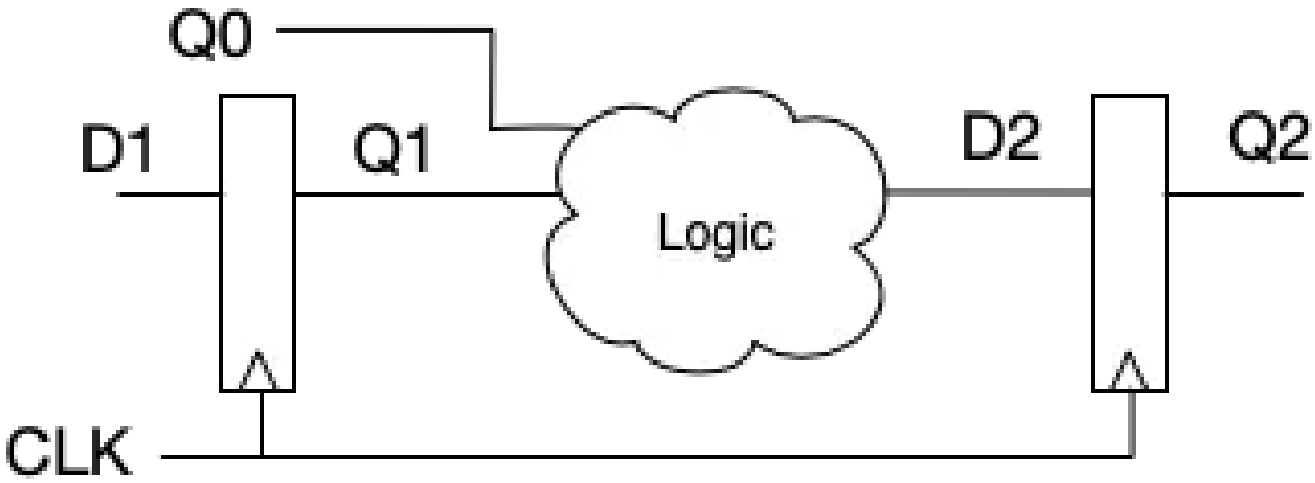
跟老李一起学习芯片设计-- CDC的那些事（1）

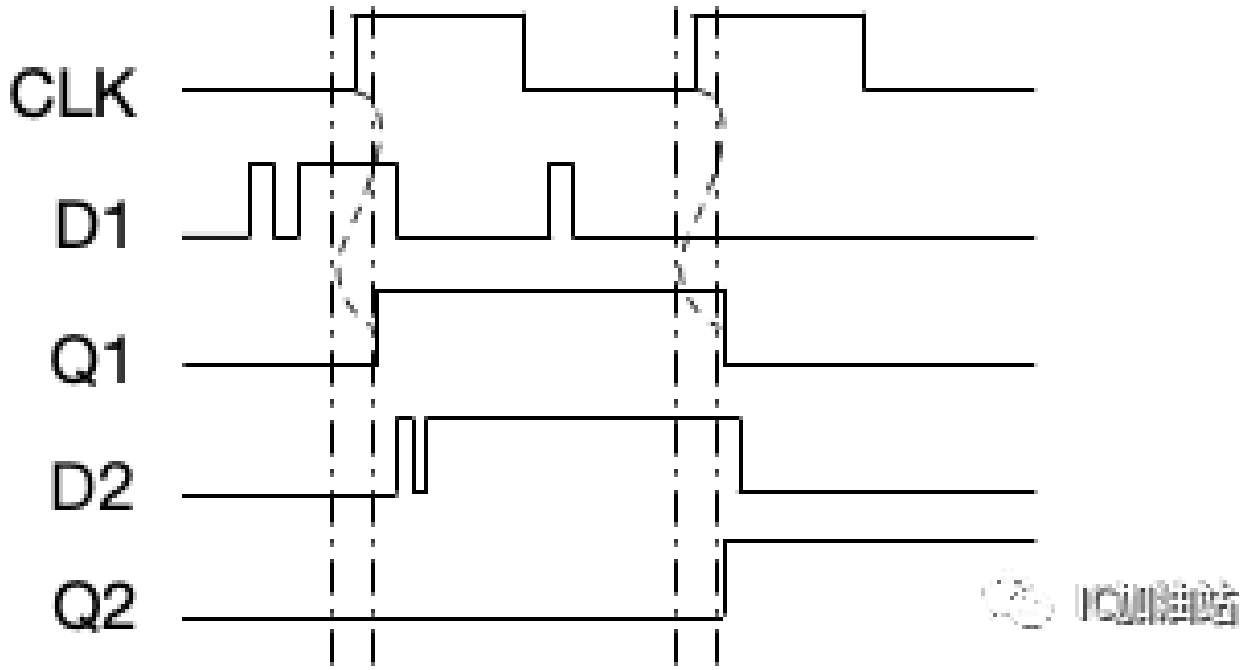
原创 李虹江 IC加油站 5月26日

一转眼，老李在数字芯片设计领域又摸爬滚打了四年。上篇推送已经四年前了，久到我差点忘记了还开过这个公众号。前两天偶然想起，便觉得还是要在这个平台输出一些内容。这些年在工作中也算是积累了一些经验，其中有些经验值得记录下来，和大家做一个交流。

从这篇开始，老李推送一系列芯片设计中跨时钟域（CDC) 的知识。跨时钟域设计几乎是现代数字芯片设计中所有设计人员必须要掌握的基本知识，但是从老李自身的经历来说，这方面的知识在本科和研究生的课程当中都没有讲过，却是面试中经常被考到的知识点。同时工作中工程师也必不可少地要和CDC打交道，考虑如何进行跨时钟域设计，以及掌握CDC 相关的EDA工具。以下文章我们就从跨时钟域设计的最基本开始讲起，在之后的推送中，老李还会依次分享跨时钟域的进阶内容。

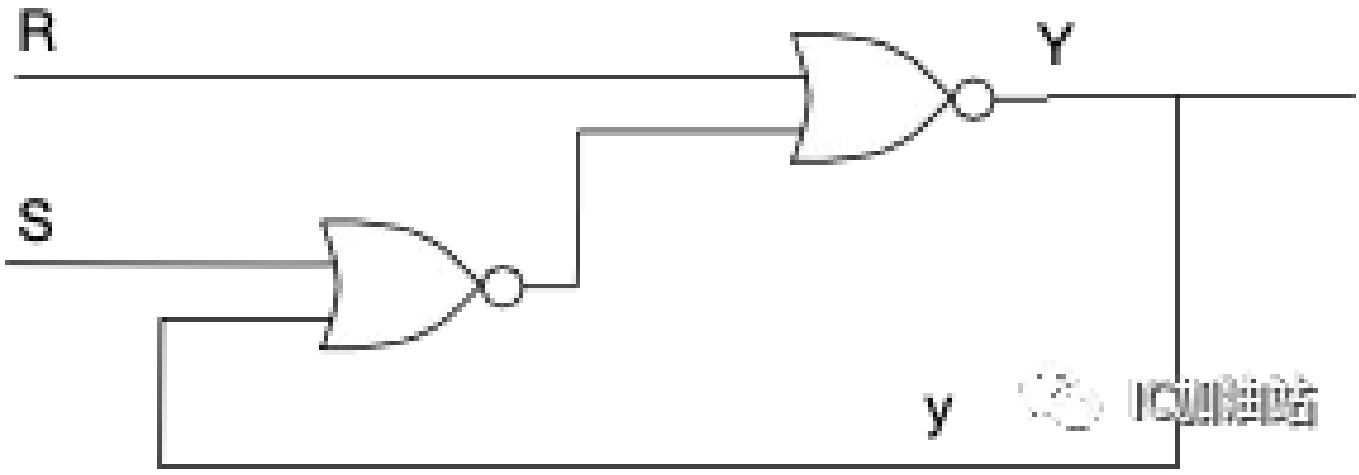
这里我们先复习一下同步电路和异步电路的概念。在现代SoC设计中，绝大多数的电路都是同步电路。同步电路是由时钟驱动存储元件的电路，也就是说存储元件的状态只在时钟沿到来的时候才能发生变化。因为组合逻辑电路在输入变化时输出可能出现毛刺(glitch)，而存储元件因为只会在时钟沿到来时才会更新状态，那么在时钟沿之间的时间状态是稳定的，这样的同步电路可以消除组合电路中的毛刺，如下图所示。





相应的，时钟周期的大小取决于最长的传输延时(propagation delay)。同步电路的好处是时序很清晰，电路中的存储元件例如触发器都是依照一个固定的节拍来工作，便于EDA工具来进行延时的分析和计算，所以同步电路几乎占据了当前数字芯片的绝大多数部分。

而异步电路就没有时钟的概念了，存储元件所存的状态跟随了输入信号的变化立刻发生变化。信号之间的传递通常通过握手(handshake)来完成，因为没有时钟的约束，每一级存储元件之间的逻辑电路都是各自独立的，可以各自进行优化，这样可以达到很好的性能。但是这既是优势，也是劣势。劣势就在于EDA工具没有满足每一级都单独优化的计算能力，而且由于相邻的级之间互相影响，使得计算总的时序时变得异常复杂，所以异步电路的规模通常无法做大，进而也限制了它的用途。目前更多的也只是在学术研究方面，并没有成为当今SoC设计的主流。下面这个例子实现了一个状态存储单元，next state Y会依据当前状态以及R,S的值立刻发生变化，显然这样的状态变化更加快速，但是分析起来也更加复杂。

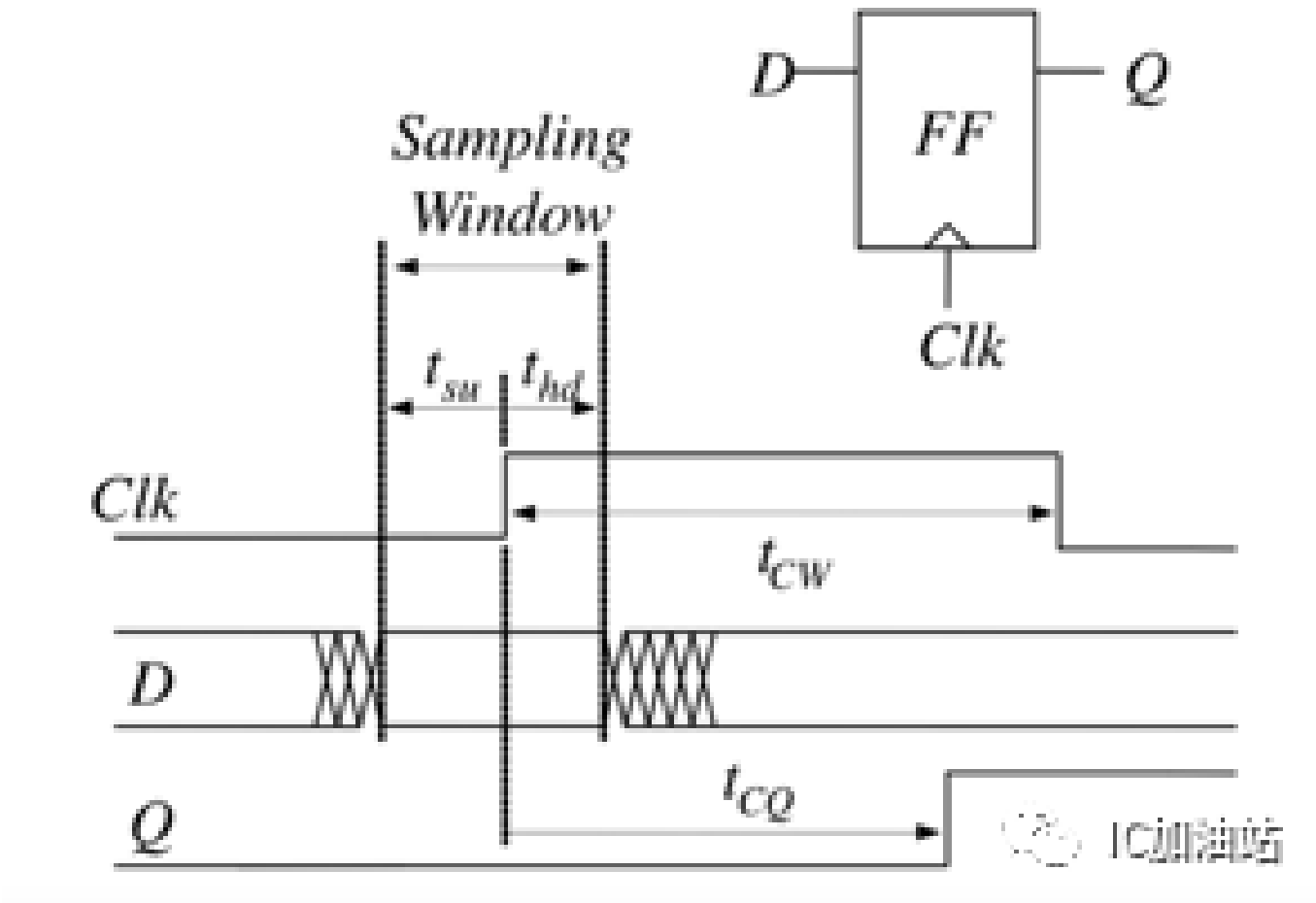


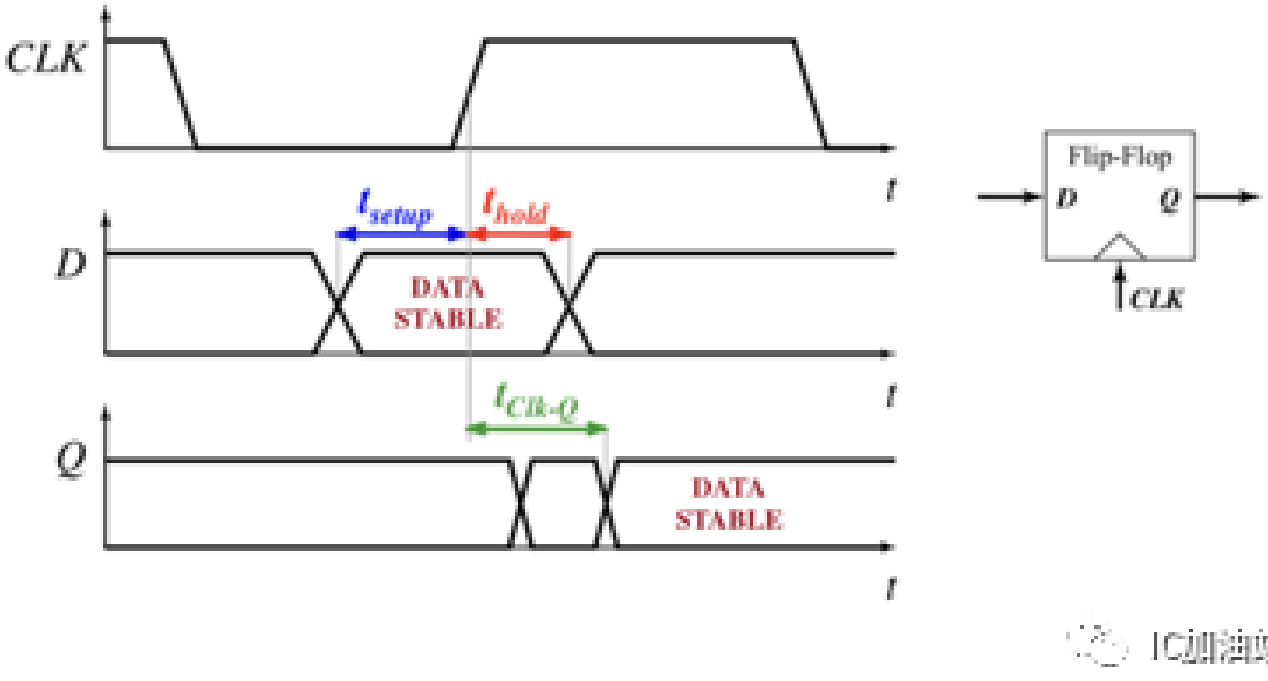
Present State	Next state			
y	SR=00	01	10	11
	Y	Y	Y	Y
0	0	0	1	0
1	1	0	1	0

我们接下来的跨时钟域分析，当然都是基于同步电路的。同步电路的核心就是触发器，触发器的种类有很多种，最常用的就是D触发器。在这里我们还是首先复习一下基本概念: 建立时间setup time和保持时间hold time, 以及亚稳态metastability。

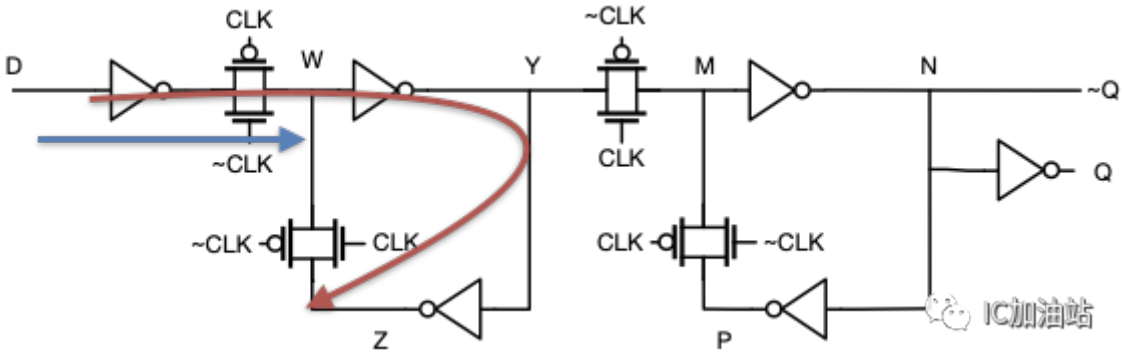
setup time: 时钟沿到来之前输入信号D必须保持稳定的最小时间
hold time: 时钟沿到来之后输入信号D必须保持稳定的最小时间
clk-to-q time: 输入D满足setup/hold time要求，从时钟沿到来时刻到输出端Q变化至稳定的时间

那么当输入信号D无法满足setup time 或者hold time的要求，我们称之为产生了setup time / hold time violation, Flop Q的输出这个时候是0还是1是不确定的，需要一定的时间才能够稳定在0或者1。所以如果当Q端在clk-to-q时间之后才变得稳定的话，我们就说这个触发器产生了亚稳态metastability。





很多工程师在面试的时候都可以回答得上setup time和hold time的定义，但是回答不上为什么D触发器有setup time和hold time的要求。这个问题其实大家在学校里学过，如果你在面试的时候被问到却不知道，那么你应该回去好好复习一下基础知识。它们与D触发器的内部结构有关系。D触发器的内部是一个主从锁存器(master-slave latch)，一个常见的D触发器结构如下图所示



Latch能够存储住状态，靠的是上面的背靠背的反相器。而这个背靠背的反相器能够锁住状态是需要时间的。由此，我们可以分析出

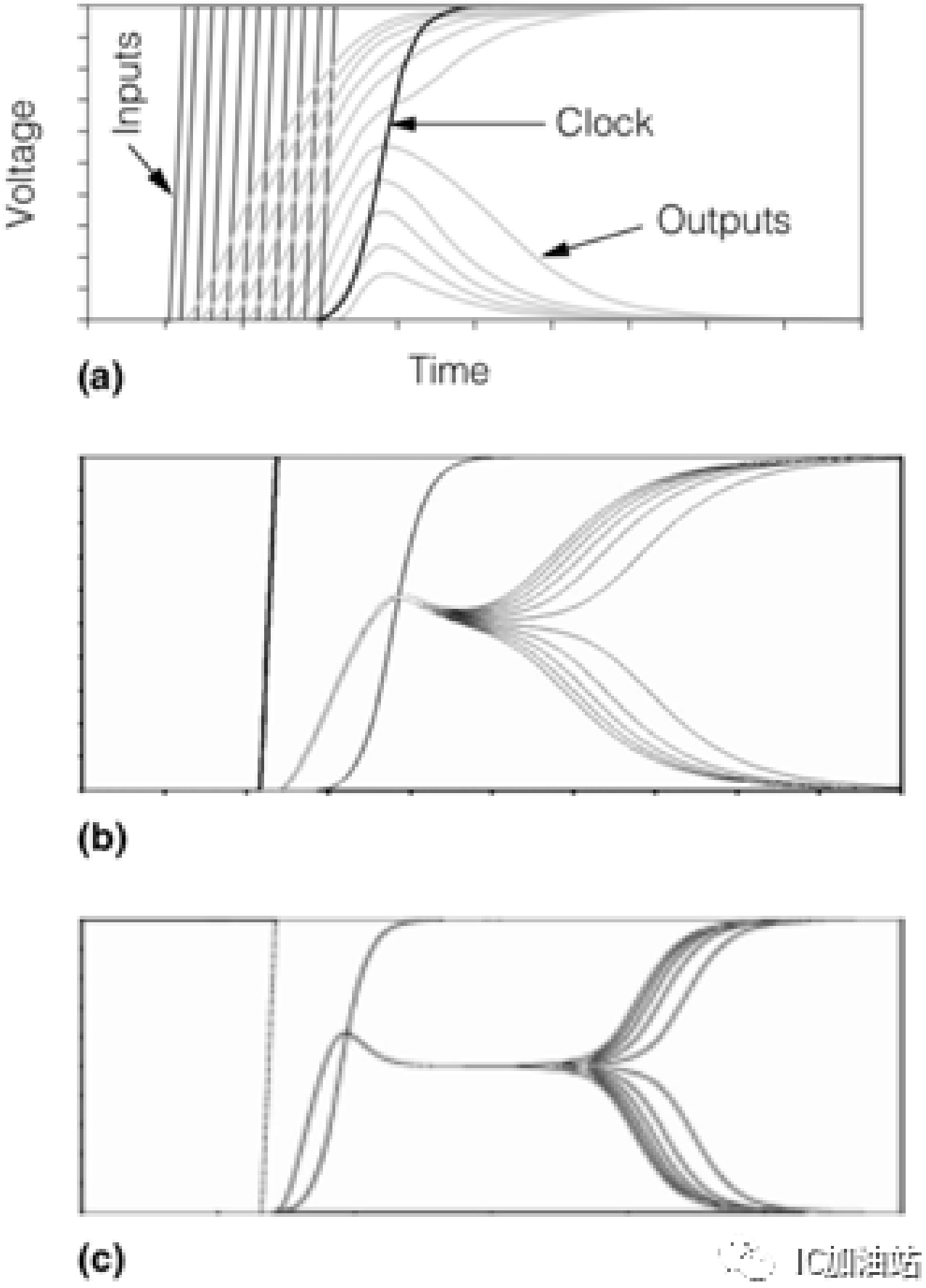
setup time: 在clk的上升沿到来之前，D要传输到Z的时间。因为当Z的值还没有稳定的时候，D如果变化，那么这个背靠背的反相器就无法锁住值。

hold time: 第一个传输门关闭需要的时间，在传输门关闭期间，D->W要保持稳定，这样在传输门关闭之后，W稳定才不会导致背靠背反相器锁住的值发生变化。

所以我们可以看出，当D在setup/hold time window内发生变化，锁存器可能无法锁住一个稳定的值，会发生的结果是

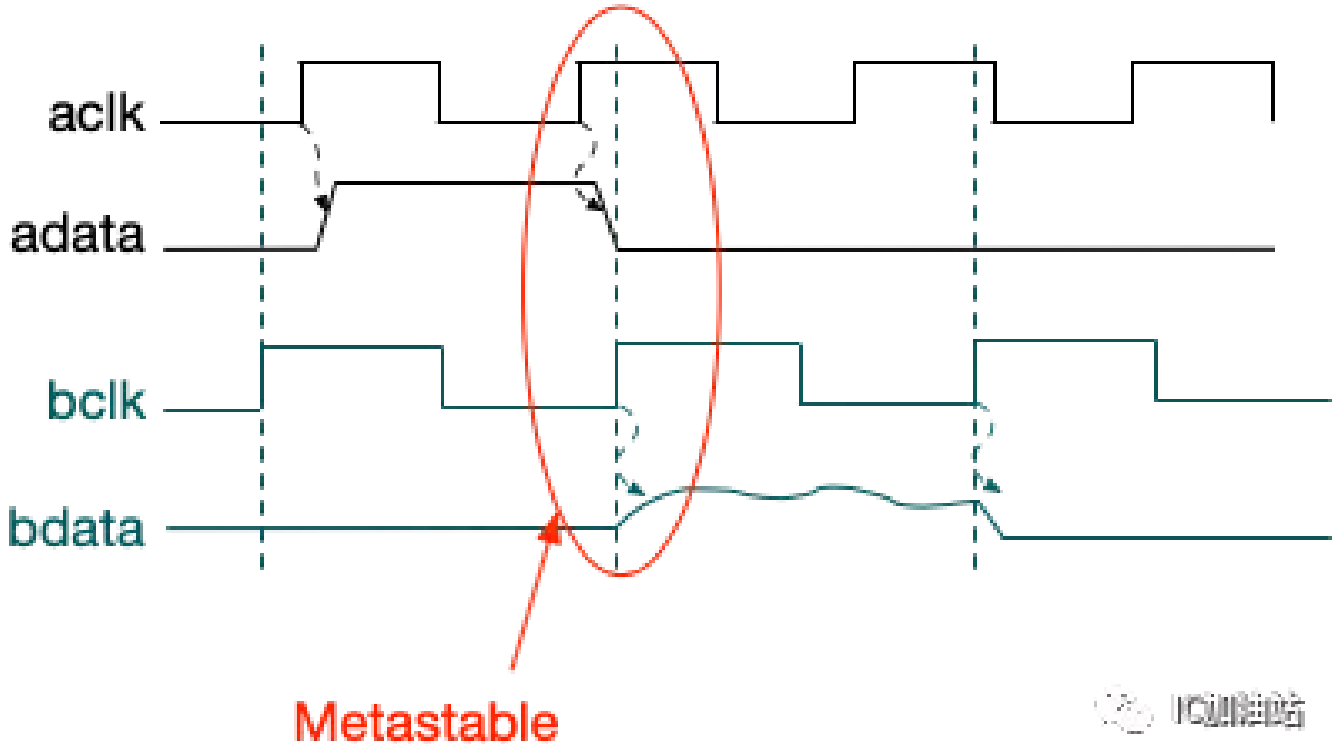
- Q的值可能不是正确的D
- 随着D的变化越靠近时钟沿，Q变稳定的时间越长
- 最后Q稳定到的值可能是随机的

注意我们并不是说Q最后的值不是稳定的1或者0， Q的值最后一定会稳定下来，稳定在高电平或者低电平，这是由于背靠背的反相器会产生正反馈，最终一定会稳定下来。但是当这个稳定的时间超出了clk-to-q的限制，我们就说产生了亚稳态。

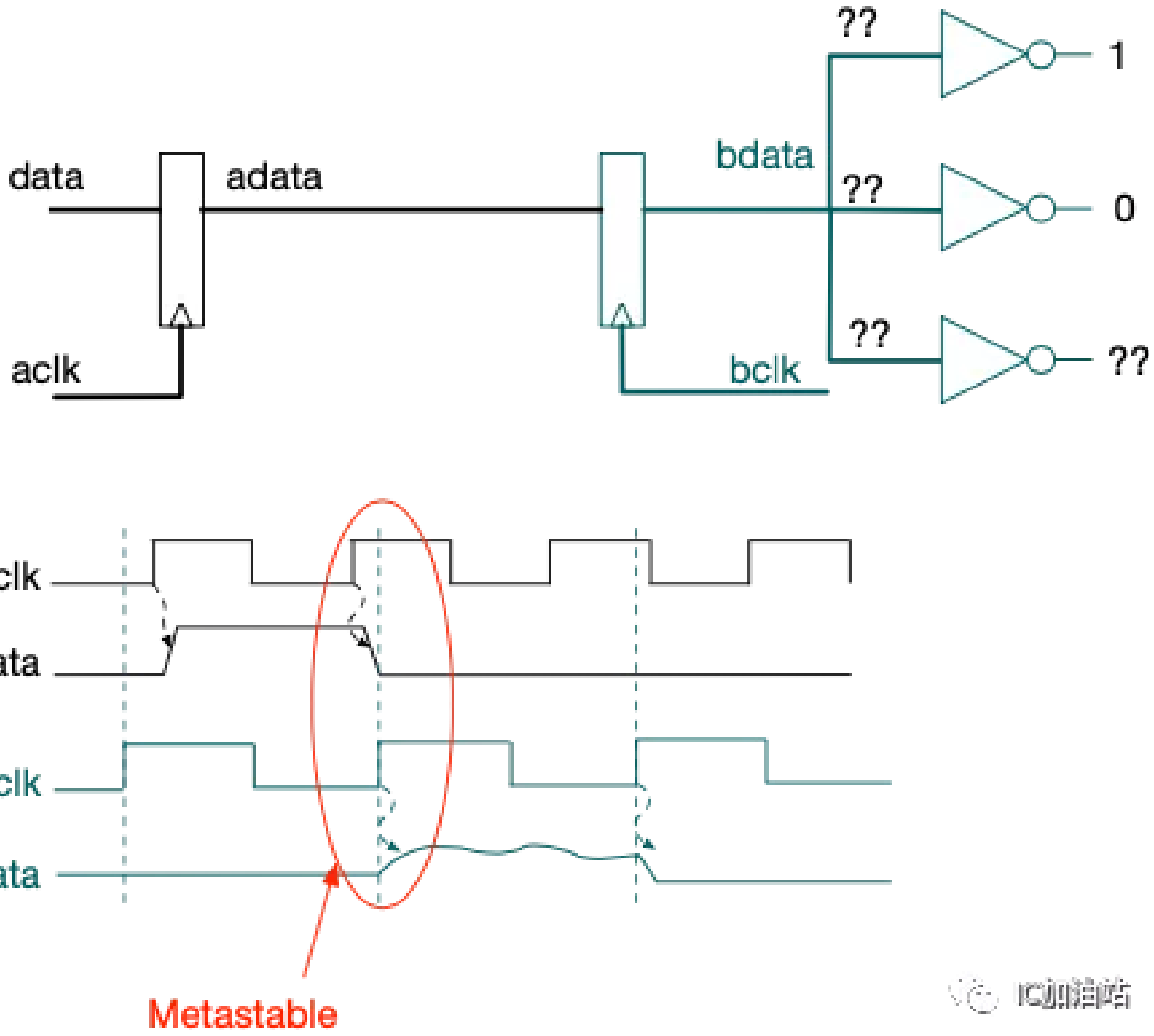


接下来我们就可以正式进入跨时钟域的讨论了。当只有一个时钟存在时，life is simple，只要保证setup/hold time就好了，那么当有多个时钟存在的时候会发生什么呢？我们来看最简单的情况，如下图所示，aclk时钟域的信号需要传输到bclk时钟域去。在各自时钟域内，EDA工具可以保证触发器不会产生metastable，但是当aclk和

bclk异步的时候，我们是无法保证aclk和bclk之间的关系的，也就是说adata相对于bclk的沿来说，可能在任何时候发生变化，这样bdata这个flop就可能产生亚稳态。



当bdata发生亚稳态的时候，会造成什么影响呢？影响主要发生在bclk时钟域的后级电路上，让后级电路无法sample到一个确定的正确的值，进而导致运算逻辑错误。



在这里要澄清的一点是，亚稳态的出现并导致逻辑错误并且芯片失效是一个概率事件，而不是一个100%会发生的确定性事件。这一点可能有点难以理解，举例来说明，很有可能bdata这个flop的后面组合逻辑的delay很小，而这个flop在发生亚稳态之后所需要稳定的时间也很短，这样即使flop发生了亚稳态，而后级的flop的setup time/hold time也可能可以得到满足，这样的话在实际芯片工作中，我们可能观察不到产生错误输出的情况。但也正是这样的原因，很多看似正常工作的芯片内部可能其实有跨时钟域设计上的问题，却从来没有暴露出来。这种情况其实非常危险，因为这种问题一旦出现，则会非常难以debug，因为出现的概率很低，看起来很随机。或者很可能同样的设计换一个工艺，以前可以工作在新的工艺上突然产生问题，造成很严重的后果。所以芯片在设计的时候需要尽量在流片前发现并解决所有的CDC问题，这一点大家要铭记在心。

芯片一旦出现CDC的问题，可能会导致以下后果

- 逻辑功能发生错误，比如控制信号，握手信号错位
- 数据发生错误，比如data bus的值，memory中存的值发生错误
- 发生错误的时间可能随机，很难以复现相同的错误
- 很难以通过软件修复，即使能够修复，也可能需要牺牲性能和功耗

那么我们能够完全消除亚稳态吗？答案是否定的。其实我们关心的并不是亚稳态，而是说能否避免由于亚稳态而造成的逻辑问题。在这里要引入一个MTBF的概念。MTBF--mean time between failure. 意思是两次失效之间的平均时间。简单来说，就是这个芯片或者这个IP或者这个电路发生两次发生错误之间的间隔。对于不同的系统和应用场景，MTBF的要求也不同。比如说对于我们的手机，没有人拿一个手机用二三十年吧？那么如果能够保证MTBF大于30年，那么也等效于在整个手机的使用寿命中，这个逻辑错误不会发生2次，那么**针对这个错误**来说，这样的MTBF是可以接受的。但是对于有些应用场景，比如通讯卫星，一个通讯卫星的寿命可能超过二三十年，那么这种情况下MTBF 如果只有30年，那么就无法接受了。

关于MTBF，还有一个误区需要澄清，就是MTBF只要大于产品的设计使用寿命就可以了，这其实是不严谨的。因为一个产品可能由多个系统组成，每个系统又是由多个子系统组成，每个子系统可能细分下去是由更小的单元组成。整个产品不发生失效的概率是所有部分不发生失效概率的乘积。所以越是小的单元，越要保证MTBF越高，这样才能不会导致整个产品的MTBF 有显著下降。

说回来触发器的MTBF，MTBF的具体公式将在下一篇推送中呈现。这里大家只需要知道，MTBF反比于采样时钟频率(destination clock frequency)，反比于数据变化频率(source data change frequency)，还和工艺、电压、温度等因素相关即可。

在下一篇，老李将带领大家开始学习如何解决跨时钟域产生的各种问题。

如果你喜欢老李的内容，希望大家分享，感谢！