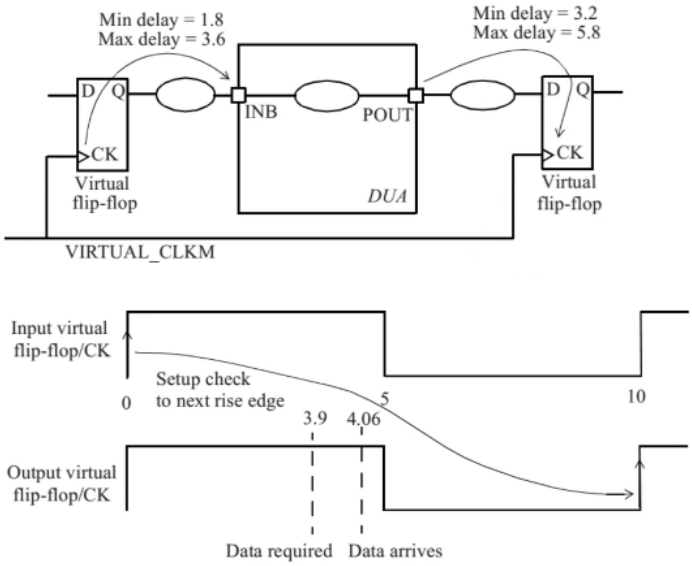
3、Flip-flop to Output Path



Output Delay具体体现在Capture路径中，包含了Capture DFF的Setup Time

- **Input-to-Output Path**

4、Input to Output Path

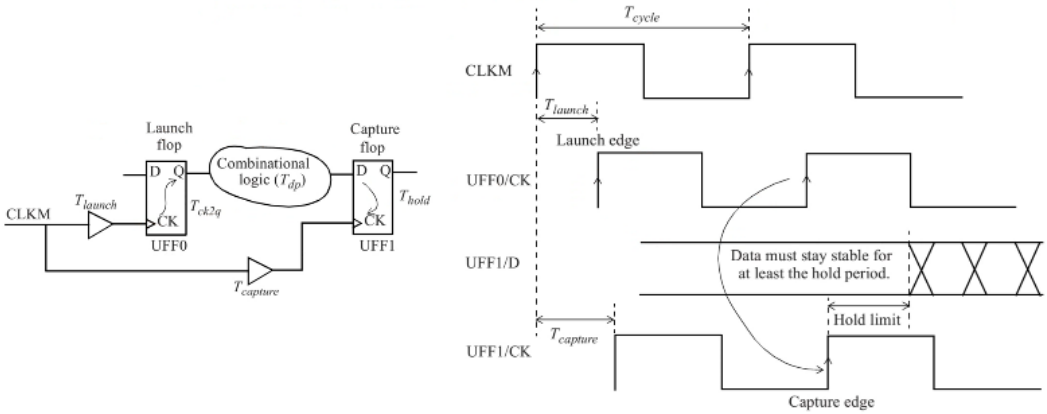


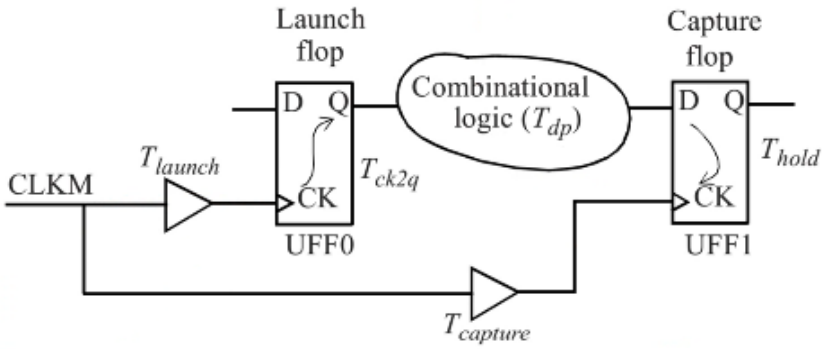
> 约束
>
> ```
> set_input_delay -clock VIRTUAL_CLKM -max 3.6 [get_ports INB]
> set_output_delay -clock VIRTUAL_CLKM -max 5.8 [get_ports ROUT]
> ```

2. **保持时间检查**（Hold Timing Check）

> Hold Timing Check 也是分Launch Path和Capture Path
>
> - 保持时间检查是从LaunchPath的有效时钟边沿到CapturePath的相同时钟边沿
> - 保持时间检查独立于时钟周期
> - 保持时间检查对CapturePath的有效时钟边沿执行

- **Reg-to-Reg**





The hold check can be mathematically expressed as:

$$T_{launch} + T_{ck2q} + T_{dp} > T_{capture} + T_{hold}$$

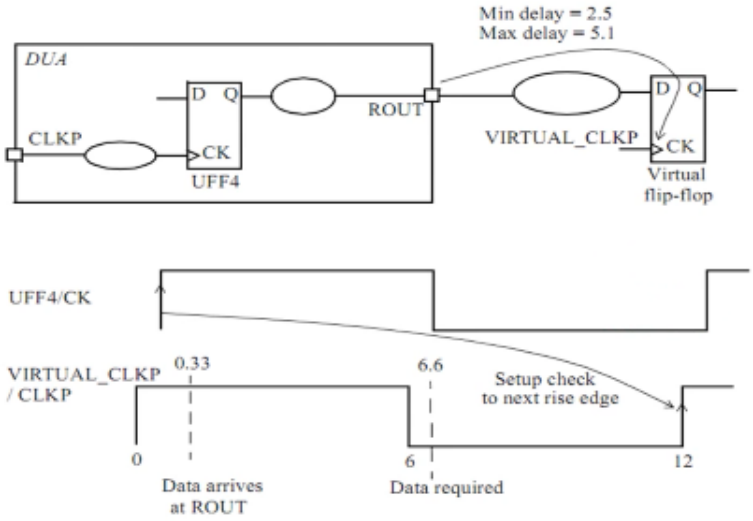
对Tdp取最小值，若能满足公式，则其他Tdp都能满足时序，不会发生时序违例

建立时间检查中Tdp取最大值，保持时间检查中Tdp取最小值

> PT工具中使用大值减去小值，在保持时间检查中使用Launch端的DataArrivalTime减去Capture端的Tskew与Thold

- Reg-to-OutputPort

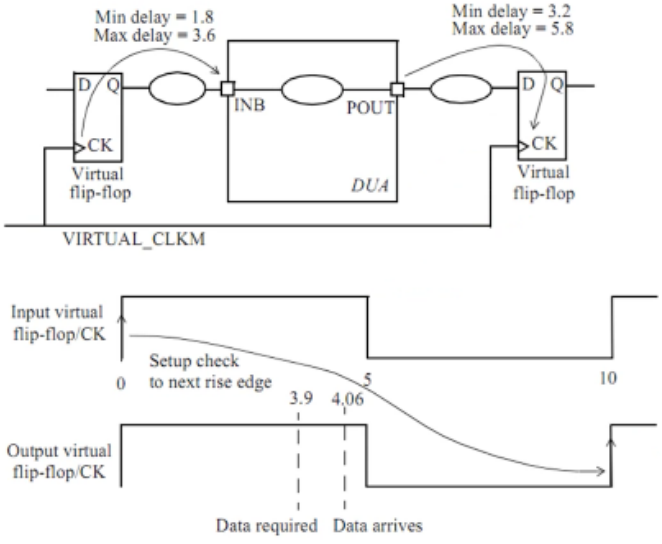
2、Flip-flop to Output Path



> 约束
>
> ```
> set_output_delay -clock VIRTUAL_CLKP -min 2.5 [get_ports ROUT]
> ```

- InputPort-to-OutputPort

3、Input to Output Path



> 约束
>
> ```
> # Input Constrain
> set_input_delay -clock VIRTUAL_CLKM -min 1.8 [get_ports INB]
> set_input_transition 0.8 [get_ports INB]
>
> # Output Constrain
> set_output_delay -clock VIRTUAL_CLKM -min 3.2 [get_ports POUT]
> set_load -pin_load 0.15 [get_ports POUT]
> ```