基于 FPGA 的跨时钟域信号处理同步设计的重要

上次提出了一个处于异步时钟域的 MCU 与 FPGA 直接通信的实现方式，其实在这之前，特权同学想列举一个异步时钟域中出现的很典型的问题。也就是要用一个反例来说明没有足够重视异步通信会给整个设计带来什么样的危害。

要举的这个反例是真真切切的在某个项目上发生过的，很具有代表性。它不仅会涉及使用组合逻辑和时序逻辑在异步通信中的优劣、而且能把亚稳态的危害活生生的展现在你面前。

从这个模块要实现的功能说起吧，如图 1 所示，实现的功能其实很简单的，就是一个频率计，只不过 FPGA 除了脉冲采集进行计数外，还要响应 CPU 的控制。

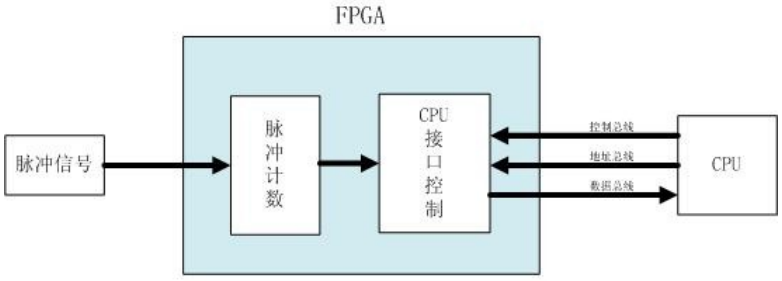

图 1 功能模块

CPU 的控制总线是指一个片选信号和一个读选通信号，当二者都有效时，FPGA 需要对 CPU 的地址总线进行译码，然后把采样脉冲值送到 CPU 的数据总线上。

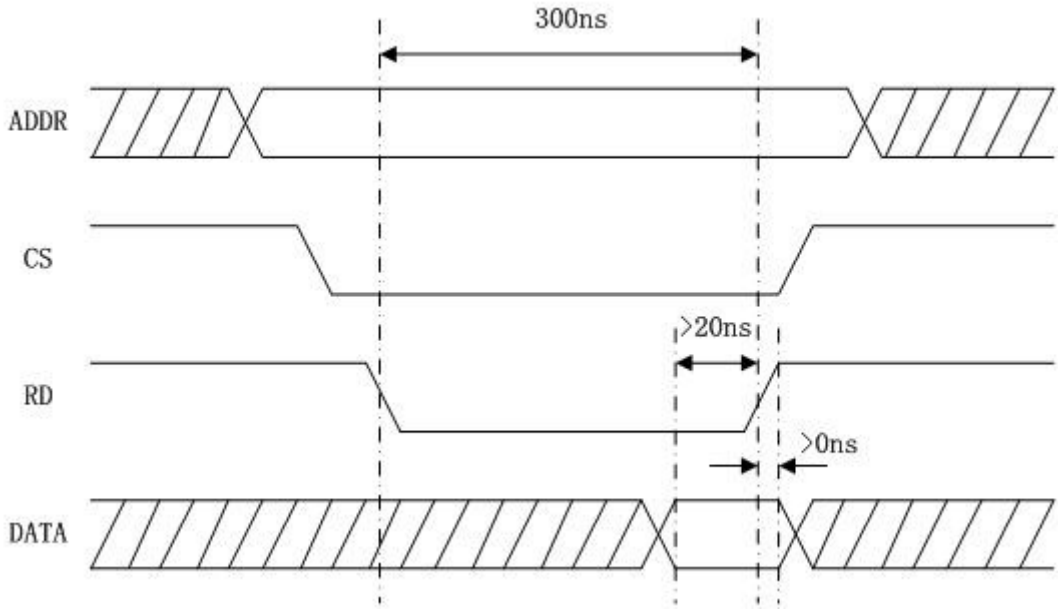

图 2 CPU 读时序

对于这样“简单”的功能，不少人可能会给出类似下面的以组合逻辑为主的实现方式：

```
input clk;

input rst_n;

input pulse;

input cs_n;

input rd_n;

input[3: 0] addr_bus;

output reg[15: 0] data_bus;

reg[15: 0] counter;

always @(posedge pulse or negedge rst_n)

if(! rst_n) counter <= 16 ‘d0;

else if(pulse) counter <= counter+1’ b1;

wire dsp_cs = cs_n & rd_n;
```

```
always @(dsp_cs or addr_bus)

if(dsp_cs) data_bus <= 16 'hzzzz;

else begin

case(addr_bus)

4' h0:  data_bus <= counter;

4 'h1:  ……;

……

default:  ;

endcase

end
```

咋一看，可能你会觉得这个代码也没什么问题，功能似乎都实现了。而且你会觉得这个代码简洁，也不需要耗费多少逻辑就能实现。但是，对于这种时钟满天飞的设计，存在着诸多亚稳态危害爆发的可能。脉冲信号和由 CPU 控制总线产生的选通信号是来自两个异步时钟域的信号。它们作为内部的时钟信号时，一个写寄存器 counter，一个读寄存器 counter。那么，很明显的，存在着发生冲突的可能。换句话说，如果寄存器正处于改变状态(被写)时被读取了，问题就随着而来，如图 3 所示。

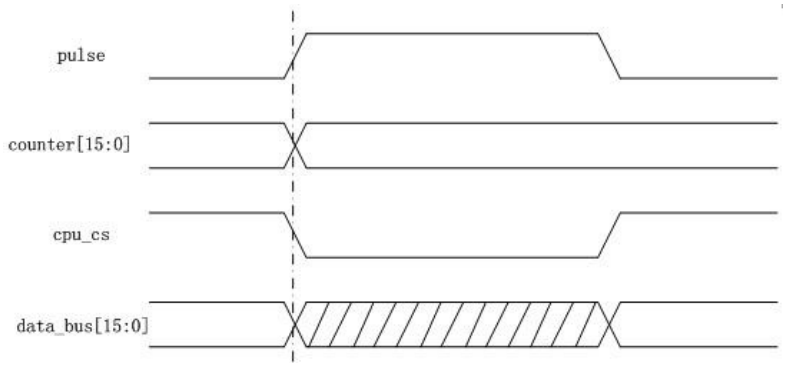

图 3 数据冲突

脉冲信号 pulse 和 CPU 读选通信号 cpu_cs 是异步信号，pulse 什么时候出现上升沿和 cpu_cs 什么时候出现下降沿是不可控的。所以，如果它们很不幸的一起触发了，那么，结果可想而知。计数器 counter[15：0]正在加一，这个自增的过程还在进行中，CPU 数据总线 data_bus[15：0]来读取

counter[15：0]，那么到底读取的值是自增之前的值还是自增之后的值呢？或者是其它的值呢？

所示，它是一个计数器的近似模型。当计数器自增一的时候，如果最低位为 0，那么自增的结果只会使最低位翻转；当最低位为 1，那么自增一的后果除了使最低位翻转，还有可能使其它任何位翻转，比如 4’b1111 自增一的后果会使 4 个位都翻转。由于每个位之间从发生翻转到翻转完成都需要经过一段逻辑延时和走线延时，对于一个 16 位的计数器，要想使这 16 位寄存器的翻转时间一致，那是不可能做到的。所以，对于之前的设计中出现了如图 3 的冲突时，被读取的脉冲值很可能是完全错误的。

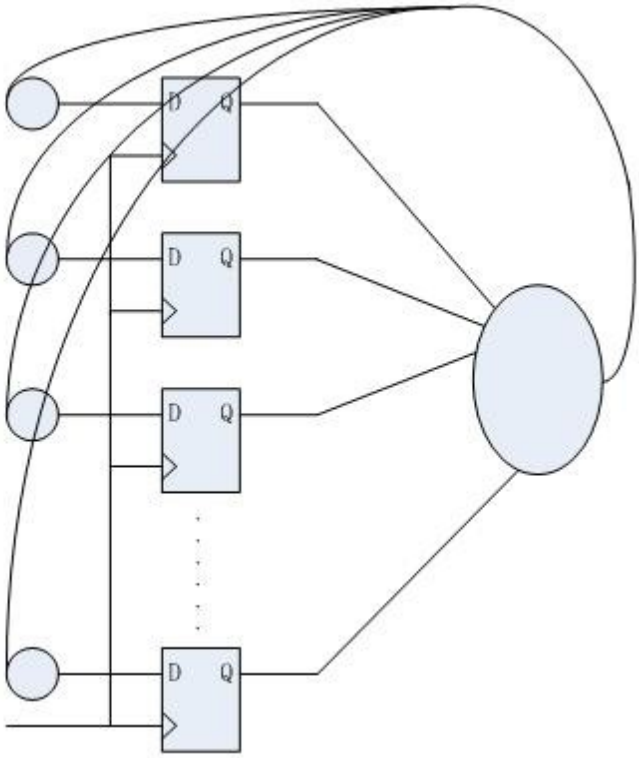

图 4 计数器模型

上面的代码是最典型的组合逻辑实现方式，是很不可行的。也许很多朋友会提出异议，也许还会提出很多类似的组合逻辑方案。但是，如果没有同步设计的思想，不把这两个异步时钟域的信号同步到一个时钟域里进行处理，冲突的问题在无法得到有效解决的。

那么，这个设计该如果同步呢？实现的方案其实上一次提到 FPGA 与 MCU 通信的博文里已经给出了答案。它的设计思想可以如图 5 所示。图 5 先是使用脉冲检测法把脉冲信号与系统时钟信号 clk 同步，然后依然使用脉冲检测法得到一个系统时钟宽度的使能脉冲作为数据锁存信号，也将 CPU 的控

制信号和系统时钟信号 clk 同步了。如此处理后，两个异步时钟域的信号就不存在任何读写冲突的情况了。

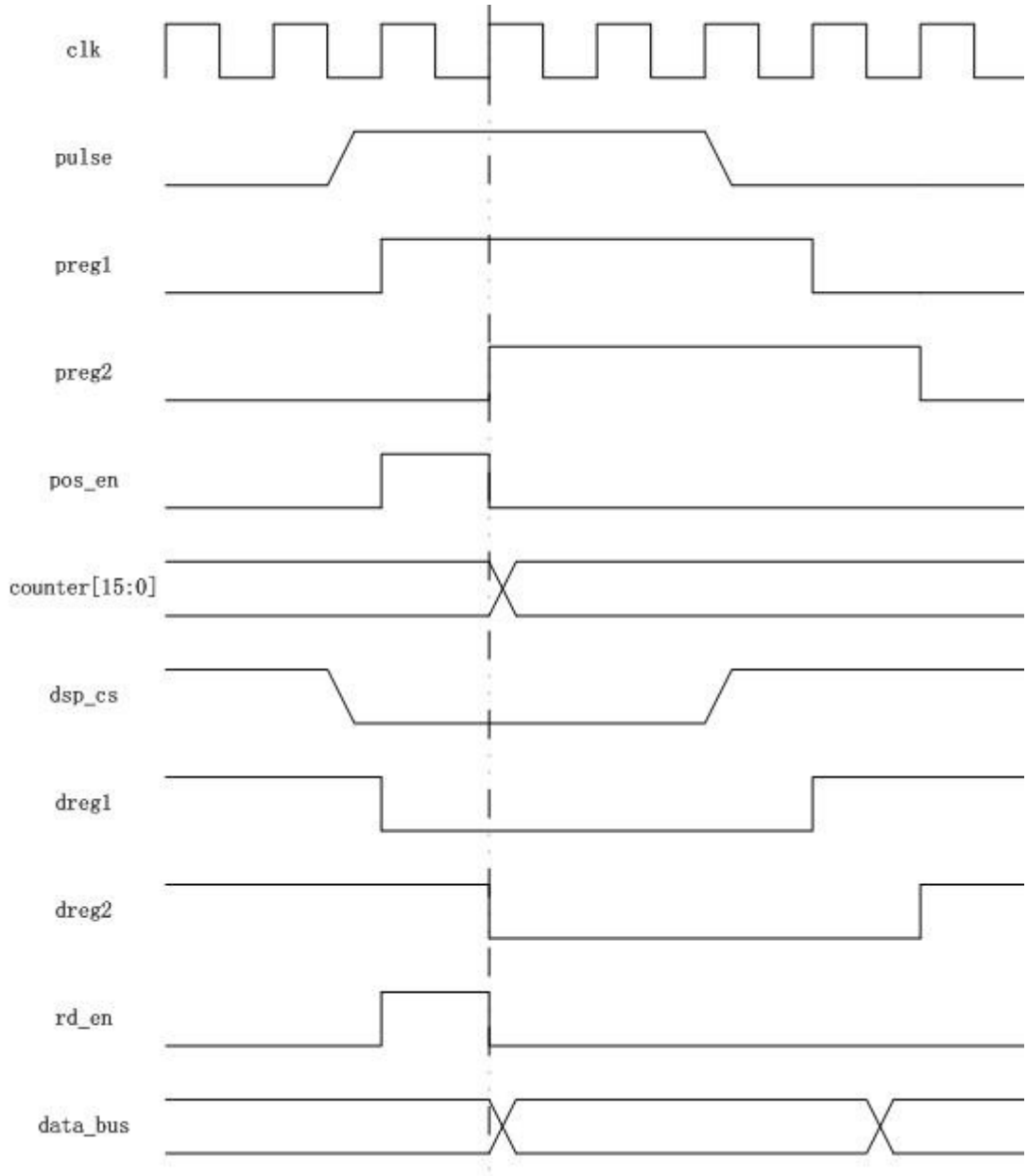

图 5 同步处理

这里提出来的解决方案就是使用了脉冲检测法进行同步，还有一些其它的同步方式，譬如专用握手信号同步、异步 FIFO 等等。