Async\_fifo 模块

# 作用

实现读写不同时钟下，对fifo中实现在各自读写时钟下的异步的数据存取

# 参数说明

module top\_asyn\_fifo(

input wclk, //写时钟

input rclk, //读时钟

input wrst, //写复位信号

input rrst, //读复位信号

input wena, //写请求信号

input rena, //读请求信号

input [`DATA\_BIT - 1: 0] w\_data, //写数据

output[`DATA\_BIT - 1: 0] r\_data, //读数据

output full, //满标志

output empty //空标志

);

# 设计流程

异步读写不同时钟的FIFO的RTL视图如图1所示，其status内部结构[1]如图2所示。当每次要读写FIFO时，需要将读写指针同步到一个时钟频率，为了解决不同时钟频率数据采样时的亚稳态问题，需要在同步的方式，由于涉及到多控制信号的不同时钟域的传输，需要采用编码的方式，减小其传递过程中延迟造成的采样问题，除此之外，由于读写指针采用累加器的方式进行修改，可以采用格雷码编码的方式来解决多控制信号不同时钟域延迟的问题。综上所述，初始情况下，读写使能信号，满标志位（高电平有效）为低电平，空标志（高电平有效）为高电平，读写指针为0。当一个读信号使能时，需要将当前的写指针（格雷码）同步到读时钟，为了能比较其空满标志需要将格雷码转换成二进制，如果全部位都相同，则为空，如果高位不同，其他位都相等则为满，根据空满标志，确定读写指针是否需要修改。读写指针的修改通过一个累加器来实现，累加器是基于二进制的运算。所以在传输到不同时钟域时需要将二进制转换成相应的格雷码。

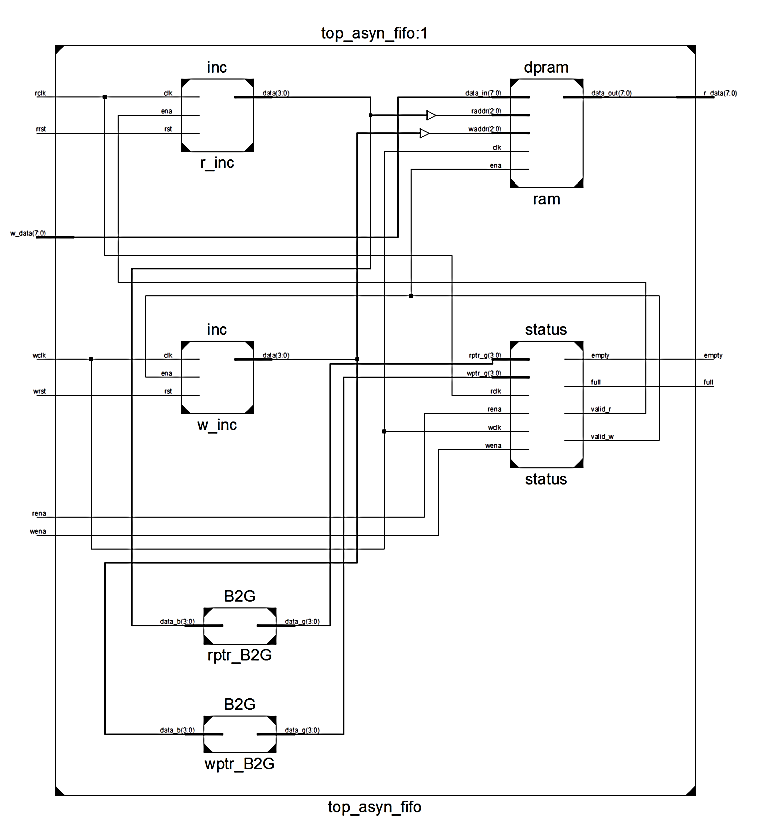


图1 异步FIFO RTL视图

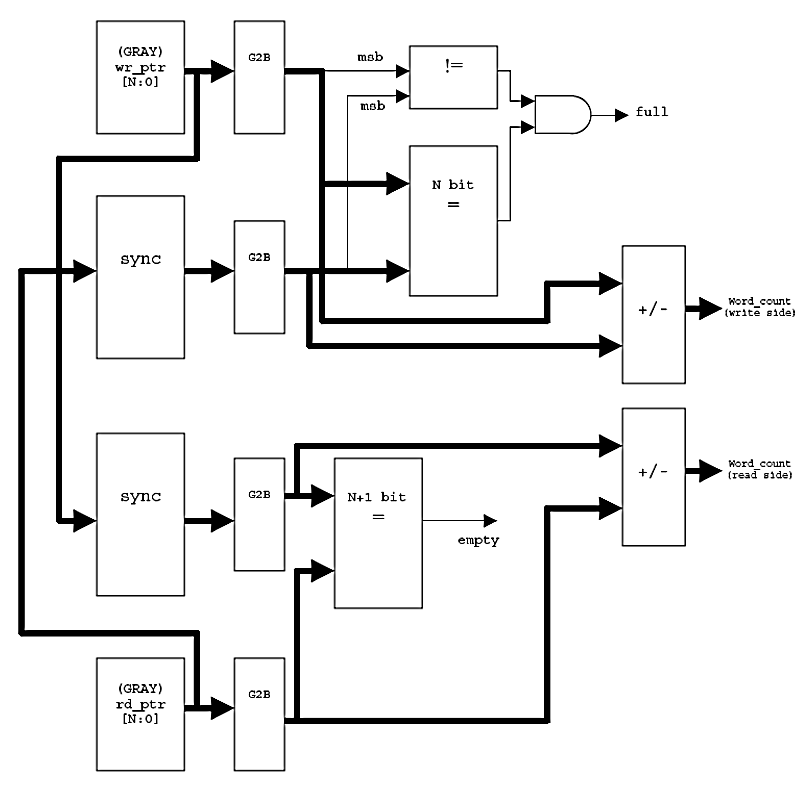


图2 status内部结构

# 设计要点

1．亚稳态问题

异步FIFO的设计重点在于，读写指针运行在不同的时钟之下，这种跨时钟的运行方式，有可能造成fifo在没有数据的时候读，在fifo满时继续往里面写数据。为了解决这种跨时钟而造成的亚稳态问题。

亚稳态：发生在一个事件试图取样另一个事件。举一个具体的实例，一个D触发器，在t时刻，数据总线从0跳变成1，那这个t时候输出的数据究竟是多少。这就好比一个球在一个山峰上，一旦有任何的风吹草动将会有不确定的结果，这种不确定性这就是亚稳态。对于一个D触发器而言，具有setup time和hold time，如图3所示，只有保证setup time和hold time才能保证有正确的输出，否则就会造成亚稳态的情况。

Setup time：触发器时钟上升沿到来之前，数据稳定不变的时间

Hold time：触发器时钟上升沿到来之后，数据稳定不变的时间

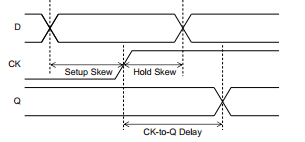


图4 触发器setup time和hold time

避免亚稳态的传播的方法一般采用双同步电路，即由两个触发器简单的组合在一起如图5所示，当第一级电路无法满足建立时间和保持时间而进入到亚稳态之后，需要一段很长的时间才能从亚稳态变成稳态，但是这个时候的稳态并不一定是输入的值，采用两级同步电路正好给了亚稳态输出一个足够长的时间，即一个clk来稳定其输出值，这样在第二级采样时，可以满足其建立时间，从而避免了亚稳态的传播，但是由于第一级产生了亚稳态，所以最后的输出不一定等于一开始的输入。

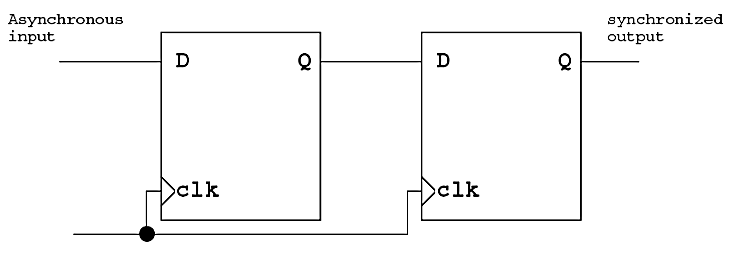


图5 双同步电路

1. 多控制信号跨时钟传播[2]

在计算空和满标志时，需要比对读写指针的值，由于读写采用两个不同的时钟，不可避免的要在异步时钟情况下进行取值，这就涉及到了多控制信号在跨时钟中的传播的问题。由于多控制信号可以能传播的过程中存在时延，导致某个或者多个控制信号处于亚稳态，所以为了尽可能避免这种状况，采用编码的方式，而格雷码作为最小距离码，能很好地避免由于亚稳态导致输出值和输入值不同而导致最终的逻辑判断错误的问题。

格雷码：最小距离码，相邻码元之间只有1位不同

采用格雷码，是一种对于亚稳态发生的处理方法，它使得错误结果取到安全侧的值，避免重大故障的发生。对于格雷码而言，相邻码元之间只有1位是不同的，所以在异步时钟采样的过程中，如果发生亚稳态问题，在其稳定之后只会出现两种情况，即相邻码元中的一种，而不会遇到一些不确定的情况。比如说对于读而言，如图6所示，当前的wptr是0001（格雷码编码，上一次的rptr是0000），rptr是0000，由于读写指针处于不同的时钟域，需要将写指针同步到读时钟，如果发生亚稳态状况，那写指针是0000或者0001（因为由于前三位在异步时钟采样处未发生翻转，保证了足够多的setup time和hold time，所以不会发生亚稳态，而最后一位在异步时钟采样处发生翻转，如果发生亚稳态的情况的话，会产生0和1两种不确定的情况，其中0为错误情况，而1为正确情况）。如果写指针是0000，则为错误的情况，对于读而言，相对于实际情况，看上去像少写了一个数，则满足了空的判断条件，而实际上读指针并没有变，所以读出的值还是原来写入的值。如果写指针是0001，则是正确的情况。采用格雷编码可以很好的屏蔽错误情况，从而使其进入安全侧，起到错误可控的局面。

图6 fifo读情况下安全侧的处理

1. 循环FIFO空和满的判断[1]

目前针对FIFO空满的判断，大致上有两种解决方案，一种是采用对读写指针编码的方式，另一种是采用空满趋势标志的方式，第一种方案较为简单，实现也比较简单，且效果比较理想，所以在本实验中采用第一种方案。

我们需要构造一个指针宽度为N+1的，深度为2N的FIFO，采用二进制编码。基本比较方法是，当指针的最高位不一致而其他位都相等时，FIFO满；当指针完全相等时，FIFO为空。（对于2N的FIFO而言，其需要N位地址位来标志其读写指针，但是对于空和满的情况而言，其读写指针的判断条件都为相等，为了区分空和满状况，需多1位地址位来区分，所以最终采用N+1位地址）

比如一个深度为8字节（采用字节编织）的FIFO而言，其指针宽度为N+1=3+1=4。一开始，r\_ptr和w\_ptr为0000，如果往FIFO中写入8个字节的数据，则w\_ptr=1000，r\_ptr=0000，这就是满条件。如果现在从FIFO中读出8个字节的数据，则w\_ptr=1000，r\_ptr=1000，这就是空条件。

**References:**

[1]. Nebhrajani, V.A., 异步FIFO结构, 2006.

[2]. 刘波, 设计异步多时钟系统的综合及描述技巧, in 精通Verilog HDL语言编程. 2007.