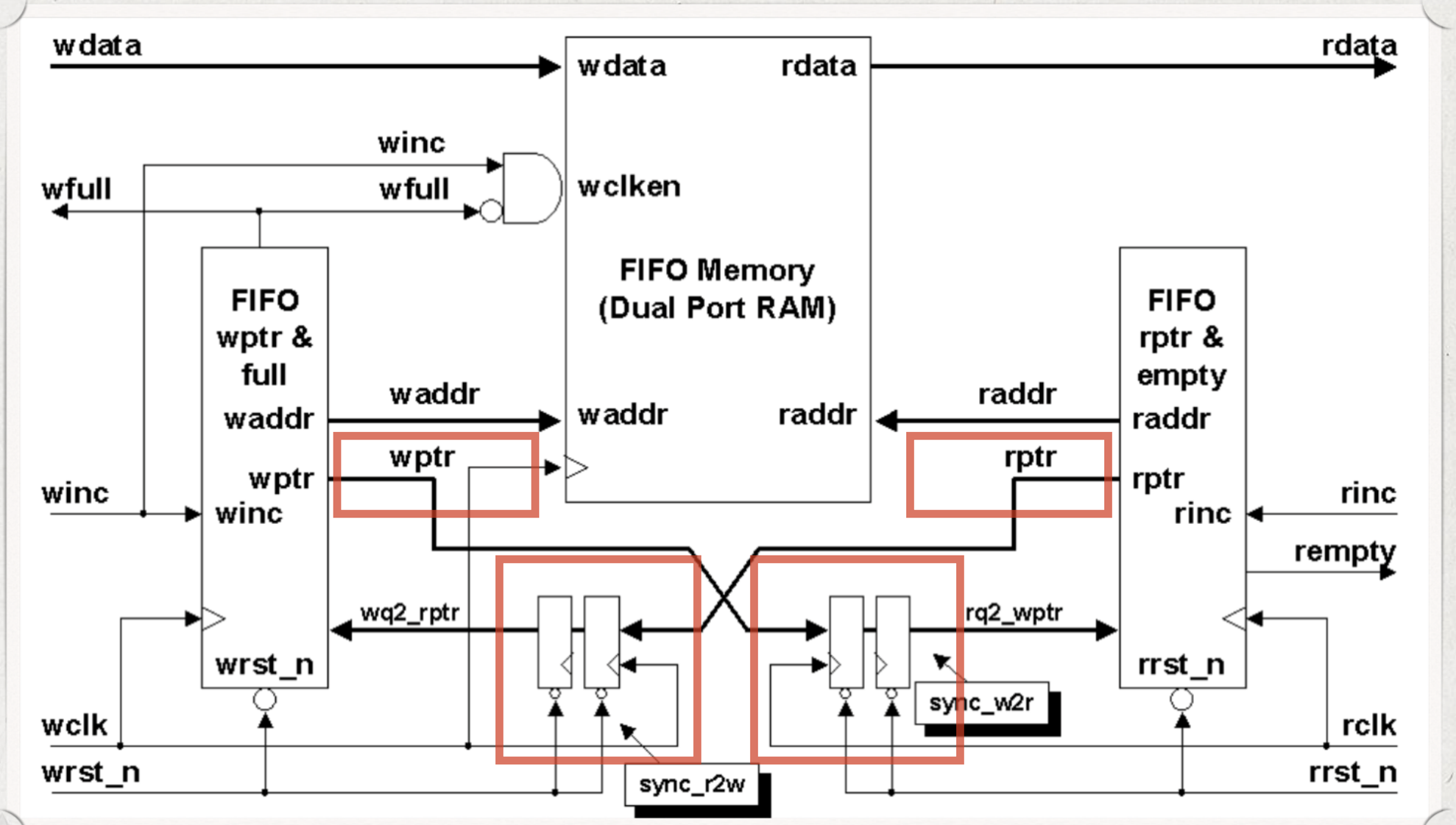
芯动力——硬件加速设计方法

第三章 同步电路与跨时钟域电路设计(4)

邸志雄@西南交通大学

zxdi@home.swjtu.edu.cn


wdata
wdata
rdata
rdata
winc
wfull
wfull
wclken
FIFO Memory
(Dual Port RAM)
FIFO
wptr &
full
waddr
waddr
waddr
raddr
raddr
FIFO
rptr &
empty
raddr
wptr
rptr
wptr
rptr
winc
winc
rinc
rinc
rempty
wq2_rptr
rq2_wptr
wrst_n
rrst_n
sync_w2r
wclk
rclk
wrst_n
rrst_n
sync_r2w

读写指针与格雷码

为什么需要格雷码做读写指针?

- 由于异步FIFO是工作在两个不同的时钟域中

读地址

写时钟

在某一时刻从0111->1000转变

采样读地址

得到的值有可能是0000~1111中的任一个值

- 这个不确定的读地址值会导致空满状态判断错误。

格雷码的特点：

- 格雷码相邻的2个数值之间只会有一位发生变化，其余各位都相同。
- 格雷码是一种循环码，0和最大数(2的n次方减1)之间也只有一位不同。

数值	二进制码	格雷码
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

引用格雷码之后，相邻数值只有1位发生翻转，1位翻转所引起的亚稳态的概率远远要小于几位同时翻转所引起的概率；因此，格雷码能很好的亚稳态出现的概率。

- 假设FIFO深度为8，则读写指针可采用格雷码进行编码；

数值	格雷码
0	000
1	001
2	011
3	010
4	110
5	111
6	101
7	100

数值	格雷码
0	000
1	001
2	011
3	010
4	110
5	111
6	101
7	100

000

110

100

- 假设FIFO深度为6，如果读写指针继续采用格雷码，那么当前首尾指针的所有比特位都不相同。此时，如果从尾部返回首部，则无法实现消除亚稳态的目的。

首地址的指针 000　　尾地址的指针 111

如何解决?

解决方法:

- 可将地址为5的指针设定为100，此时其与首地址的指针“000”相差一个bit位，与地址为4的指针“110”也相差一个bit位，满足消除亚稳态的要求。

最前面的3个问题:

- 如何在2次幂和非2次幂深度下设计相应的读写指针?
- 什么情况下，需要用格雷码来进行读写指针编码?
- 什么时候不可以使用格雷码?

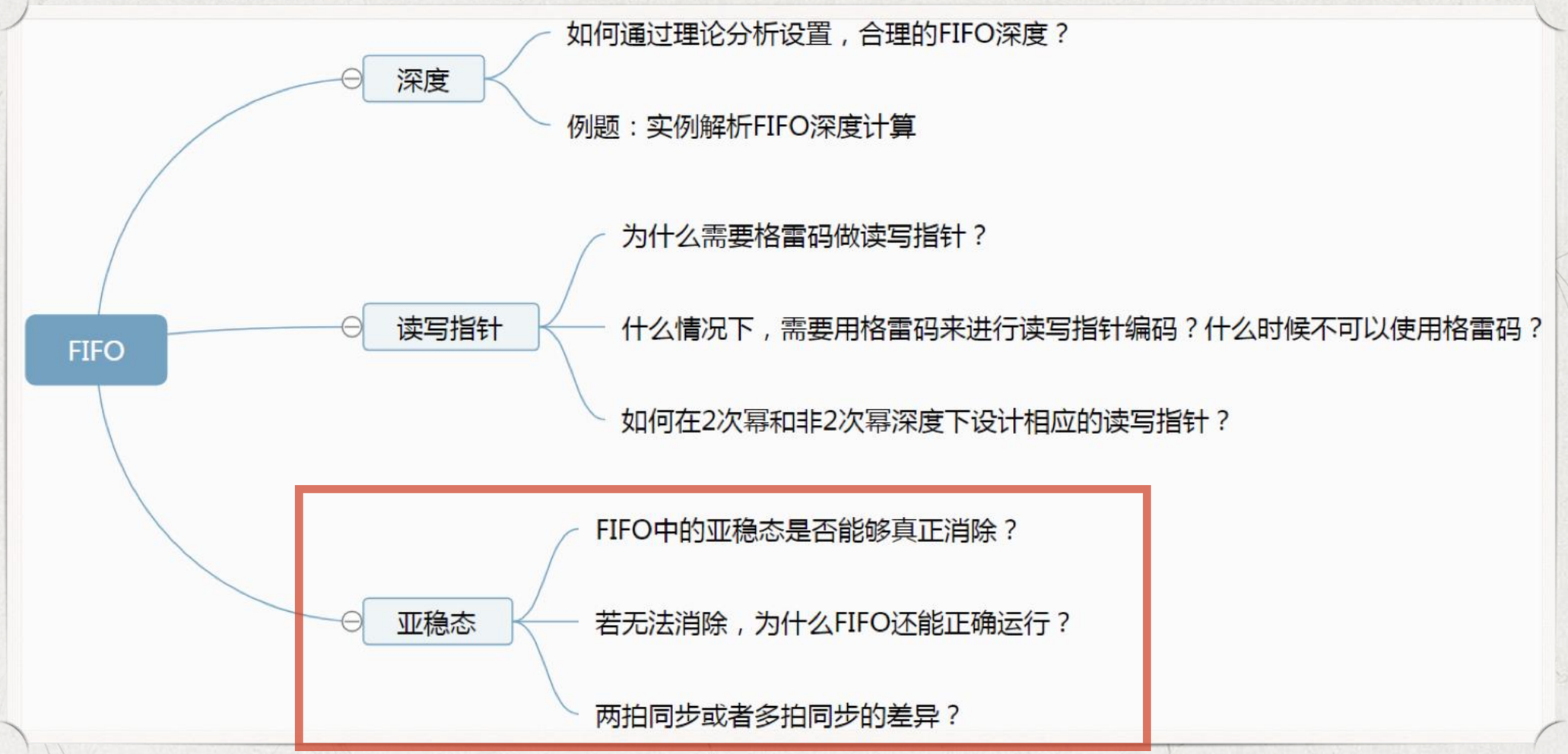
- 并不是一定要用格雷码做读写指针，而是当深度为2次幂的时候，刚好格雷码满足消除亚稳态的需求；

- 在非2次幂深度情况下，格雷码已经不再适用，此时的解决方法通常有：

① 若深度为偶数，可采用最接近的2次幂的格雷码编码，在此基础上修改；

② 深度为一般数值时，可自行设计一种逻辑电路，或者查找表，以实现指针每次只跳变一次的功能；

③ 以上方法通常在设计层面较为复杂，若无特定需求，可将FIFO深度设置为2次幂，浪费一些存储空间，来化简控制电路的复杂度。


FIFO
深度
如何通过理论分析设置，合理的FIFO深度？
例题：实例解析FIFO深度计算
读写指针
为什么需要格雷码做读写指针？
什么情况下，需要用格雷码来进行读写指针编码？什么时候不可以使用格雷码？
如何在2次幂和非2次幂深度下设计相应的读写指针？
亚稳态
FIFO中的亚稳态是否能够真正消除？
若无法消除，为什么FIFO还能正确运行？
两拍同步或者多拍同步的差异？

亚稳态相关

- 亚稳态不能从根本上消除，但可以通过采取一定的措施使其对电路造成的影响降低。

$$MTBF = \frac{e^{t_{MET}/C_2}}{C_1 f_{CLK} f_{DATA}}$$

既然亚稳态无法消除，那为什么FIFO还能正常工作?

如果指针为格雷码，失效的后果?

- 格雷码一次只有一位数据发生变化，这样在进行地址同步的时候，只有两种情况：

地址同步正确

地址同步出错，但是只有1位出错；

写地址 000->001

读时钟 同步出错

写地址 000->000

也就是地址没有跳变，但是用这个错误的写地址去做空判断不会出错，最多是让空标志在FIFO不是真正空的时候产生，而不会出现空读的情形。

gray码

- 保证的是同步后的读写地址即使在出错的情形下依然能够保证FIFO功能的正确性，当然同步后的读写地址出错总是存在的。

如果指针为格雷码，失效的后果？

- 格雷码一次只有一位数据发生变化，这样在进行地址同步的时候，只有两种情况：

地址同步正确　　地址同步出错，但是只有1位出错；

gray码

- 保证的是同步后的读写地址即使在出错的情形下依然能够保证FIFO功能的正确性，当然同步后的读写地址出错总是存在的。

需要注意gray码

- 只是在相邻两次跳变之间才会出现只有1位数据不一致的情形

- 超过两个周期则不一定，所以，地址总线bus skew一定不能超过一个周期，否则可能出现gray码多位数据跳变的情况，这个时候gray码就失去了作用，因为这时候同步后的地址已经不能保证只有1 位跳变了。

两拍同步或者多拍同步的差异?

将地址总线打两拍

- 为了避免亚稳态传播

不能消除亚稳态现象

- 因为时钟异步，亚稳态不可避免，但是可以极大降低亚稳态传播的概率，

低频情况

- STA不需要分析这里的异步时序，因为寄存器都可以在一拍内将亚稳态消除，恢复到正常0/1态。

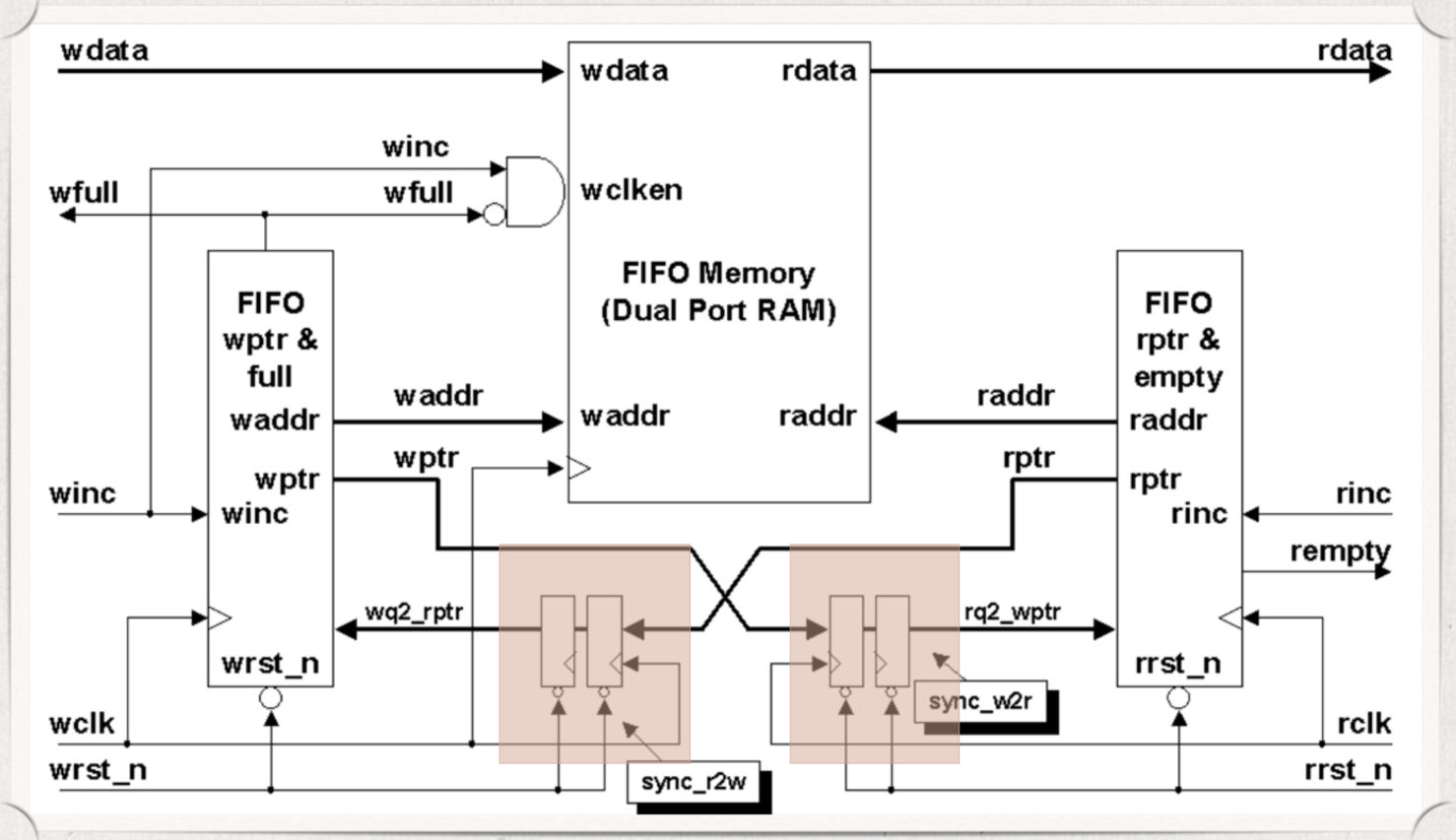
高频情况

- 不一定，尤其在 28nm工艺以下，需要检查两级触发器的延迟，保证延迟低，提高系统MTBF。

多拍

- 能够将亚稳态出现的概率进一步降低。

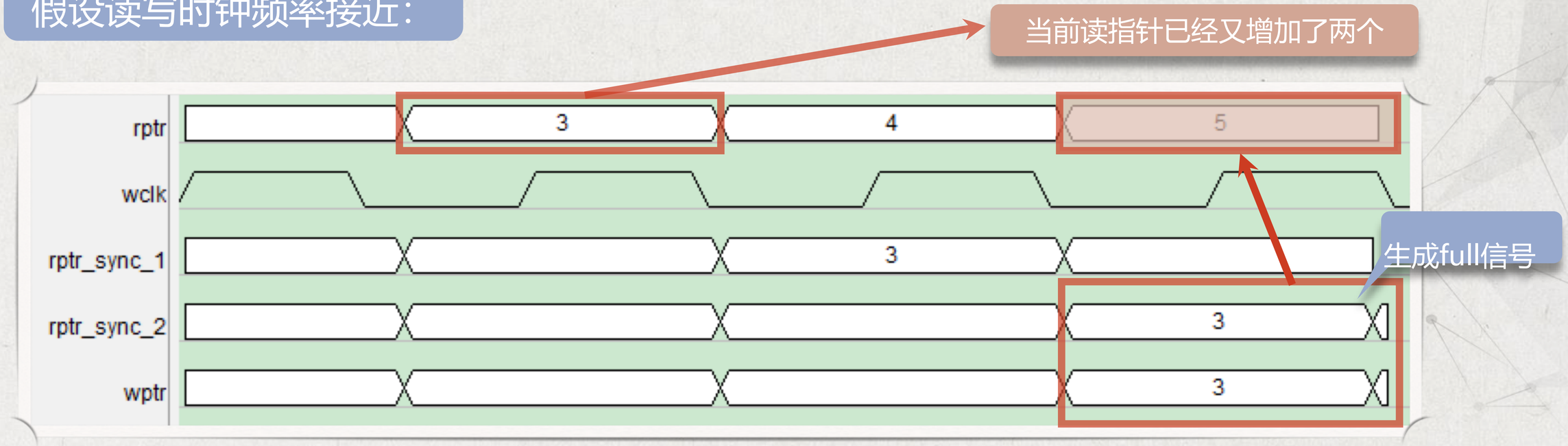
空满标志的判断方法是否有漏洞?



- rptr同步两个“wclk”后，在wclk时钟域与wprt进行比较，生成full信号
- wptr同步两个“rclk”后，在rclk时钟域与rprt进行比较，生成empty信号

空满标志的判断方法是否有漏洞?

假设读写时钟频率接近:

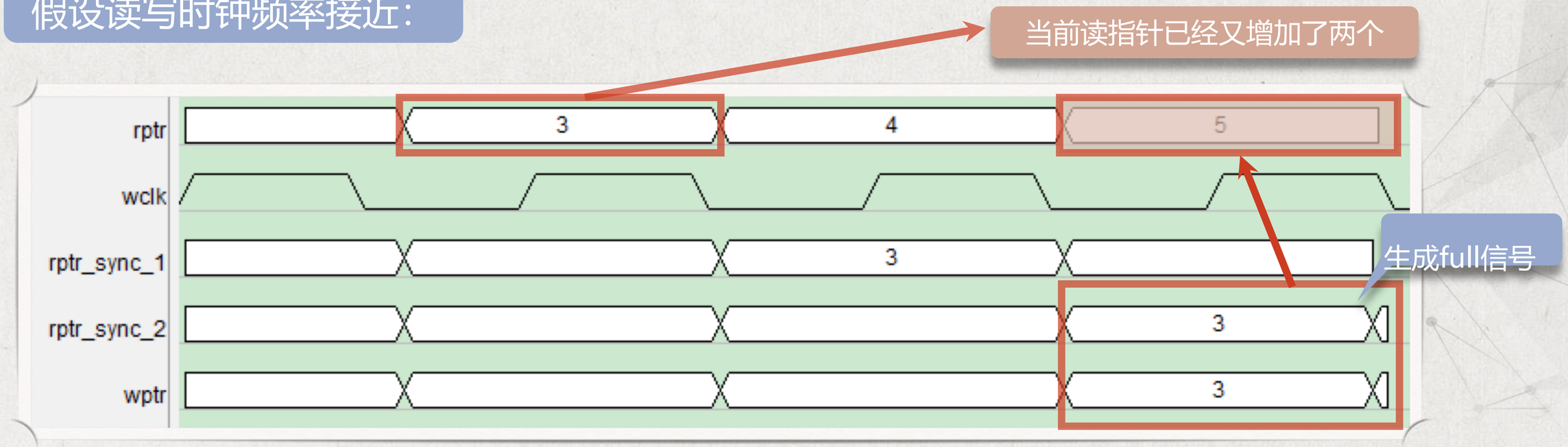


结论:

- 对于full信号的生成机制，同步后的读地址一定是小于或者等于当前的读地址，所以此时判断FIFO为满不一定是真满，这样更保守；
- Empty信号的机制同样成立，“空”时，不一定是真“空”。

空满标志的判断方法是否有漏洞?

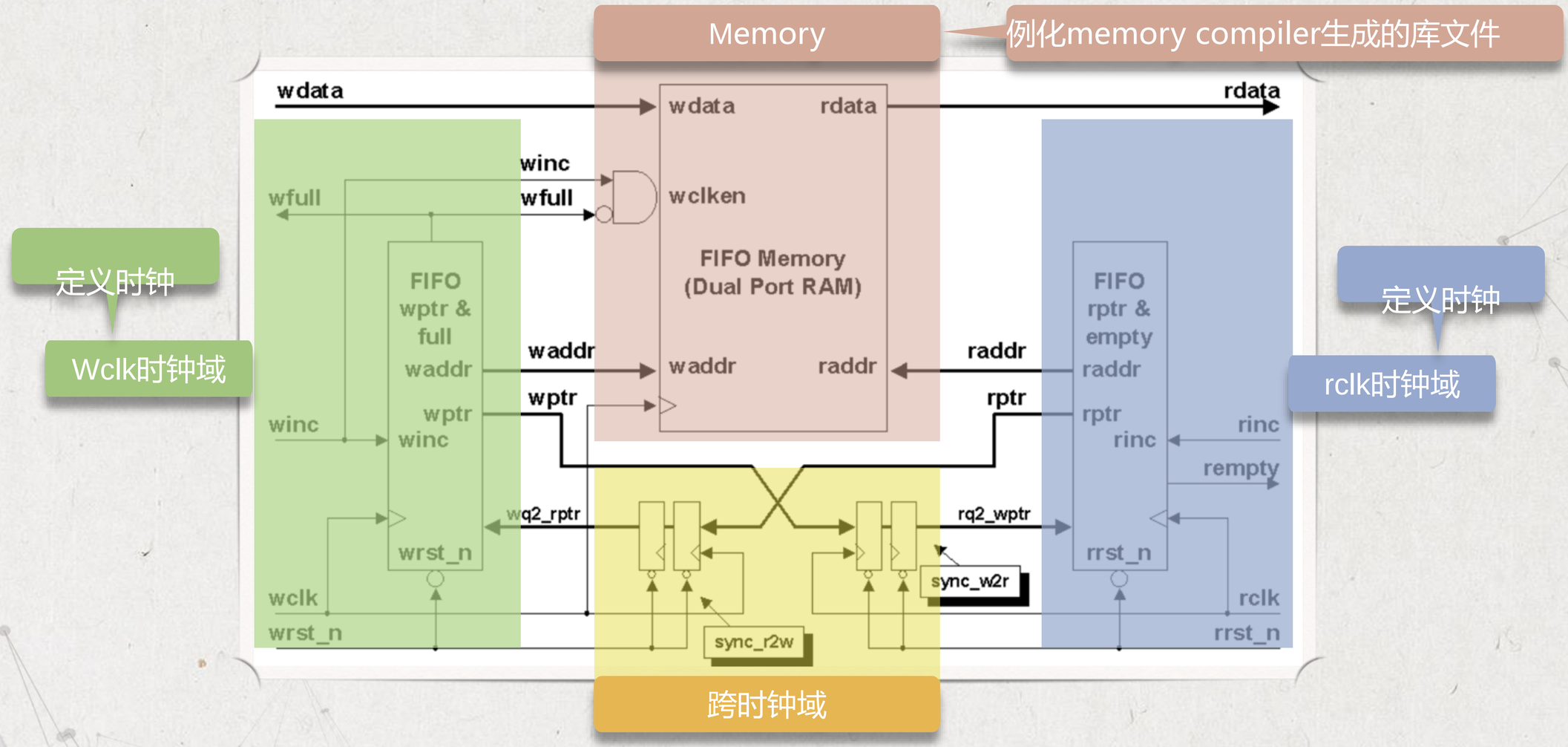
假设读写时钟频率接近:



总结:

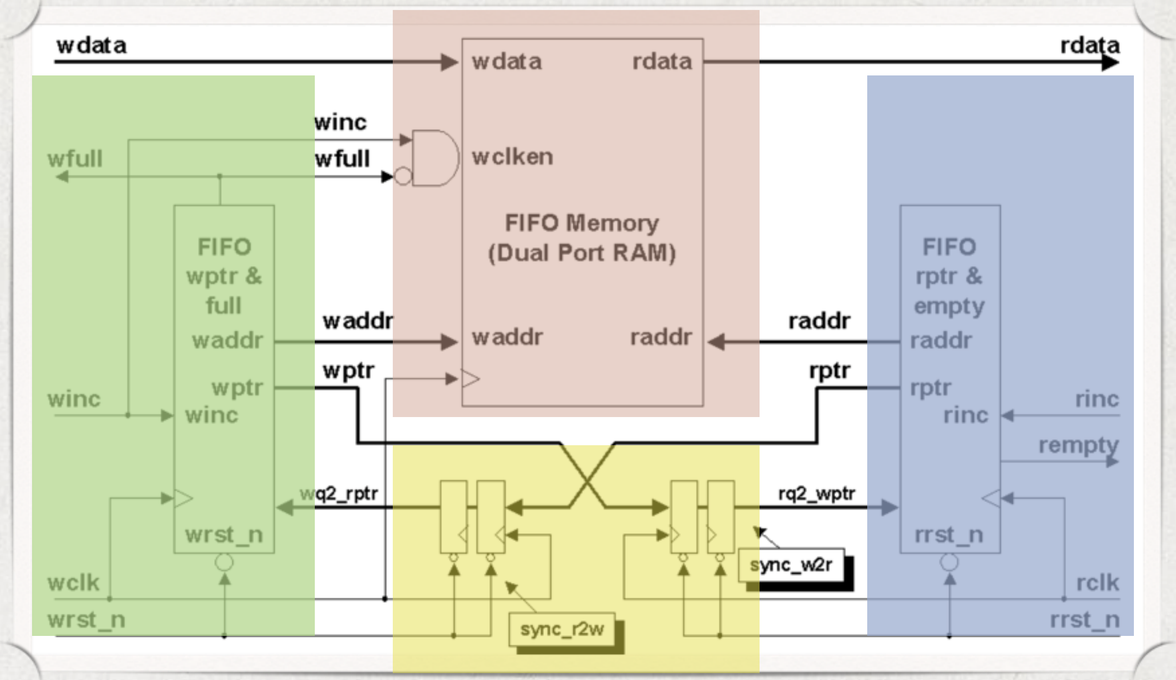
- 异步FIFO通过比较读写地址进行满空判断，但是读写地址属于不同的时钟域，所以在比较之前需要先将读写地址进行同步处理，此机制保证了FIFO在空满极限情况下，依然留有余量，存在一定的冗余空间。

如何对 FIFO 进行逻辑综合和静态时序分析?



如何对 FIFO 进行逻辑综合和静态时序分析?

跨时钟域



低频情况
- 设置wclk和rclk之间的false path;

高频情况
- 尤其在 28nm工艺以下，需要检查两级触发器的延迟，保证延迟低;

P&R时
- 要注意将两级同步器放置在一起，不要被工具自动分开;