跨越鸿沟：同步世界中的异步信号

只有最初级的逻辑电路才使用单一的时钟。大多数与数据传输相关的应用都有与生俱来的挑战，即跨越多个时钟域的数据移动，例如磁盘控制器、CDROM/DVD 控制器、调制解调器、网卡以及网络处理器等。当信号从一个时钟域传送到另一个时钟域时，出现在新时钟域的信号是异步信号。

在现代 IC、 ASIC 以及 FPGA 设计中，许多软件程序可以帮助工程师建立几百万门的电路，但这些程序都无法解决信号同步问题。设计者需要了解可靠的设计技巧，以减少电路在跨时钟域通信时的故障风险。

基础

从事多时钟设计的第一步是要理解信号稳定性问题。当一个信号跨越某个时钟域时，对新时钟域的电路来说它就是一个异步信号。接收该信号的电路需要对其进行同步。同步可以防止第一级存储单元（触发器）的亚稳态在新的时钟域里传播蔓延。

亚稳态是指触发器无法在某个规定时间段内达到一个可确认的状态。当一个触发器进入亚稳态时，既无法预测该单元的输出电平，也无法预测何时输出才能稳定在某个正确的电平上。在这个稳定期间，触发器输出一些中间级电平，或者可能处于振荡状态，并且这种无用的输出电平可以沿信号通道上的各个触发器级联式传播下去。

对任何一种触发器，在时钟触发沿前后的一个小时间窗口内，输入信号必须稳定。这一时间窗口是多种因素的函数，包括触发器设计、实现技术、运行环境以及无缓冲输出上的负载等。输入信号陡峭的边沿可以将此窗口减至最小。随着时钟频率的升高，会出现更多有问题的时间窗口，而随着数据频率的提升，这种窗口的命中概率则会增加。

FPGA 制造商和 IC 晶片厂用"MTBF"来标识合格的触发器，并且确定它们的特性。"MTBF"（平均无故障时间）用统计方法描述了一个触发器的亚稳态特性，即确定某个触发器出现故障的概率。在计算 MTBF 时，制造商部分基于输入信号改变导致触发器不稳定期间的时间窗口长度。另外，MTBF 的计算还使用了输入信号的频率以及驱动触发器的时钟频率。

在一个 ASIC 或 FPGA 库中，每种触发器都有时序要求，以帮助你确定容易出问题的窗口。"建立时间"(Setup time)是指在时钟沿到来之前，触发器输入信号必须保持稳定的时间。"保持时间"(Hold time)则是指在时钟沿之后，信号必须保持稳定的时间。这些指标通常比较保守，以应对电源电压、工作温度、信号质量以及制造工艺等各种可能的差异。如果一个设计满足了这些时序要求，则触发器出现错误的可能性可以忽略不计。

现代 IC 与 FPGA 设计中使用的综合工具可以保证设计能满足每个数字电路触发器对建立与保持时间的要求。然而，异步信号却给软件提出了难题。对新的时钟域来说，从其它时钟域传来的信号是异步的。大多数综合工具在判定异步信号是否满足触发器时序要求时遇到了麻烦。因为它们不能确定触发器处于非稳态的时间，所以它们也就不能确定从一个触发器通过组合逻辑到达下一个触发器的总延迟时间。所以，最好的办法是使用一些

电路来减轻异步信号的影响。

信号同步

信号同步的目的是防止新时钟域中第一级触发器的亚稳态信号对下级逻辑造成影响。简单的同步器由两个触发器串联而成，中间没有其它组合电路。这种设计可以保证后面的触发器获得前一个触发器输出时，前一个触发器已退出了亚稳态，并且输出已稳定。设计中要注意将两个触发器放得尽可能近，以确保两者间有最小的时滞(clock skew)。

IC 制造厂提供同步单元，帮助完成信号同步工作。这些单元通常包括一个有非常高增益的触发器，它比普通触发器耗电更高，也比较大。这种触发器降低了对输入信号建立-保持时间的要求，并且当输入信号导致亚稳态时，它可以防止出现振荡。另一种同步器单元包括两个触发器，省去了将两个单独触发器靠近放置的工作，也防止设计人员误在两个触发器间加入任何其它的组合逻辑。

为了使同步工作能正常进行，从某个时钟域传来的信号应先通过原时钟域上的一个触发器，然后不经过两个时钟域间的任何组合逻辑，直接进入同步器的第一个触发器中（图 1）。这一要求非常重要，因为同步器的第一级触发器对组合逻辑所产生的毛刺非常敏感。如果一个足够长的信号毛刺正好满足建立-保持时间的要求，则同步器的第一级触发器会将其放行，给新时钟域的后续逻辑送出一个虚假的信号。

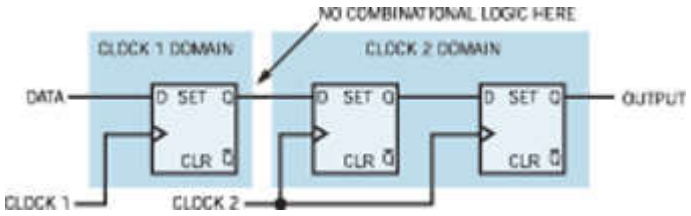

图 1，在一个全同步器电路中，从某个时钟域传来的信号应先通过原时钟域上的一个触发器，然后不经过原触发器和同步器的第一个触发器两个时钟域间的任何组合逻辑，直接进入同步器的第一个触发器中。

一个经同步后的信号在两个时钟沿以后就成为新时钟域中的有效信号。信号的延迟是新时钟域中的一到两个时钟周期。一种粗略的估算方法是同步器电路在新时钟域中造成两个时钟周期的延迟，设计者需要考虑同步延迟将对跨时钟域的信号时序造成的影响。

同步器有许多种设计方法，因为一种同步器不能满足所有应用的需求。同步器的类型基本上有三种：电平、边沿检测和脉冲（表 1）。虽然还存在着其它类型的同步器，但这三种类型的同步器可以解决设计者遇到的多数应用问题。在电平同步器中，跨时钟域的信号在新时钟域中要保持高电平或低电平两个时钟周期以上。这种电路的要求是，在再次成为有效信号前，信号需要先变成无效状态。每一次信号有效时，接收逻辑都会把它看作一个单个事件，而不管信号的有效状态保持了多久。这种电路是所有同步器电路的核心。

表1 同步器的类型与应用

Type	Application	Input	Output	Restriction
Level signals	Synchronizes level	Level	Level	Input must be valid for at least two clock periods in the new domain. Each time output goes valid counts as a single event.
Edge-detecting	Detects rising or falling edge of input	Level or pulse	Pulse	Input must be valid for at least two clock periods in the new domain.
Pulse	Synchronizes single clockwide pulses	Pulse	Pulse	Input pulses must have at least two clock periods between them in the new domain.

表 1

边沿检测同步器在电平同步器的输出端增加了一个触发器（图 2）。新增触发器的输出经反相后和电平同步器的输出进行与操作。这一电路会检测同步器输入的上升沿，产生一个与时钟周期等宽、高电平有效的脉冲。如果将与门的两个输入端交换使用，就可以构成一个检测输入信号下降沿的同步器。将与门改为与非门可以构建一个产生低电平有效脉冲的电路。

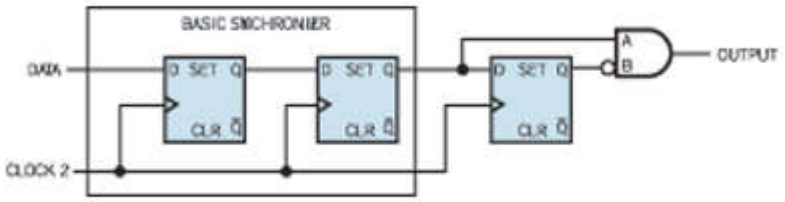

图 2，边沿检测同步器在电平同步器的输出端增加了一个触发器。

当一个脉冲进入更快的时钟域中时，边沿检测同步器可以工作得很好。这一电路会产生一个脉冲，用来指示输入信号上升或下降沿。这种同步器有一个限制，即输入脉冲的宽度必须大于同步时钟周期与第一个同步触发器所需保持时间之和。最保险的脉冲宽度是同步器时钟周期的两倍。如果输入是一个单时钟宽度脉冲进入一个较慢的时钟域，则这种同步器没有作用，在这种情况下，就要采用脉冲同步器。

脉冲同步器的输入信号是一个单时钟宽度脉冲，它触发原时钟域中的一个翻转电路（图 3）。每当翻转电路接收到一个脉冲时，它就会在高、低电平间进行转换，然后通过电平同步器到达异或门的一个输入端，而另一个信号经一个时钟周期的延迟进入异或门的另一端，翻转电路每转换一次状态，这个同步器的输出端就产生一个单时钟宽度的脉冲。

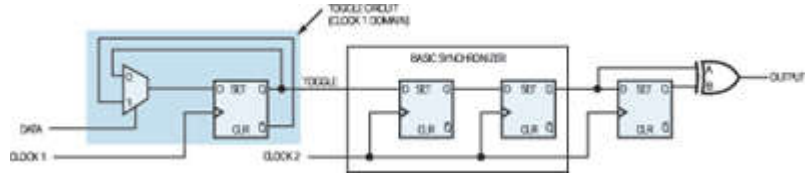

图 3，脉冲同步器的输入信号是一个单时钟宽度脉冲，它触发原时钟域中的一个翻转电路

脉冲同步器的基本功能是从某个时钟域取出一个单时钟宽度脉冲，然后在新的时钟域中建立另一个单时钟宽度的脉冲。脉冲同步器也有一个限制，即输入脉冲之间的最小间隔必须等于两个同步器时钟周期。如果输入脉冲相互过近，则新时钟域中的输出脉冲也紧密相邻，结果是输出脉冲宽度比一个时钟周期宽。当输入脉冲时钟周期大于两个同步器时钟周期时，这个问题更加严重。这种情况下，如果输入脉冲相邻太近，则同步器就不能检测

到每个脉冲。

握手与 FIFO

在许多应用中，跨时钟域传送的不只是简单的信号，数据总线、地址总线和控制总线都会同时跨域传输。工程师们用一些其它的手段来处理这些情况，如握手协议和 FIFO 等。

当几个电路不能预知相互的响应时间时，握手方法能让数字电路间实现有效的通信。例如，仲裁总线结构可以让一个以上的电路请求使用单个的总线，用仲裁方法来决定哪个电路可以获得总线的访问权，例如 PCI 或 AMBA（高级微控制器总线架构）。每个电路都发出一个请求信号，由仲裁逻辑决定谁是"赢家"。获胜的电路会收到一个应答，表示它可以访问总线。该电路于是中断请求，开始使用总线。

不同时钟域电路使用的握手协议有两种基本类型：全握手(Full-handshake)和部分握手(partial-handshake)。每种类型的握手都要用同步器，每种都各有自己的优缺点。对全握手信号，双方电路在声明或中止各自的握手信号前都要等待对方的响应（图 4）。首先，电路 A 声明它的请求信号，然后，电路 B 检测到该请求信号有效后，声明它的响应信号。当电路 A 检测到响应信号有效后，中止自己的请求信号。最后，当电路 B 检测到请求无效后，它中止自己的响应信号。除非电路 A 检测到无效的响应信号，否则它不会再声明新的请求信号。

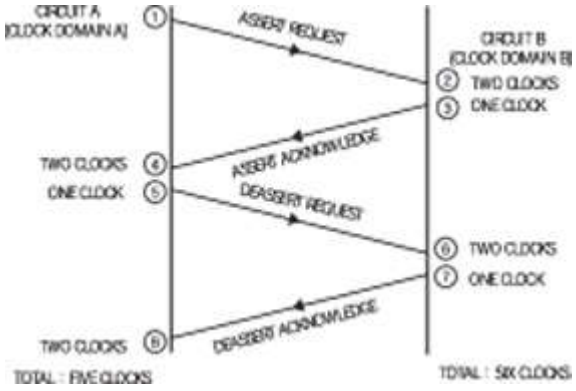

图 4，对全握手信号，双方电路在声明或中止各自的握手信号前都要等待对方的响应。

这种类型的握手使用了电平同步器。设计人员将这种技术用在如下情况：响应电路（电路 B）需要告知请求电路（电路 A）它可以处理请求。这种握手方法要求请求电路延迟它的下一个请求，直到它检测到响应信号无效。可以用经验估算法判断这个协议的时序：信号跨越一个时钟域要花两个时钟周期的时间，信号在跨越多个时钟域前被电路寄存。全部的时间序列是：A 时钟域中最多五个周期加上 B 时钟域最多六个周期。全握手类型很强健，因为通过检测请求与响应信号，每个电路都清楚地知道对方的状态。这种方式的不足之处是完成所有交互的整个过程要花费很多时钟周期。

另一种类型是部分握手，它可以缩短这些事件的过程。使用部分握手信号时，通信双方的电路都不等对方的响应就中止各自的信号，并继续执行握手命令序列。部分握手类型比全握手类型在健壮性方面稍弱，因为握手信号并不指示各自电路的状态，每一电路都必

须保存状态信息（在全握手信号里这个信息被送出去）。但是，由于无需等待其它电路的响应，完整的事件序列花费时间较少。

当使用部分握手信号方式时，响应的电路必须以正确的时序产生它的信号。如果响应电路要求先处理完一个请求，然后才能处理下一个请求，则响应信号的时序就很重要。电路用它的响应信号来指示它的处理任务何时完成。一种部分握手方法混合了电平与脉冲信号，而其它的方法则只使用脉冲信号。

在第一种部分握手方法中，电路 A 以有效电平声明其请求信号，电路 B 则以一个单时钟宽度脉冲作为响应。此时，电路 B 并不关心电路 A 何时中止它的请求信号。但为了使这种方法成立，电路 A 中止请求信号至少要有一个时钟周期长，否则，电路 B 就不能区别前一个请求和新的请求。在这种握手方式下，电路 B 为请求信号使用一个电平同步器，电路 A 为响应信号使用一个脉冲同步器。只有当电路 B 检测到请求信号时才发出响应脉冲。这种情况可以使电路 A 通过控制其请求信号的时序，控制同步器接收到的脉冲间隔（图 5）。同样可以用经验估算法确定时序，即信号跨越一个时钟域要花两个时钟周期并且在跨越时钟域前被电路寄存。

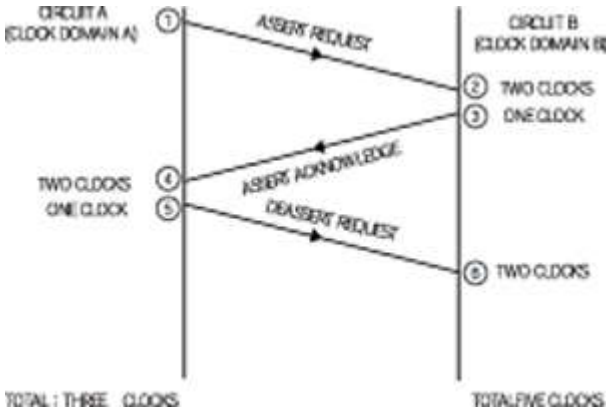

图 5，在一种部分握手方法中，电路 A 发出它的请求信号，电路 B 则以一个单时钟宽度脉冲作为响应。

全部的序列为 A 时钟域最多三个周期加上 B 时钟域最多五个周期。这种部分握手方法比全握手方法在 A、B 两个时钟域分别少用了两个和一个时钟周期。如果采用第二种部分握手方法可以再减少一些时钟周期，此时电路 A 用一个单时钟宽度脉冲发出它的请求，而电路 B 也用一个单时钟宽度脉冲响应这个请求。这种情况下，两个电路都需要保存状态，以指示请求正待处理。

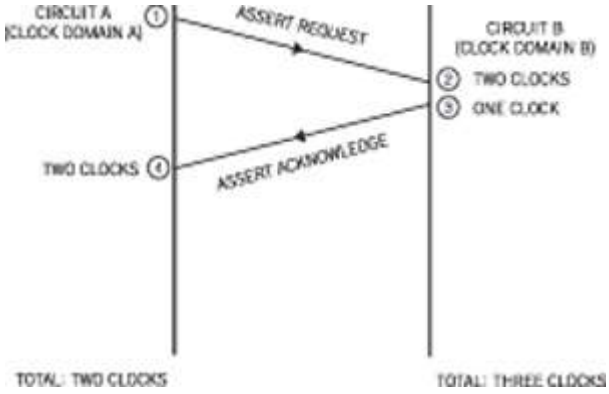

图 6，这种握手类型使用的是脉冲同步器，但如果其中一个电路时钟比另一个电路时钟快两倍，则可以用边沿检测同步器来代替。

这种握手类型使用的是脉冲同步器，但如果其中一个电路时钟比另一个电路时钟快两倍，则可以用边沿检测同步器来代替（图 6）。完整的时序是：A 时钟域最多两个周期加上 B 时钟域最多三个周期。所以这种部分握手技术与全握手方法相比，在 A 时钟域少用三个时钟周期，在 B 时钟域也少用三个时钟周期。同时，也比第一种部分握手方法分别在 A、B 时钟域快了一个和两个周期（表 2）。这些握手协议针对的都是跨越时钟域的单一信号。但当几组信号要跨越时钟域时，设计人员就需要使用更加复杂的信号传送方法。

表2 握手技术

Handshake type	Circuits	Signaling type	Sequence length	Synchronizer	Restrictions
Full	Circuit A (request)	Level	Five clocks	Level	Sequence is long. Request must be invalid for at least two of the Circuit B clock periods.
	Circuit B (acknowledge)	Level	Six clocks	Level	Acknowledgment must be invalid for at least two of the Circuit A clock periods.
Partial I	Circuit A (request)	Level	Three clocks	Pulse or edge-detect	Must control rate of acknowledgment pulses.
	Circuit B (acknowledge)	Pulse	Five clocks	Level	Request must be invalid for at least two of the Circuit B clock periods.
Partial II	Circuit A (request)	Pulse	Two clocks	Pulse or edge-detect	Must save pending request information.
	Circuit B (acknowledge)	Pulse	Three clocks	Pulse or edge-detect	Must register request and acknowledgment signals.

表 2

数据路径设计

在进行信号同步时有一个重要的规则，那就是不应当在设计中的多个地方对同一信号进行同步，即单个信号扇出至多个同步器。因为同步要花一到两个时钟周期，设计者不能确切地预测到每个信号何时跨越一个时钟域。此外，在新时钟域中一组经同步后的信号其时序是不定的，因为同步延迟可以是一到两个时钟周期，这与输入信号到达同步器的时间有关。这种情况会在各个同步信号间形成一种"竞争状况"。这种竞争状况在需要跨越时钟域传输的多组信号间也会发生，例如数据总线、地址总线和控制总线等。因此，不能对组中的每个信号单独使用同步器，也不能对数据或地址总线的每一位单独使用同步器，因为在新的时钟域中，要求每个信号同时有效。

一种解决总线同步问题的方法是使用一个保持寄存器和握手信号。这种电路包括一个保持信号总线的寄存器，以及一个握手机制（图 7）。握手信号指示新时钟域的电路何时可以对总线采样，以及源电路何时可以更换当前寄存器中保存的内容。

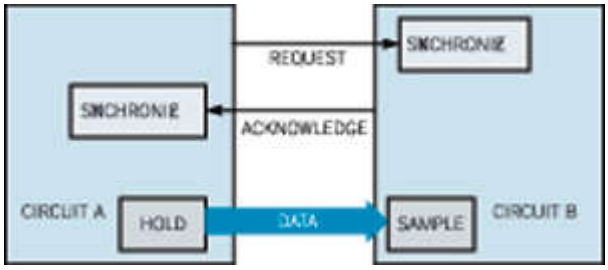

图 7，一种数据路径同步器设计使用一个保持寄存器和握手信令。

在这种设计中，传输电路将数据（信号总线）存储在保持寄存器，同时发出请求信号。这两个动作可以同时发生，因为请求信号至少要花一个时钟周期才能让接收电路检测到它（最小的握手-同步延迟）。当接收电路采样到数据（信号总线）时，它发出一个响应信号。这种设计使用了全握手方法，所以要花较长时间才能完成整个传输。对接收电路而言，使用全握手信号的设计有较大的时间窗口用于对信号总线采样，因而效率较低。如用部分握手方法代替全握手方法则可以加快传输速度。

用这种总线同步方式，你可以同步握手信号，但不能同步信号总线。信号总线来自于保持寄存器，它在接收电路采样前一直保持稳定。注意，如果传输电路向接收电路提交数据太快以致来不及处理，则应用中的总线同步可能不起作用。

高级数据路径设计

在许多情况下，数据在跨越时钟域时需要"堆积"起来，因此使用单个保持寄存器无法完成工作。例如一种情况是某个传输电路猝发式发送数据，接收电路来不及采样。另一种情况是接收电路采样速度超出传输电路发送数据的速度，但采样的数据宽度不够。这些情况就要使用 FIFO 了。

基本上，设计者使用 FIFO 有两个目的：速度匹配或数据宽度匹配。在速度匹配时，FIFO 较快的端口处理猝发的数据传输，而较慢的端口则维持恒定的数据流。但是，虽然访问方式和速度不同，但进出 FIFO 的平均数据速率必须是相同的，否则 FIFO 就会出现上溢(over flow)或下溢(underflow)问题。与单寄存器设计相同，FIFO 将数据保存在寄存器或存储器中，同时同步状态信号，判断何时可以把数据写入 FIFO 或从 FIFO 中读出。

在速度匹配应用中，每个端口（读或写）的时钟不同。FIFO 中的寄存器使用写端口时钟，就像保持寄存器使用电路时钟来改变寄存器内容一样。信号同步发生在指针逻辑中，而且比握手信号要复杂得多。

现在指针逻辑的设计有多种方法。第一种方法是将读、写选通进行同步，同时在各个时钟域使用计数器来跟踪 FIFO 中可用的项。计数器反映出可用于读写的 FIFO 项目号，计数器也与相应的端口同步。读计数器跟踪包含有效数据的项数，而写计数器则跟踪可以存储数据的项数。当对指针逻辑进行复位时，由于没有数据可读，读计数器从零起始。写计数器则从 FIFO 中项的总数开始计数，即所有项均可用来存储数据。

读选通信号累减读计数器，并与写时钟域同步，因为它同时也累加写计数器。写选通信号则累减写计数器，并与读时钟域同步，因为它同时也累加读计数器。

这种设计需要单时钟宽度脉冲以及用于读、写选通的脉冲同步器，因为当一个电平信号从一个时钟域跨越到另一个更快的时钟域时，在较快时钟域中它能在更多的时钟周期中保持有效。由于只要读或写信号是有效的，每个计数器就会发生变化，因此较快的时钟域就检测到更多的读、写，超出较慢时钟域实际发生的数量。脉冲同步器可以将一个时钟域的时钟宽度脉冲转换为新时钟域的时钟宽度脉冲，每个脉冲都表示一次 FIFO 的读或写。

这种 FIFO 状态技术对读、写状态都不太有利。当 FIFO 中所有项均充满时，写端口状态指示为满，并在读选通触发后继续指示 FIFO 满，因为同步过程会使选通信号延迟送给写计数器。读端口为空时也会出现这种情况，因为同步过程会使写选通信号延迟到达读计数器。

这种设计的另一种考虑是及时检测全满/全空状态。如果 FIFO 还有一项可用，并且有写选通触发，则 FIFO 必须立即置为全满状态。这样才能提前一个时钟给出全满标志，使 FIFO 有足够时间防止下一个数据写入而产生溢出。对 FIFO 的读端口也是这样。这种情况下，如果 FIFO 里只有一个数，并且有读选通触发，则必须置全空状态，以给读电路足够的时间防止读空 FIFO。

这种指针逻辑限制电路在每个时钟周期中访问 FIFO，即使在慢速时钟域中也是这样。这一功能的优点在于访问 FIFO 的电路至少有一个时钟周期来评估 FIFO 的状态。FIFO 可以将所有项都填满数据，而不会出现数据被覆盖或全空无数据可读的情况。这种设计的另一个优点是每一端都可以读其相应的计数器，来判断 FIFO 中还有多少项可用。设计者可以将这种 FIFO 设计用在进行多次数据读/写的电路中，而不会造成上溢或下溢的情况。

这种设计的不足之处是由计数器来判断状态，而不是直接比较读、写指针。对大型 FIFO 来说，这些计数器也很大。而且，由于使用脉冲同步时，来自较快时钟域的读、写脉冲在较慢时钟域的脉冲间至少必须有两个时钟周期，因此平均数据速率为最低时钟频率的一半。解决这些问题的一种方法是采用直接指针比较法。

在这种 FIFO 设计中，读、写指针的比较决定了 FIFO 的状态。异步设计中的指针比较更富有挑战性，因为每个指针位于不同的时钟域中，对信号总线的同步要求在同步握手信号期间总线不发生改变。将这种技术用于指针同步的 FIFO 设计可能会很慢。要解决这个问题，FIFO 指针逻辑使用了格雷码，代替指针使用的二进制码。

格雷码在每一次计数增减时只改变其中的一位（表 3）。你可以在格雷码总线上使用同步器，因为每一次总线改变时只有一根信号线有变化，于是就消除了格雷码总线各位通过不同同步器时的竞争情况。这种设计的指针为格雷码计数器。使用二进制指针时需要将其变换成格雷码后的同步指针，而使用变换逻辑会违反对同步信号的限制，即同步的信号在跨越时钟域前要来自触发器。

表3 格雷码与二进制码比较

Decimal	Binary	Gray
0	0	0
1	1	1
2	10	11
3	11	10
4	100	110
5	101	111
6	110	101
7	111	100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

表 3

格雷码计数器是一个二进制累加器，在累加器前、后各带有一个转换器，分别用于格雷码转换为二进制码，和二进制码转换为格雷码（图 8）。格雷码与二进制码的转换是一个异或运算，所以只需比一个二进制计数器多几个逻辑电平。在格雷码转换成二进制码时，使用：BN=GN；BN-1=BN+GN-1；BN-2=BN-1+GN-2 ．B1=B2+G1；B0=B1+G0。而将二进制码转换成格雷码时，使用：GN=BN；GN-1=BN+BN-1；GN-2=BN-1+BN-2 ．G1=B2+B1；G0=B1+B0。在设计中可以采用同样的技术来比较格雷码指针的值，即在各个指针与二进制比较逻辑之间增加转换器。

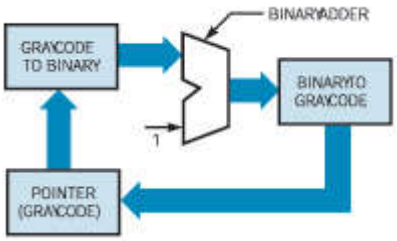

图 8，格雷码计数器是一个二进制累加器，在累加器前、后各带有一个转换器。

用这种指针逻辑的 FIFO 很快，每个时钟周期中电路都可以读写 FIFO。但是，在每个周期都访问 FIFO 意味着 FIFO 状态要包括"将满"和"将空"两种指示，这样读写 FIFO 的电路才能有停止时间。"将满"表示只能再写入一项，"将空"则表示只有一项可读。这种情况描述了一个要求最少的可能状态信号的设计，以及一个需要更多指示的设计，如果在固定的最小尺寸情况下用猝发方式访问 FIFO 的电路的话。

这种 FIFO 状态技术会给读、写带来不良状态。当 FIFO 满时，写端口的状态指示已满，而在电路从 FIFO 中读出一项后，该状态仍为满，因为同步机制使读指针相对写入一侧的比较逻辑有个延迟。同样，在读出一侧的空状态指示也有这个问题，因为同步机制使写指针相对读出一侧的比较逻辑有延迟。

如果你在设计跨不同时钟域电路时，使用一些技术来降低通信失败的风险，则处理跨

时钟域的信号就不再是艰巨的任务。同步机制可以防止接收跨时钟域信号的触发器出现亚稳态，从而避免导致不可预知的电路行为。对于在多个时钟周期内一直保持有效的信号来说，电平同步器的效果很好。对于要转换成新时钟域脉冲的较慢时钟域电平信号，要采用边沿检测同步器。最后，对跨时钟域的脉冲信号应使用脉冲同步器。还要记住，当一个信号总线跨越时钟域时，整个总线要在同一个时钟周期内到达新的时钟域。不要分别同步每一个信号，而要采用一个保持寄存器和握手方式。握手用来表示寄存器中的信号何时有效，何时可以采样。对数据总线来说，握手和一个保持寄存器很有用，但每次向新时钟域传送的数据字不超过一个。