**同步FIFO与异步FIFO**

1. 同步FIFO

FIFO表示先进先出的意思。是基于RAM的存储模块，一般多是用于缓冲数据，令模块独立，调用方便、随性。特点是没有外部的读写地址。没有外部的地址信号，所以只能顺序的读写，不能跳读。学习FIFO最重要的如何建立先进先出的机制。

图1：先用一个简单的示意图说明：



图1

左边有写入使能Write，然后写入数据 Data\_Write，还有写满标志位full。右边则有读使能Read，读出数据Data\_Read，还有读空标志位empty。Write必须拉高Data\_Write才能写入，一旦FIFO写满，那么full就会拉高；Read必须拉高，数据才能经由Data\_Read读出，一旦FIFO读空，empty就会拉高。

难点是如何判断写满和读空：

首先要知道FIFO的运作机制。

建立一个4bit,深度为4的RAM，

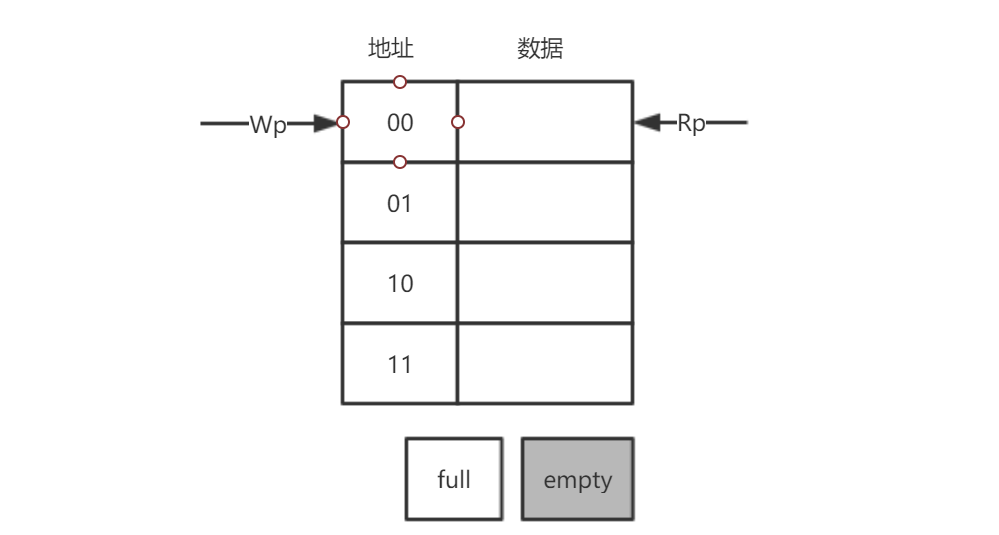


图2

Wp表示下一个将要去写的地址，比如现在还没开始，它就会一直停在00的位置等待

Rp表示下一个将要去读的地址，

1.此时图2所示为空，可以推断当Wp==Rp时为空。

empty = ( Wp==Rp );

2.当有数据输入时，如图3

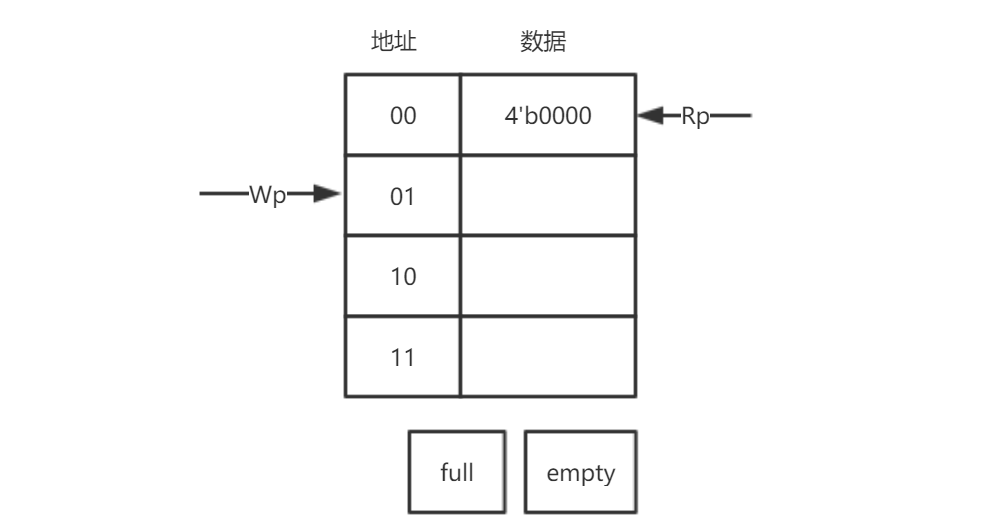


图3

此时可以发现Wp会指向下一个地址，等待数据，而Rp则指向将要读取的数据，Wp和Rp不再相等，整个RAM不满也不空。如图4：

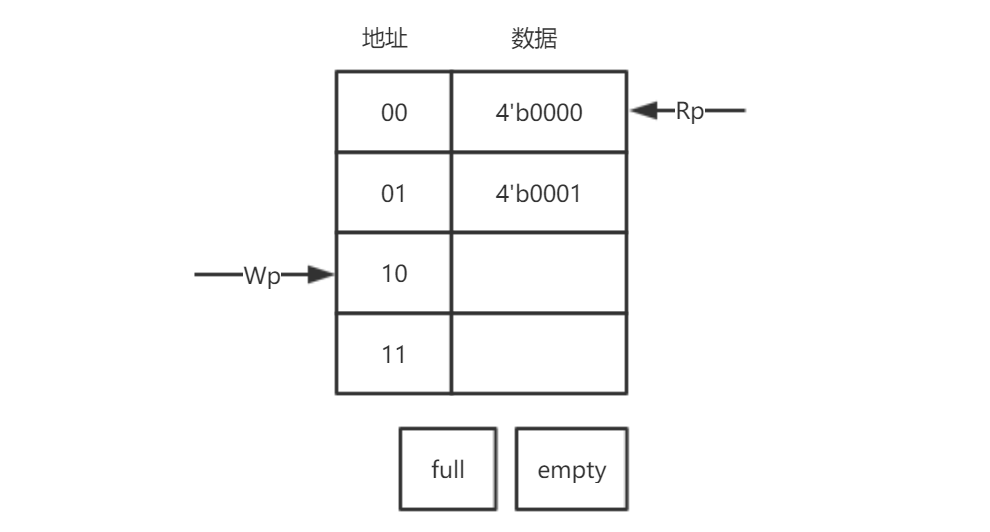


图4

那么，什么时候会满，可以发现在数据写满整个RAM后，Wp会指向下一个地址，即就是在一个循环之后Wp会回到第一个位置，如图5：

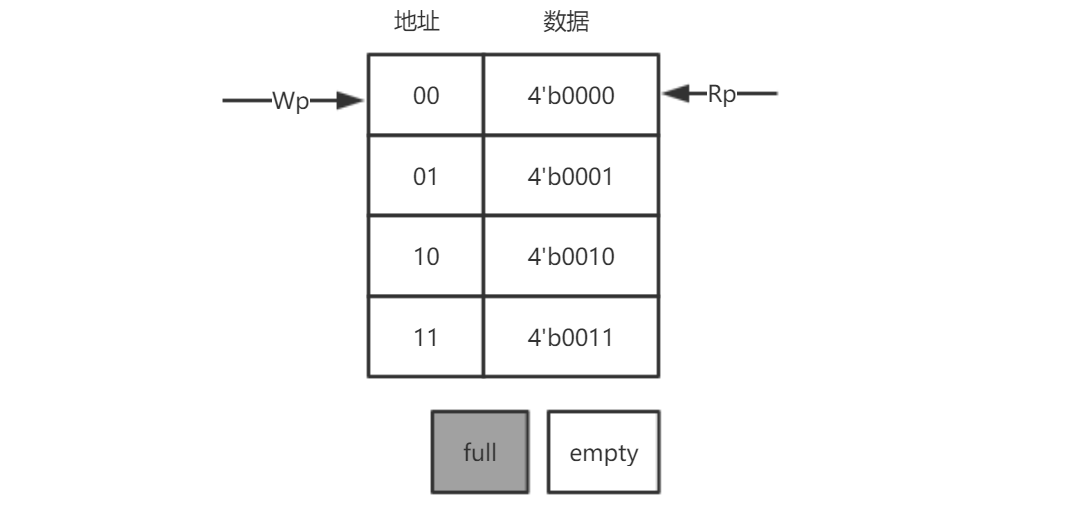


图5

此时，就无法得出full时的关系式。根据黑金原创教程（FPGA那些事儿）中提到，我们可以给Wp和Rp加一位，当一个循环后，给最高位加1，如图6

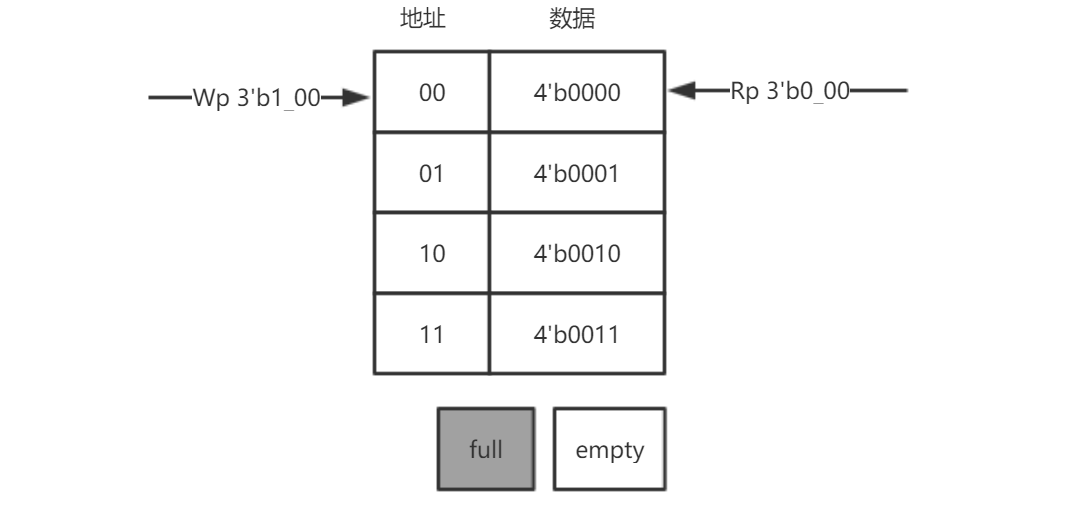


图6

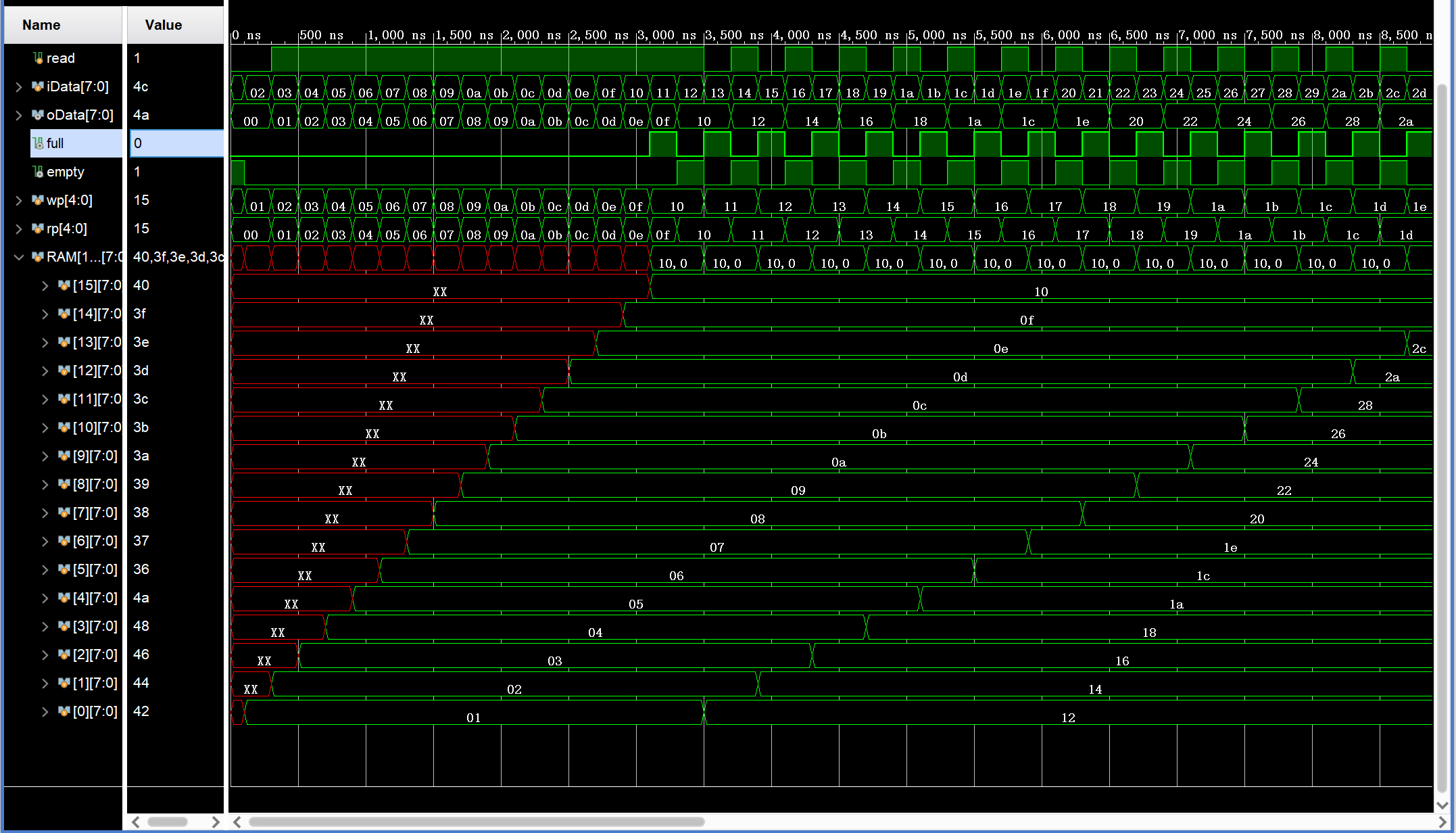
原本Wp和Rp只需2位即可（），现在加一位就可以表现出一轮循环后的标志。所以当Wp和Rp需要N位时，我们要加一位高位，即Wp和Rp是N+1位。

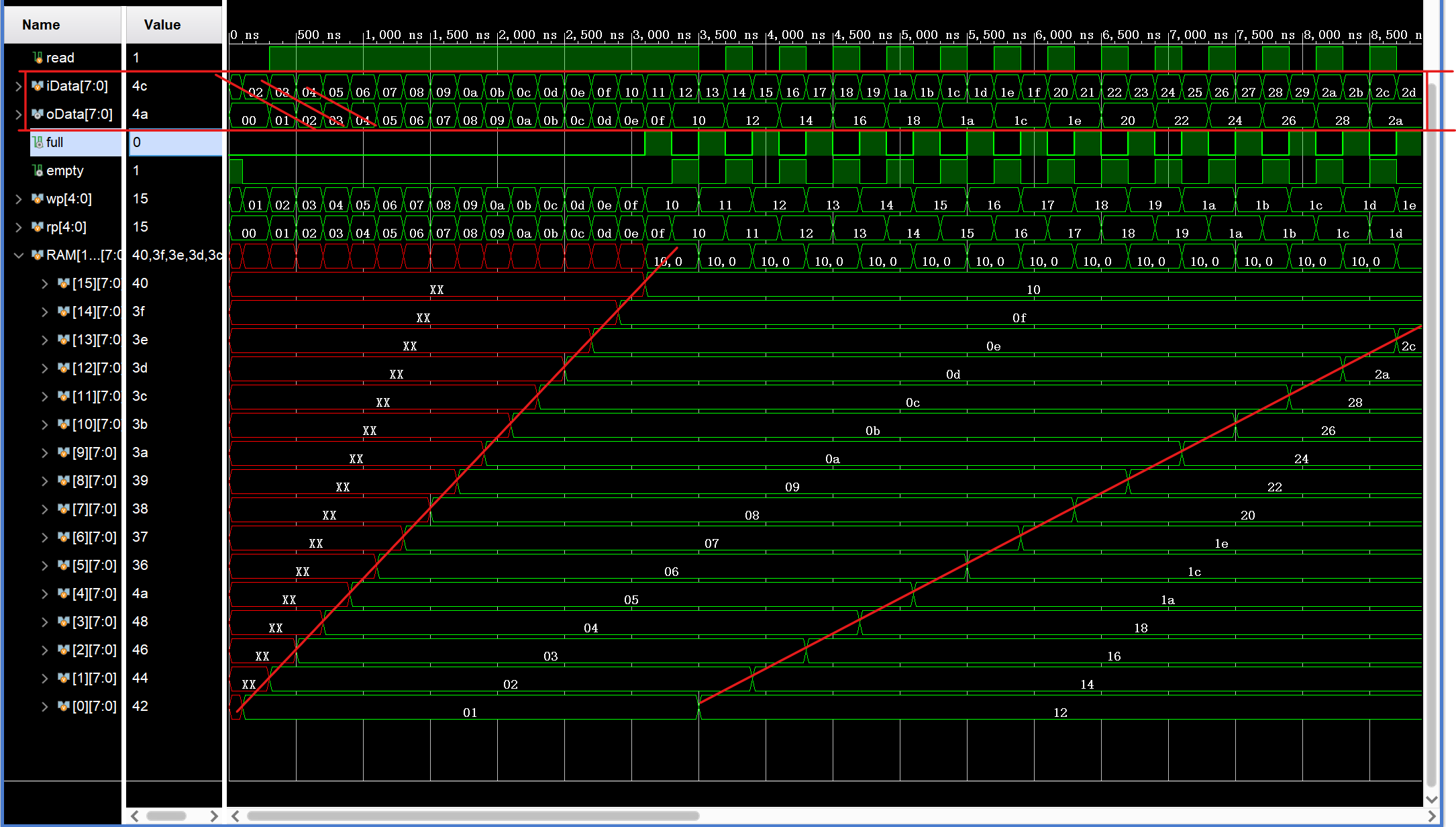
full = ( Wp [2] ^ Rp [2] & Wp [1:0] == Rp [1:0] );

所以有full和empty的计算方式：

full = ( Wp [2] ^ Rp [2] & Wp [1:0] == Rp [1:0] );

empty = ( Wp==Rp );



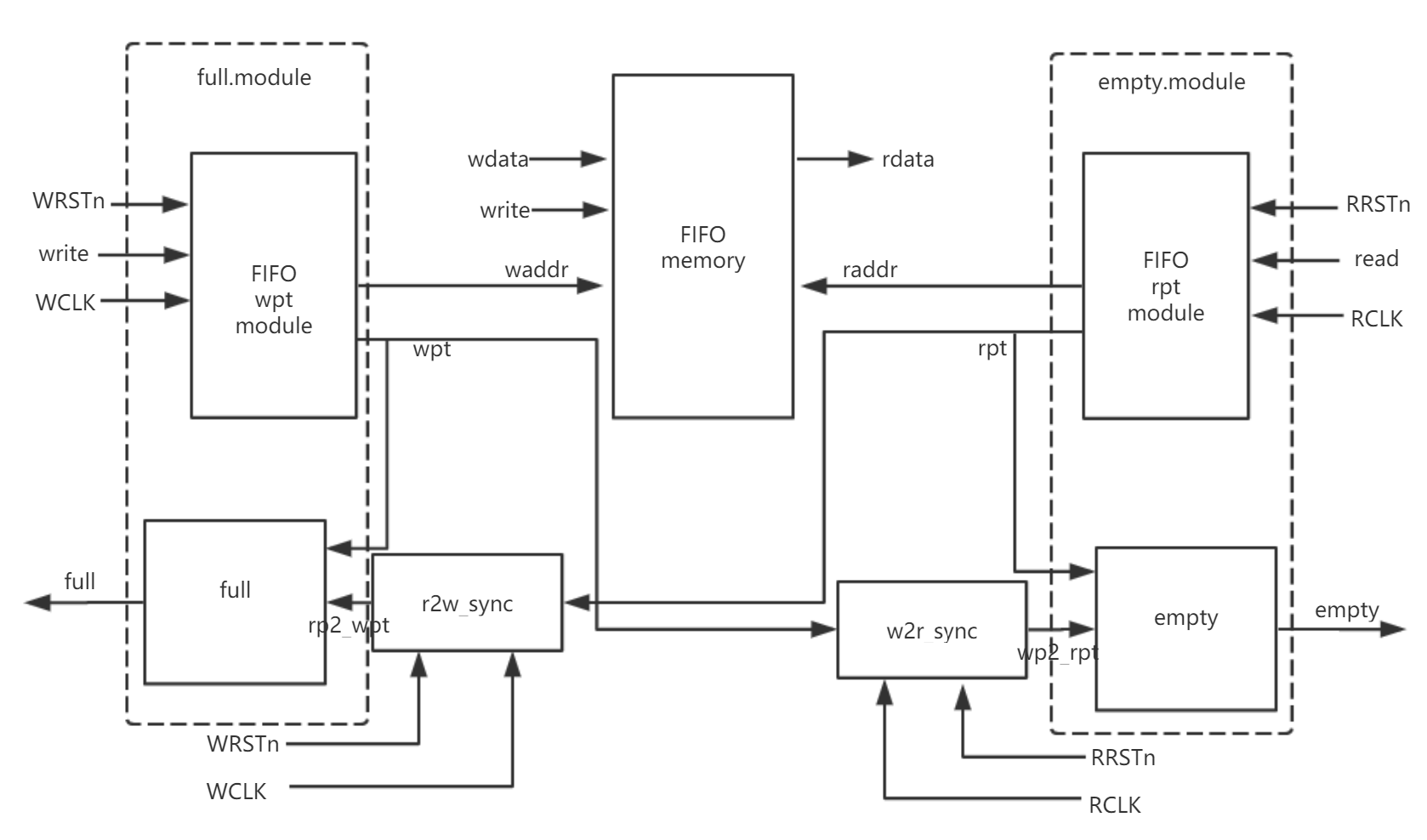


1. 异步FIFO

异步FIFO简单的说就是读写的时钟不同。可以在两个不同的时钟域之间传输数据，起到跨时钟域处理的作用，多用于处理跨时钟域的问题。

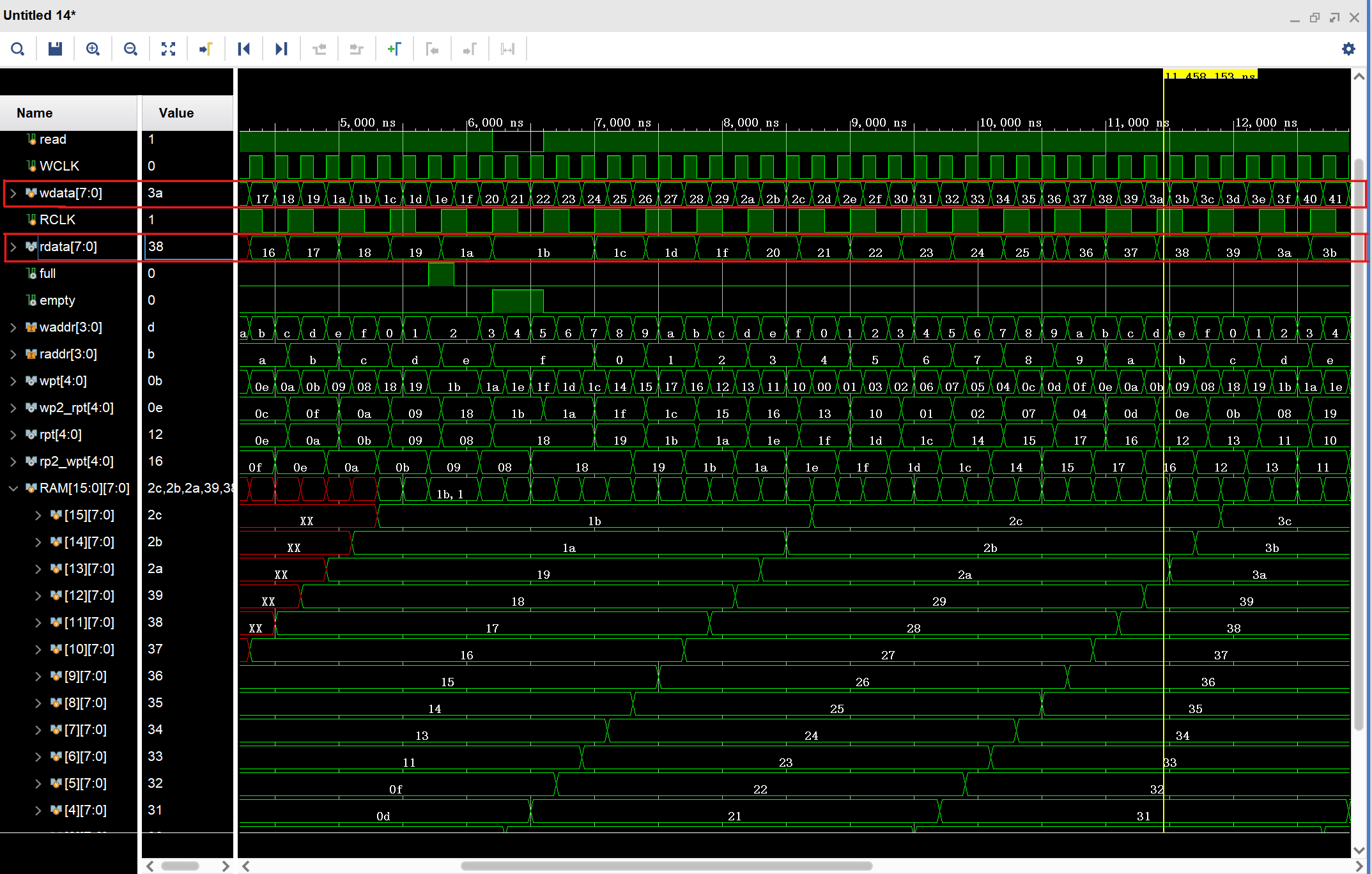
依旧是如何判断满和空的状态：

在上面的同步FIFO中已经说明判断满空的方法，可是那是在同时钟下；显然在不同的时钟域下需要将读、写指针进行同步才能进行判断



**conv\_out卷积结果**

**选出的窗口数据**



**读出的数据**

**写入的的数据**

Reference:

https://www.cnblogs.com/alinx/p/4223450.html

https://blog.csdn.net/MaoChuangAn/article/details/88783320

By zhangzek