静态时序分析基础

电工电子工程基础II

内容

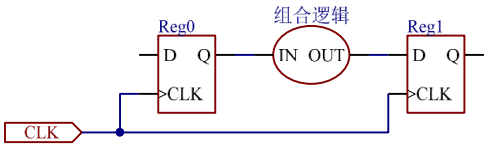
- 同步路径
- 外部同步路径
- 多周期
- 编译工具中的时序约束和分析

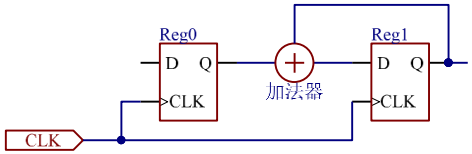
静态时序分析

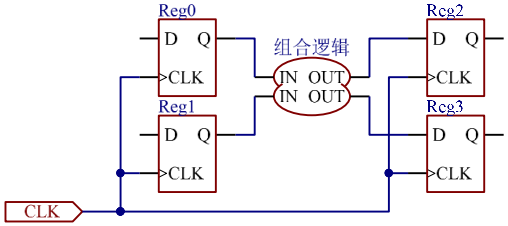
- 什么是时序分析
  - Quartus和Vivado中均有时序分析器
    - 帮助分析和验证一个编译完成的设计是否符合时序要求
      - 所有的数据路径都会按照对应的约束被分析
      - 整个设计必须符合时序要求
    - 指导编译过程编译出符合要求的结果

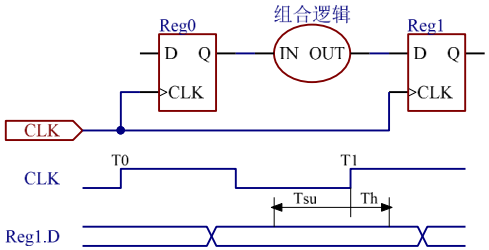
同步路经

- FPGA中逻辑的一般结构、建立和保持时间



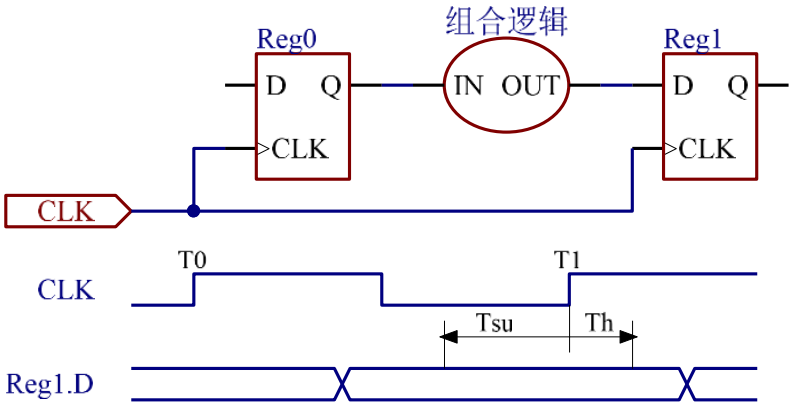






同步路径

- 一些术语：
  - 源寄存器
  - 目的寄存器
  - 启动沿、启动沿时刻
  - 锁存沿、锁存沿时刻
  - 建立时间、保持时间
  - 数据到达时刻
  - 建立时刻要求、保持时刻要求
  - 建立时间裕量、保持时间裕量

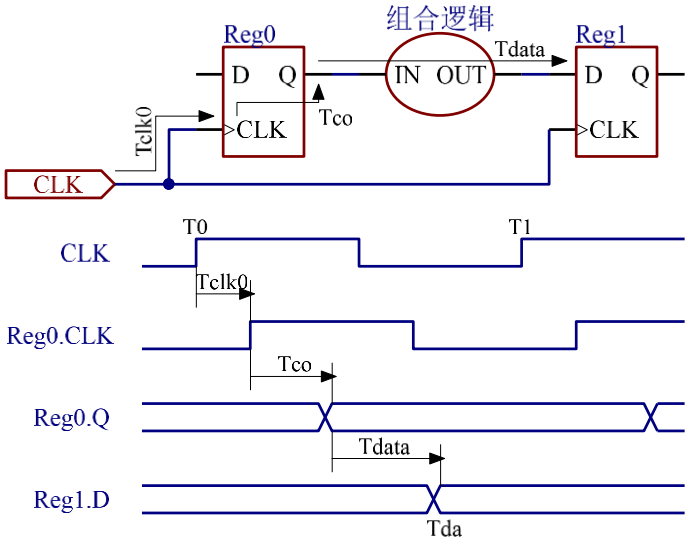


同步路径

- 数据到达时刻

$$T_{DA0} = T_0 + T_{clk0} + T_{co} + T_{data}$$

$$T_{DA1} = T_1 + T_{clk0} + T_{co} + T_{data}$$



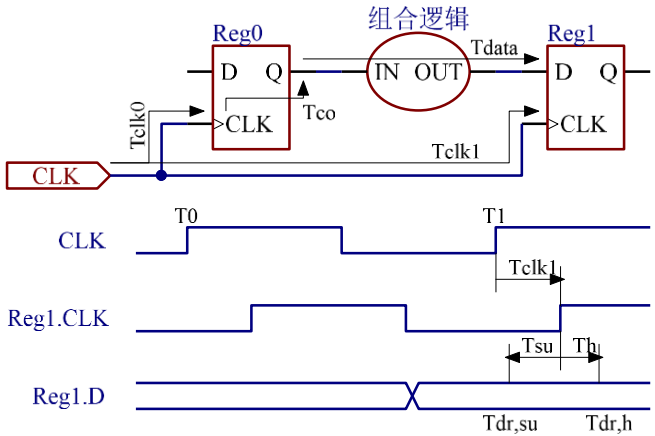
同步路径

- 建立时刻要求

$$T_{DR,SU} = T_1 + T_{clk1} - T_{SU}$$

- 保持时刻要求

$$T_{DR,H} = T_1 + T_{clk1} + T_H$$



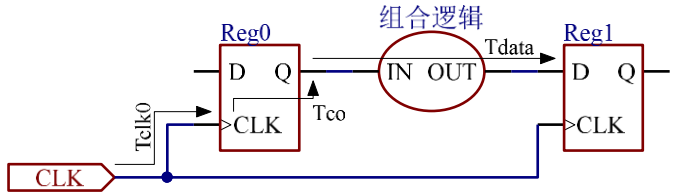
同步路径

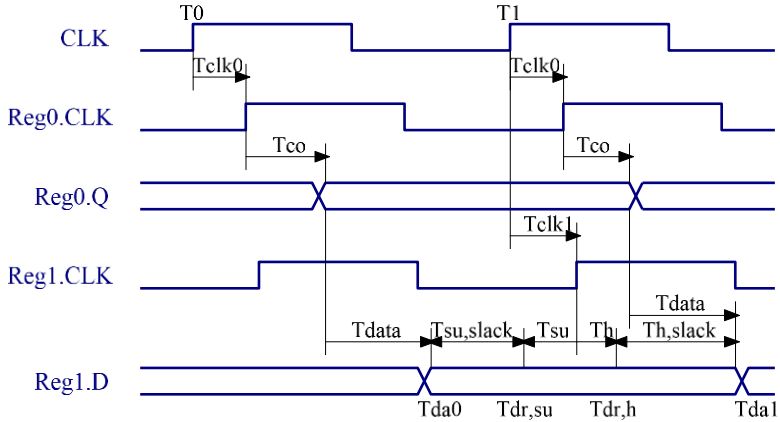
- 建立时间裕量

$$T_{SU,slack} = T_{DR,SU} - T_{DA0}$$
$$= \left(T_1 + T_{clk1} - T_{SU}\right) - \left(T_0 + T_{clk0} + T_{co} + T_{data}\right)$$
$$= T_P + \left(T_{clk1} - T_{clk0}\right) - T_{SU} - T_{co} - T_{data}$$

- 保持时间裕量

$$T_{H,slack} = T_{DA1} - T_{DR,H}$$
$$= \left(T_1 + T_{clk0} + T_{co} + T_{data}\right) - \left(T_1 + T_{clk1} + T_H\right)$$
$$= -\left(T_{clk1} - T_{clk0}\right) - T_H + T_{co} + T_{data}$$





同步路径

- 两个裕量
  - 建立时间要求数据不能出现得太晚，而保持时间则要求数据不能变化得太早。
  - 当一个包含时序约束的设计综合实现完成时，所有上述时间的最坏情况也就确定了，如有需要，时序分析器会自动地获取这些时间参数，帮我们分析所有路径的建立时间裕量和保持时间裕量。

异步路径

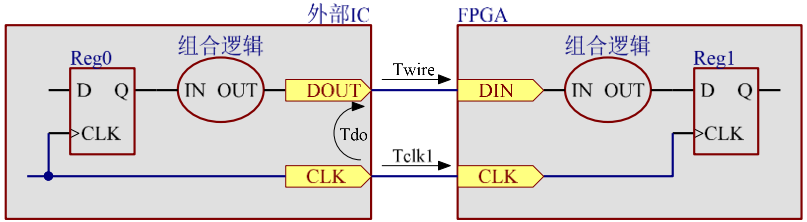
- FPGA内的异步路径
  - 异步复位
  - 异步复位过程虽不需要时钟参与，但并不是完全与时钟无关，在时钟上升沿附近，异步复位信号也是必须保持不变的，这点很像数据的建立和保持时间
  - 于是，类比地，在时钟有效沿之前异步信号必须保持稳定的最小时间称为**恢复时间**（Recovery Time）；在时钟有效沿之后异步信号必须保持稳定的最小时间称为**移除时间**（Removal Time）
  - 异步路径的具体分析过程，与同步路径分析几乎完全一样，恢复时间和移除时间的裕量也是时序分析器会自动地帮我们分析的

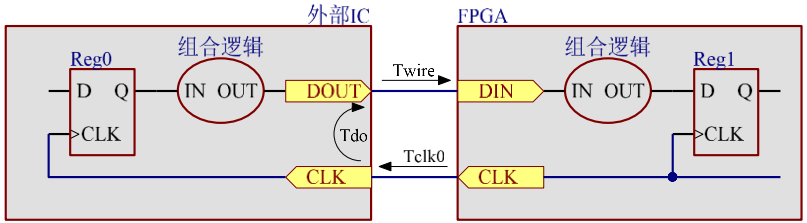
外部同步路径

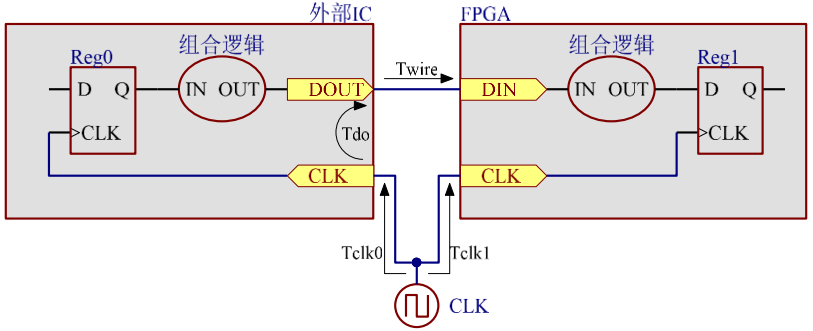
- 外部同步路径
  - FPGA与外部同步接口的外设的连接路径
    - 并行ADC、DAC
    - SSRAM、SDRAM
  - FPGA内部的时序关系，时序分析器知道，它会帮我们计算分析
  - FPGA和外部电路接口构成的路径，时序分析器不知道，需要我们通过时序约束命令告知时序分析器
    - 外部同步路径包括同步输入和同步输出

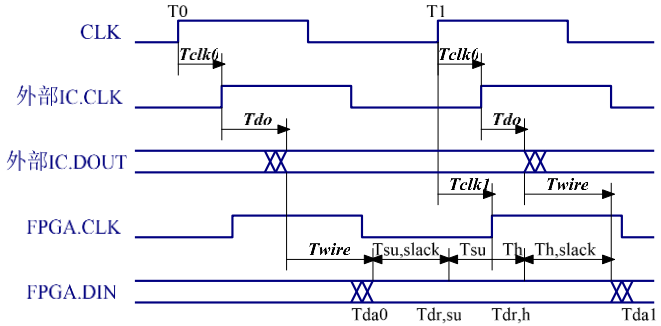
外部同步路径

- 外部同步输入



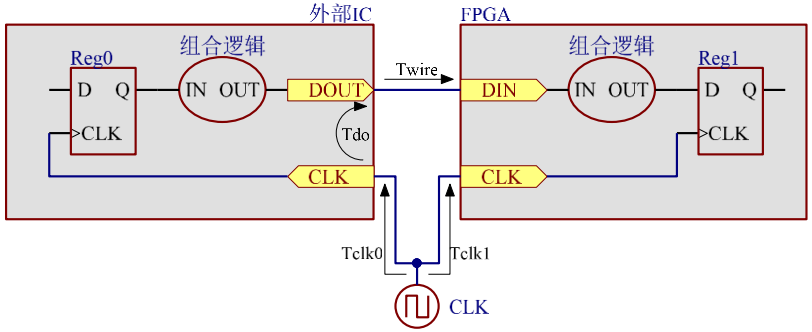


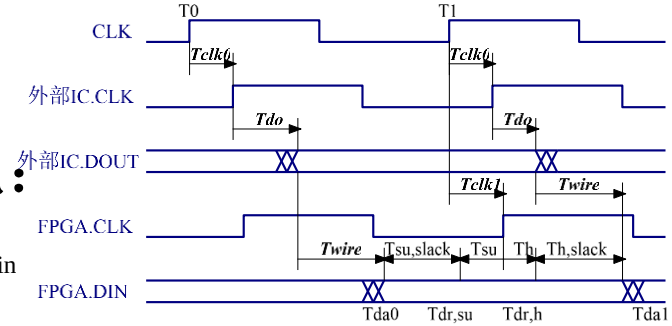




外部同步路径

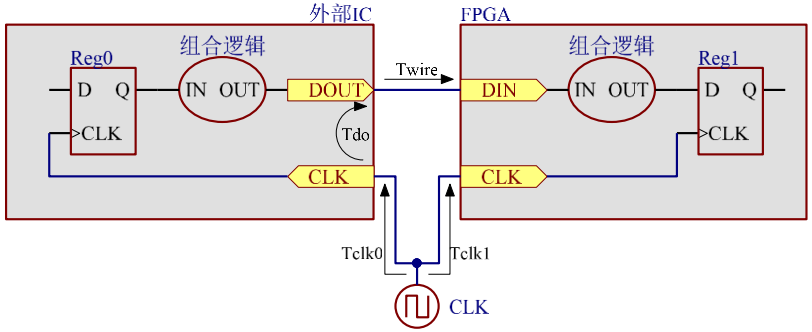
- 外部同步输入的裕量
  - 建立时间：
    - $T_{SU,slack} = (T_1 + T_{clk1} - T_{SU}) - (T_0 + T_{clk0} + T_{DO} + T_{wire})$
      $= T_P - T_{SU} - (T_{clk0} + T_{DO} + T_{wire} - T_{clk1})$
  - 括号内为外部时间
    - 最坏情况是它取最大值，记：
    - $T_{inputdelay,\max} = T_{clk0,\max} + T_{DO,\max} + T_{wire,\max} - T_{clk1,\min}$

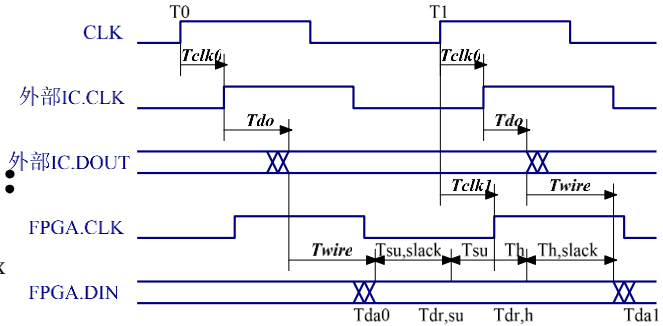




外部同步路径

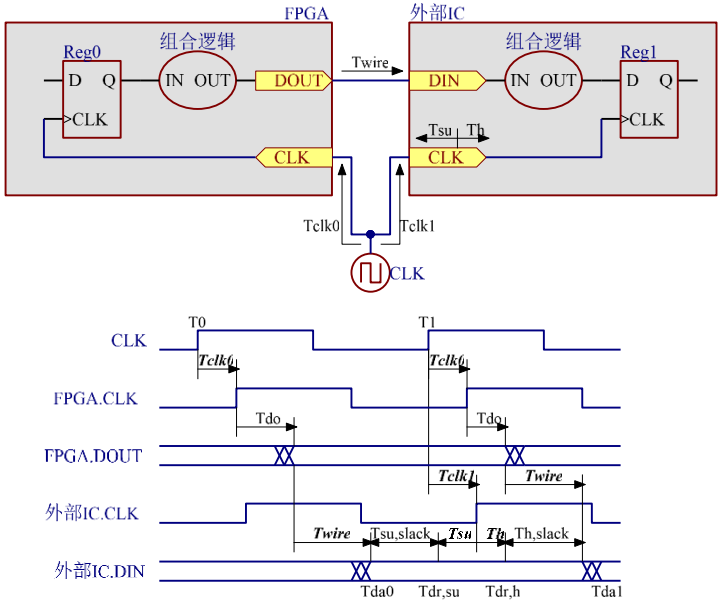
- 外部同步输入的裕量
  - 保持时间：
    - $T_{H,slack} = (T_1 + T_{clk0} + T_{DO} + T_{wire}) - (T_1 + T_{clk1} + T_H)$
      $= -T_{SU} + (T_{clk0} + T_{DO} + T_{wire} - T_{clk1})$
  - 括号内为外部时间
    - 最坏情况是它取最小值，记：
    - $T_{inputdelay,\min} = T_{clk0,\min} + T_{DO,\min} + T_{wire,\min} - T_{clk1,\max}$





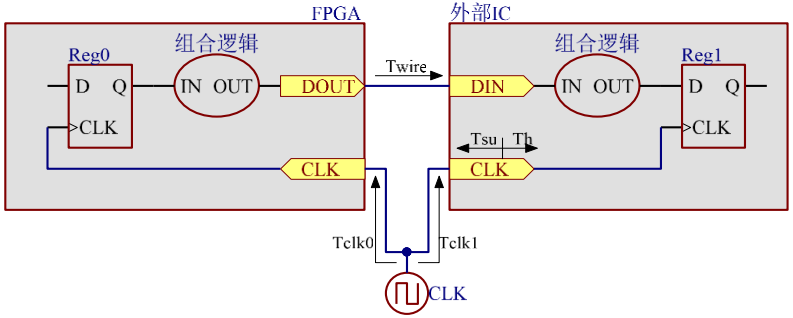
外部同步路径

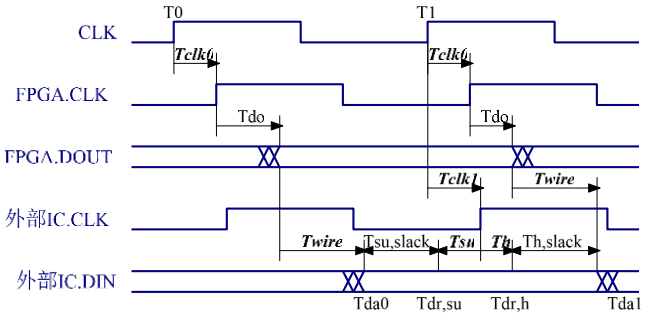
- 外部同步输出



外部同步路径

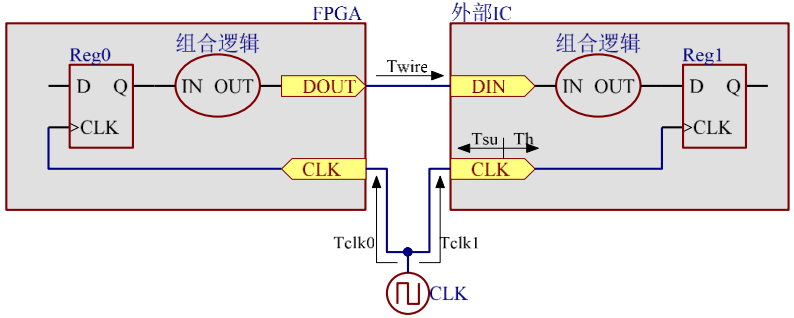
- 外部同步输出的裕量
  - 建立时间
    - $T_{SU,slack} = (T_1 + T_{clk1} - T_{SU}) - (T_0 + T_{clk0} + T_{DO} + T_{wire})$
      $= T_P - T_{DO} - (T_{clk0} + T_{wire} + T_{SU} - T_{clk1})$
  - 括号内为外部时间
    - 最坏情况是它取最大值，记：
    - $T_{outputdelay,\max} = T_{clk0,\max} + T_{wire,\max} + T_{SU,\max} - T_{clk1,\min}$

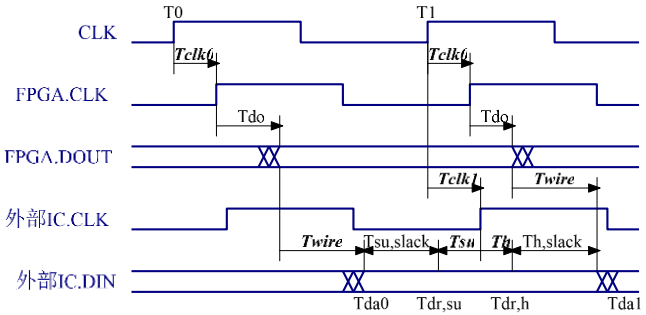




外部同步路径

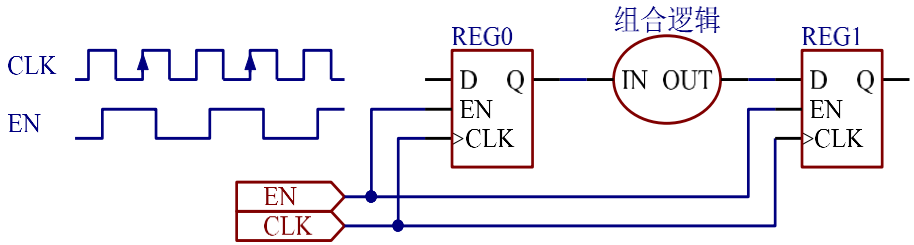
- 外部同步输出的裕量
  - 保持时间
    - $T_{H,slack} = \left(T_1 + T_{clk0} + T_{DO} + T_{wire}\right) - \left(T_1 + T_{clk1} + T_H\right)$
      $= T_{DO} + \left(T_{clk0} + T_{wire} - T_H - T_{clk1}\right)$
  - 括号内为外部时间
    - 最坏情况是它取最小值，记：
    - $T_{outputdelay,\min} = T_{clk0,\min} + T_{wire,\min} - T_{H,\max} - T_{clk1,\max}$

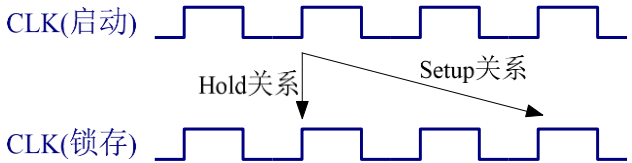




多周期路径

- 多周期路径
  - 带有使能控制的路径一般是多周期路径



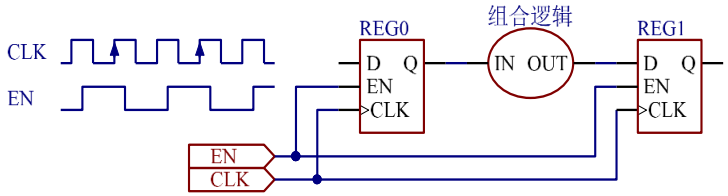


时序约束和分析

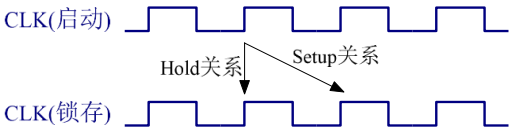
- 内部路径的约束
  - 时钟
    - create_clock -name <clock_name>
      -period <period_ns> [get_ports <port_name>]
    - eg.:
      create_clock -name input_clk -period 20.0 [get_ports clkin]

时序约束和分析

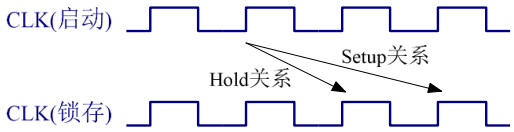
- 内部路径的约束
  - 多周期路径



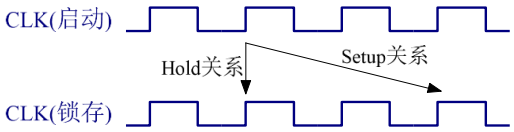
- 建立时间关系：
  set_multicycle_path -setup -from [get_pins <path_to_reg0>]
  -to [get_pins <path_to_reg1>] <setup_relation>
- 保持时间关系：
  set_multicycle_path -hold -from [get_pins <path_to_reg0>]
  -to [get_pins <path_to_reg1>] <hold_relation>



s=1, h=0



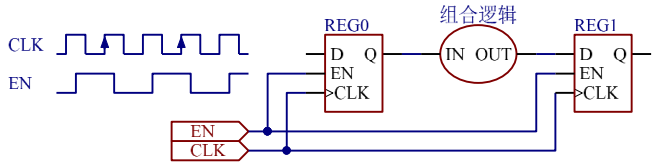
s=2, h=0



s=2, h=1

时序约束和分析

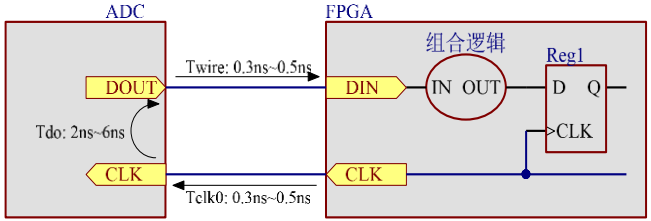
- 内部路径的约束
  - 多周期路径



    - eg.(vivado .xdc):
      - set_multicycle_path -from [get_pins {inst1/q_reg[*]/C}] -to [get_pins {inst1/data_reg[*]/I}] -setup -start 2
      - set_multicycle_path -from [get_pins {inst1/q_reg[*]/C}] -to [get_pins {inst1/data_reg[*]/I}] -hold -start 1
    - eg.(quartus .sdc):
      - set_multicycle_path -from [get_pins {inst1|q[*]}] -to [get_pins {inst1|data[*]}] -setup -start 2
      - set_multicycle_path -from [get_pins {inst1|q[*]}] -to [get_pins {inst1|data[*]}] -hold -start 1

时序约束和分析

- 外部同步路径的约束
  - 外部同步输入



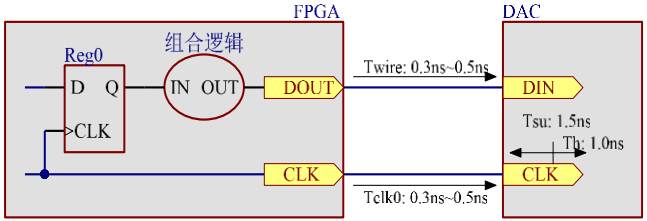

```
create_generated_clock –name adc_clk –source \
[get_ports clk_in] [get_ports ADCLK]

set_input_delay –clock adc_clk –max [expr .5+6+.5-0] \
[get_ports ADC_D*]

set_input_delay –clock adc_clk –min [expr .3+2+.3-0] \
[get_ports ADC_D*]
```

$$T_{inputdelay,\max} = T_{clk0,\max} + T_{DO,\max} + T_{wire,\max} - T_{clk1,\min}$$

$$T_{inputdelay,\min} = T_{clk0,\min} + T_{DO,\min} + T_{wire,\min} - T_{clk1,\max}$$

时序约束和分析

- 外部同步路径的约束
  - 外部同步输出




```
create_generated_clock –name dac_clk –source \
[get_ports clk_in] [get_ports DACLK]

set_output_delay –clock dac_clk –max [expr 0+.5+1.5-.3] \
[get_ports DAC_D*]

set_output_delay –clock dac_clk –min [expr 0+.3-1.0-.5] \
[get_ports DAC_D*]
```

$$T_{outputdelay,\max} = T_{clk0,\max} + T_{wire,\max} + T_{SU,\max} - T_{clk1,\min}$$

$$T_{outputdelay,\min} = T_{clk0,\min} + T_{wire,\min} - T_{H,\max} - T_{clk1,\max}$$

时序约束和分析

- 时序工具和分析报告
  - Quartus
    - TimeQuest Timing Analyzer
  - Vivado
    - Timing Report

时序约束和分析

- 时序工具和分析报告
  - Vivado Timing Report