FPGA内数据传输模型

本文中讨论的 DFF 、D触发器、寄存器是等价的；

详解D 触发器

下图为一个D触发器的典型符号图：

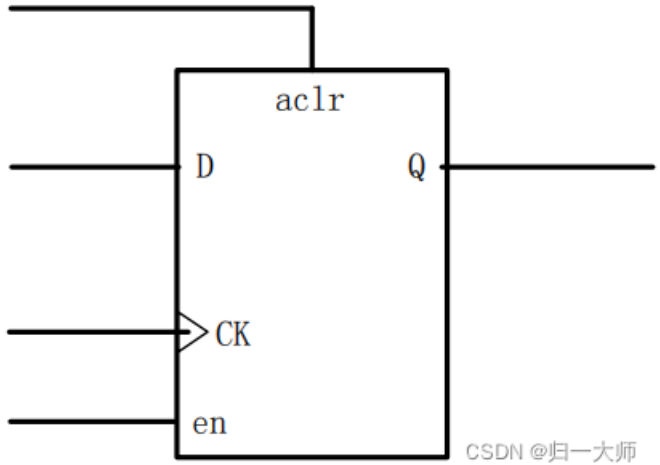

图1 D触发器

D 触发器概念：

1. D 触发器是一个具有记忆功能的，具有两个稳定状态的信息存储器件，是构成多种时序电路的最基本逻辑单元，也是数字逻辑电路中一种重要的单元电路。
2. D 触发器在数字系统和计算机中有着广泛的应用。触发器具有两个稳定状态，即"0"和"1"，在一定的外界信号作用下，可以从一个稳定状态翻转到另一个稳定状态。
3. D 触发器在 CLK(时钟脉冲)的前沿（正跳变 0→1）发生翻转，触发器的次态取决于 CLK的脉冲上升沿到来之前D端的状态，即次态Q=D。因此，它具有置 0、置 1 两种功能。由于在 CLK=1 期间电路具有维持阻塞作用，所以在 CK=1 期间，D 端的数据状态变化，不会影响触发器的输出状态

在上面的介绍中，我们需要提取出以下几个关键信息：

1、 D 触发器具有2个稳定状态“0”和“1”，在时钟（CLK）的上升沿，D端的数据被更新到Q端。
2、 在时钟的高电平期间，D端的数据变化。Q端的数据不会跟着变化。

针对第一条信息，可以知道D触发器在时钟（CLK）的上升沿时将D端的数据存起来，所以，要想保证D端的数据能被正确的存起来，需要确保D端的数据在时钟上升沿时刻是稳定的，不能处于变化时期。假如在时钟上升沿的那个点，刚好D端的数据也在变化，那就真的不知道存储起来的究竟是0还是1。而且，理想的D触发器，认为时钟信号从0到1的变化所需时

间无限短，那么这个上升沿时间究竟是多长呢？理想的时钟上升沿可以被认为是从0到1这个变化的过程所需的时间是无限短的，但是实际在电路中，这个变化还是需要一定的时间的，不可能无限短。而且D触发器也不是真的就只需要在时钟上升沿的那一个时间点需要D端的数据是稳定即可，从实际的器件工作情况来看，任何一个D触发器要想在时钟的上升沿将D端的数据寄存并输出到Q端，有2 个关键的时间参数需要保证。

建立时间，保持时间，输出延迟

为了更加直观的说明这个问题，还是先画一张图说明。

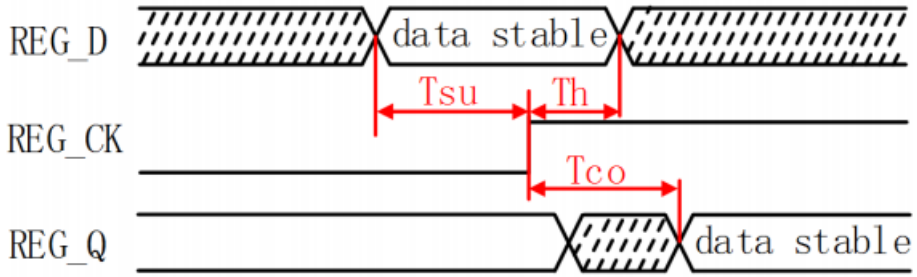

图2 数据传输示意图

对于实际的 D 触发器来说，为了保证在时钟的上升沿能够正确的将D端的数据寄存并输出到Q端，需要满足以下两点：

1. D端的数据必须在时钟上升沿到来之前的一定时间内就已经保持稳定，该时间被称为D触发器的建立时间（Tsu）。
2. D端的数据必须在时钟上升沿到来之后的一定时间内继续保持稳定，该时间被称为D触发器的保持时间（Th）。

如果不能同时满足上述两个条件，那么D端的数据就可能无法正确的被寄存并输出至Q端。同时，对于D触发器来说，D端的数据也不可能会在时钟上升沿出现的那一刻就立即更新到Q端，从时钟的上升沿到D端的数据稳定出现在Q端，也有一个时间，该时间称为 寄存器 的时钟到输出延迟（Tco）。这就是D触发器的实际电路结构决定的，可以认为是通过测量手段测量出来的经验值。而且，从更深次的来说，如果D触发器没有这个特性，而是真正理想的特性，那么D触发器将没有存在的意义。

实例讲解数据传输路径（不考虑时钟延迟）

此案例假设CLK到达所有D触发器的时间一致，没有延迟；

分析以下代码：

```
module reg_test(
    input         clk   ,
    input         rst_n ,

    input         a     ,

```

```
    input         b     ,

    output reg    c
    );

    reg       a_reg     ;
    reg       b_reg     ;

    always  @(posedge clk or negedge rst_n)begin
        if(rst_n==1'b0)begin
            a_reg <= 1'b0;
            b_reg <= 1'b0;
        end
        else begin
            a_reg <= a;
            b_reg <= b;
        end
    end

    always  @(posedge clk or negedge rst_n)begin
        if(rst_n==1'b0)begin
            c <= 1'b0;
        end
        else begin
            c <= a & b;
        end
    end

    endmodule
```

内部综合之后的电路如下图所示：

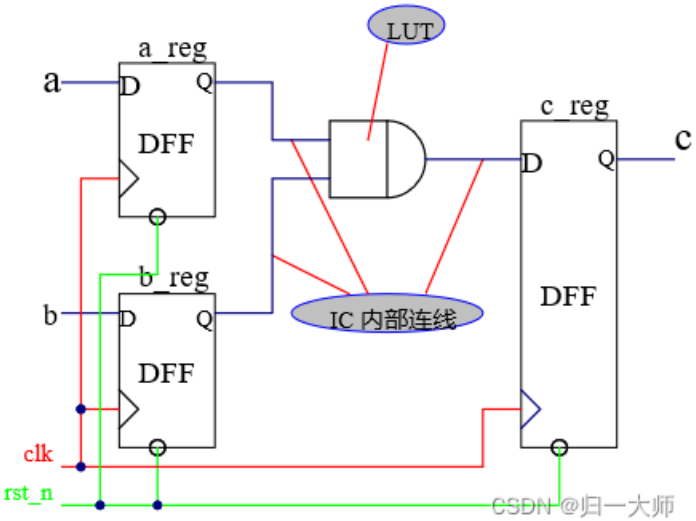

图3 代码综合后的电路示意图

假如初始时a=b=1，某一时刻a=0，则当D触发器延迟，内部连线延迟，组合逻辑单元延迟（与门延迟），内部连线延迟，D触发器延迟之后C=0；两个D触发器之间的组合逻辑单元的延迟加上内部连线延迟的结果通常称为**数据传输延迟（Tdata）**；

在复杂的组合逻辑运算时，两个触发器之间可能会通过多次组合逻辑单元延迟，内部连线延迟，尤其是有多个if_else结构时，多个组合逻辑串行连接时，组合逻辑延迟就会很大，所以if_else不要过多级联；

a=b=1，某一时刻a=0，之后的时序图如下：

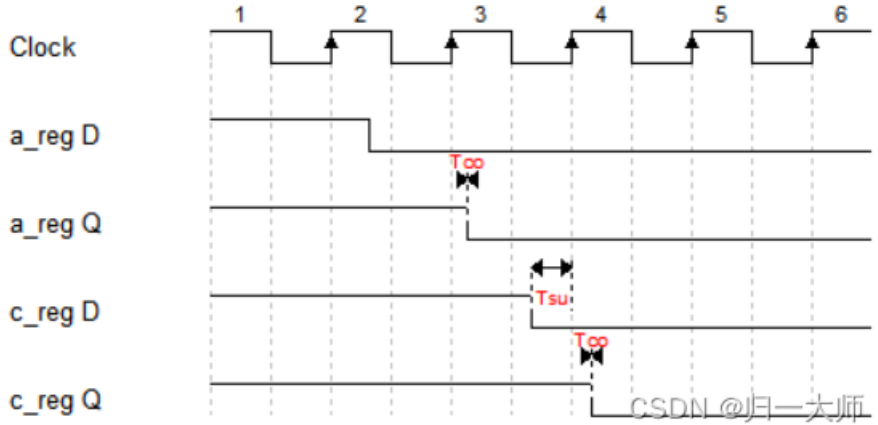

图4 数据传输示意图

D触发器（DFF）延迟时间Tco：时钟上升沿到达D触发器 到 有效数据输出到Q端的延迟。

建立时间Tsu：目的寄存器自身的特性决定，在时钟信号上升沿到达其时钟接口时，其数据输入端（D）的数据必须提前Nns稳定下来，否则就无法确保数据正确存储。

建立时间Tsu：D触发器D端口的数据必须比时钟上升沿提前Nns到达D触发器的端口。

时钟和数据传输路径（考虑时钟延迟）

从此处开始引入时钟信号到达各个D触发器（DFF）的延迟：

在实际的芯片之中，时钟到达各个触发器如下图所示，外部时钟信号clk进入FPGA芯片之后通过时钟buff，经全局时钟树后传输到各个D触发器的时钟端作为时钟信号，由此两个不同的D触发器的时钟必然存在时间差；

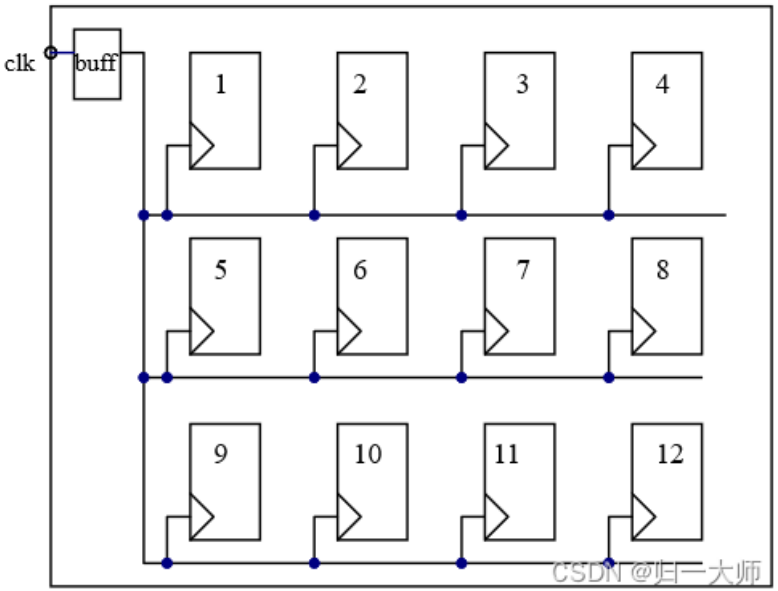

图5 FPGA各个寄存器与时钟管脚分布示意图

通过上一节内容中，可以提取归纳出一个通用的 FPGA 内部信号传输模型，如下图所示：

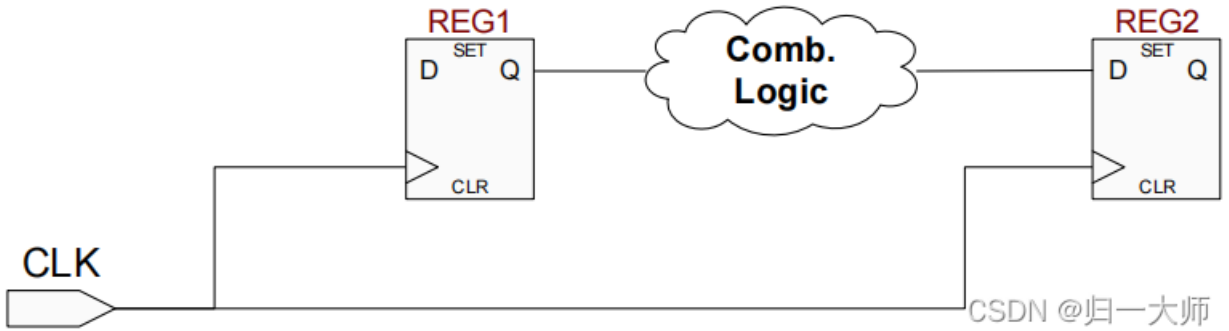

图6 FPGA内部信号传输模型

根据上图，可以概括出以下几个**关键点**：

1、 数据总是从一个D触发器（REG1）传到另一个D触发器（REG2），我们称数据发出D触发器为源寄存器，数据接收D触发器为目的寄存器。

2、 数据在传输过程中，可能参与多次组合逻辑变换。

3、 数据在传输过程中，需要经过可编程互联线传递。

4、 分析信号在两个寄存器之间的传递时，需要保证这两个寄存器的时钟信号是同源时钟。

针对第四点，需要补充的是，为啥要说是同源时钟，不说同一个时钟呢。其实，在实际使用中，有可能两个寄存器的工作时钟信号并非同一个。典型的例如一个 PLL 输出的两个时钟分别作为目的寄存器和源寄存器的时钟，这种情况下看似时钟不同，但是实际上在分析时，起点都是从 PLL 的时钟输入端开始，而从时钟输入到 PLL 的两个输出时间会算入时钟路径，因此本质上还是同一个时钟，所以上面说的是要求两个时钟是同源即可。

FPGA内数据传输典型时序

下图就是Altera的手册中FPGA典型的数据从寄存器到寄存器的传输时序图：

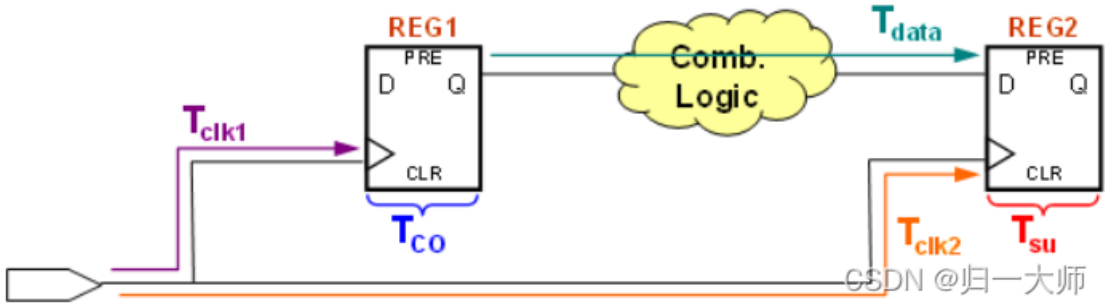

图7 FPGA内部信号传输时序图

将源触发器发送数据的时钟沿称为发生沿（launch），目的触发器接收数据的时钟沿称为接受沿（latch）。上图中可以看到，两个寄存器的时钟脚都是连在一起的，且来自同一个外部 Pin，只是，从外部 pin 到寄存器 1（REG1）和寄存器 2（REG2）的时钟端口，中间都是有一段路要走的。所以REG1.CLK、REG2.CLK与外部管脚pin的时钟信号CLK之间存在一些延迟，分别为Tclk1和Tclk2。将接收沿延迟时间（Tclk2）减去发送沿延迟时间（Tclk1）的时间差称为**时钟偏斜（Tskew）**；

D触发器存在延迟时间，REG1触发器必须在REG1.CLK上升沿到来之后经过Tco延迟时间才能在Q端接收到D端的数据。之后Q端输出数据经过Tdata延迟（组合逻辑延时以及互联线延迟）后到达REG2触发器的D端；

REG2触发器的时钟会延迟Tclk2，根据建立时间的要求，D端的数据必须比REG2.CLK提前Tsu时间稳定，才能保证REG2触发器能接收到有效的数据，时序图如下图所示：

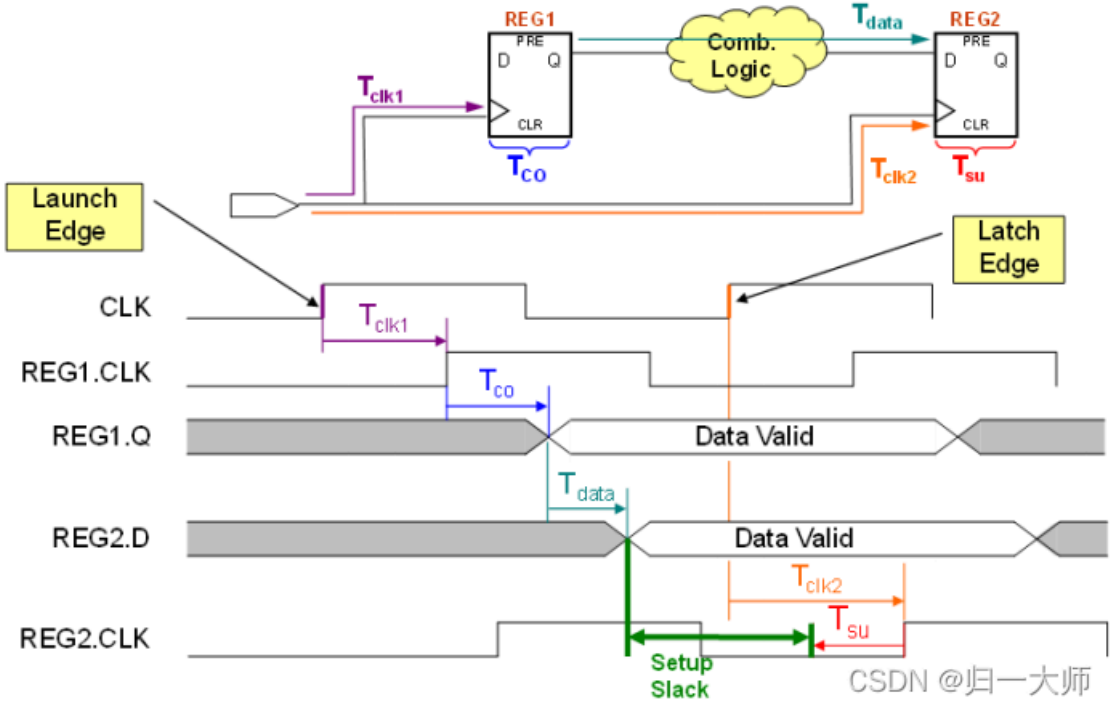

图8 结合传输模型分析建立时间

根据上述分析可得：

💡 目的寄存器能够正确接收源寄存器发射过来的数据需要满足的条件：

$$Tclk1 + Tco + Tdata \le Tclk + Tclk2 - Tsu$$

由于Tskew = Tclk2 - Tclk1,移位变换得：

$$Tclk + Tskew - Tsu - Tco - Tdata \ge 0$$

建立时间余量：Slack = Tclk + Tskew - Tsu - Tco - Tdata

当Slack ≥ 0时，目的寄存器能够正确的接收源寄存器发射过来的数据；

Tclk1：时钟信号从时钟源端口出发，到达源寄存器时钟端口的时间；

Tco：时钟上升沿到达寄存器时钟端 到 数据输出到寄存器Q端口的时间；

Tdata：数据从源寄存器Q端出发，到达目的寄存器D端的时间；

Tclk：时钟周期；

Tclk2：时钟信号从时钟源端口出发，到达目的寄存器时钟端口的时间；

Tsu：寄存器要求的 其数据端口的值必须提前于时钟上升沿到达其时钟端口的时间值；

Tskew：时钟从源端口出发，到达目的寄存器和源寄存器时钟端口的时间差值（Tclk2 - Tclk1）；

数据到达时间（TimeQuest里面的计算方式）：Tclk1+Tco+Tdata；

数据需求时间：Tclk+Tclk2-Tsu；

Slack：数据需求时间和数据到达时间的差值，如果为正值，则表明数据能被目的寄存器正确接收，如果为负值，则表明数据不能被目的寄存器正确接收；

当时钟频率增大时，时钟周期Tclk将减小，相同的逻辑电路，相同的器件，对应的建立时间余量Slack将减小，当Slack减小到0时，系统时钟的周期达到最小值，频率达到最大值，此时的频率就是对应的最大时钟频率；

当系统有多条路径时，系统所能运行最大时钟频率由最长的路径决定；

您的支持是我更新的最大动力！将持续更新工程，如果本文对您有帮助，还请多多点赞👍、评论💬和收藏⭐！