**异步流水线的设计与实现**

杨智杰 18020146

一、实验原理

用异步的方式实现一个4级流水线。使用的异步方式是两段握手协议和数据打包的方式。

二、实验步骤

1. 实现muller C门（图1所示）

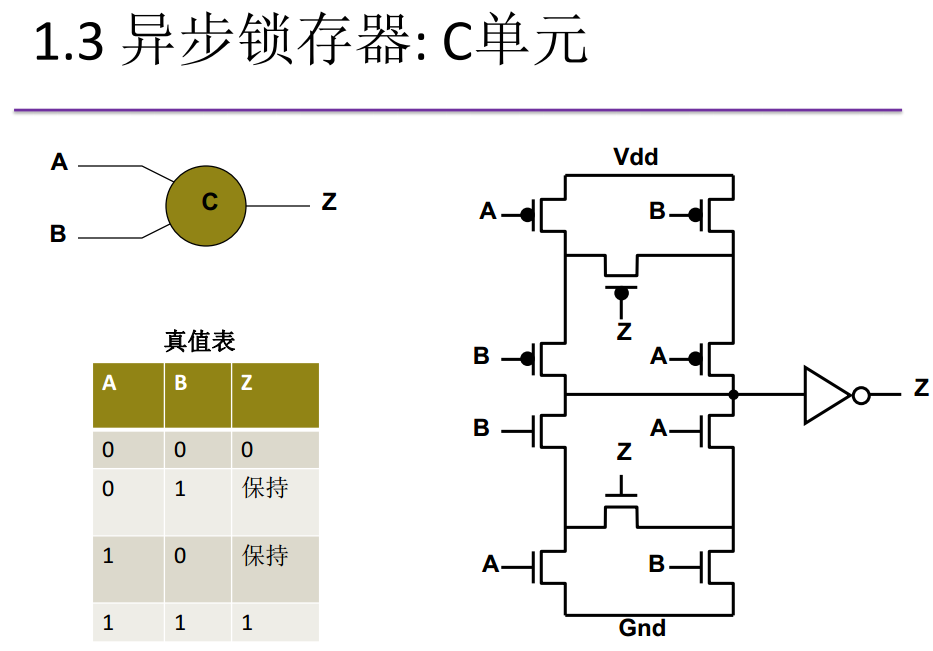


图1

1. 实现capture-pass 锁存器（图2所示）

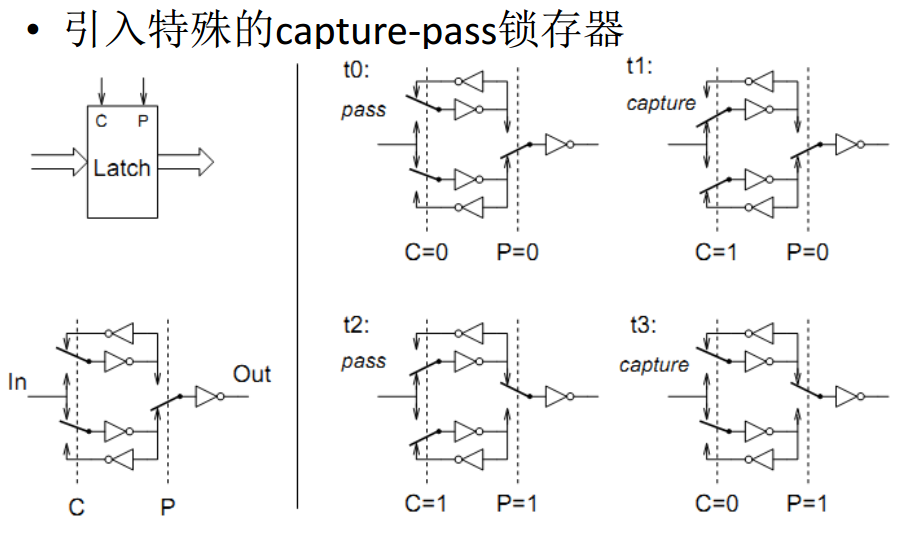


图2

1. 实现流水线的一个stage（图3所示）

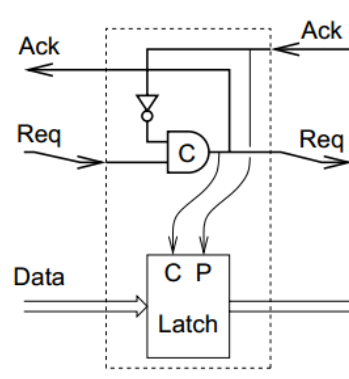


图3

1. 组合实现4级流水线（图4所示）

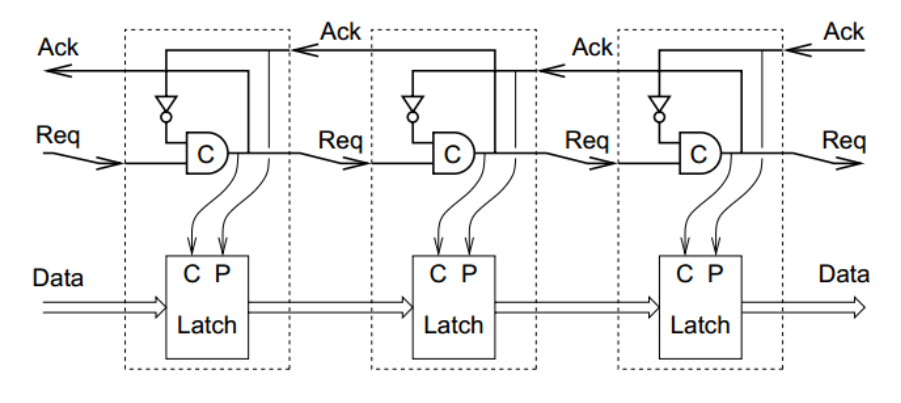
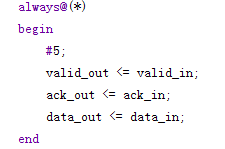
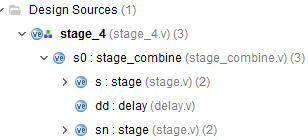


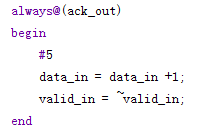
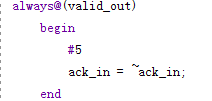
图4

三、实验结果

1.顺序流出1,2,3,4,5。（图5所示）其中每段流水线级中加增了一个延迟模块，对级与级之间的信号加增了5ns的延迟，使得异步的现象显现更加明显。



在testbench当中加入了判断，当接收到valid\_out（req\_out）信号时，才将ack\_in信号输入，当接收到ack\_out信号时，才更新输入的数据以及valid\_in(req\_in)信号以开始下一次的传输。



以下是波形结果。（如图5）。

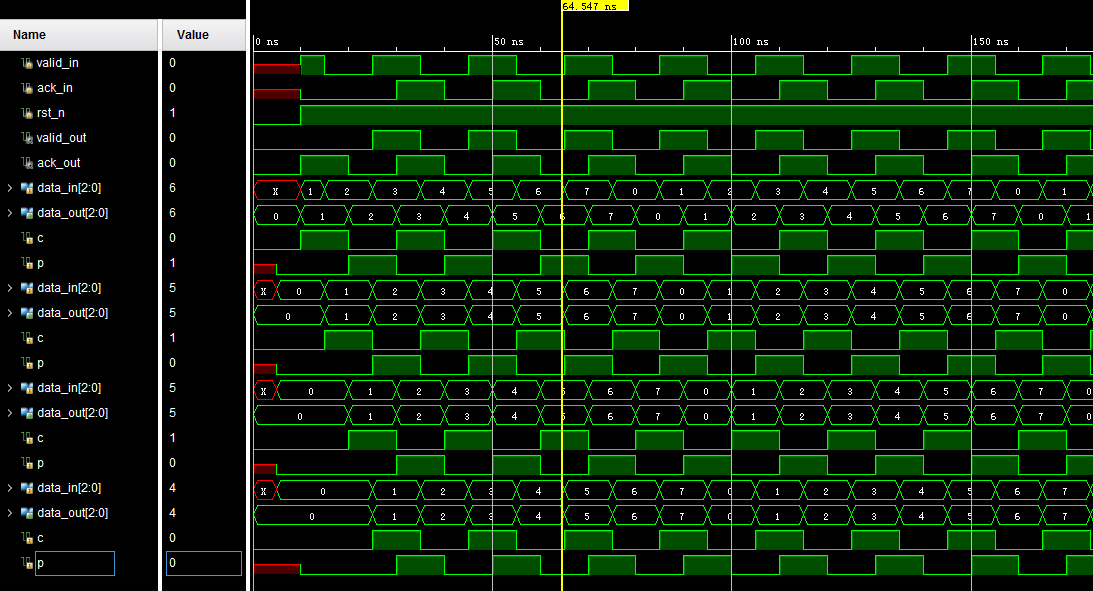


图5

2.不给ack信号，观察流水线各阶段状态。（图6所示）从上至下分别是第1-4级流水线的输入和输出。可以看到尽管我们不断的对该流水线进行输入（第一行data\_in）但是其输出已经被锁定为4，第二级锁定为3，第三级锁定为2，第四级锁定为1，最终只有1能够通过4级流水线而输出。也就是说，不断的输入，流水线充满的情况下各级中的数分别是4,3,2,1，最终只能输出1。

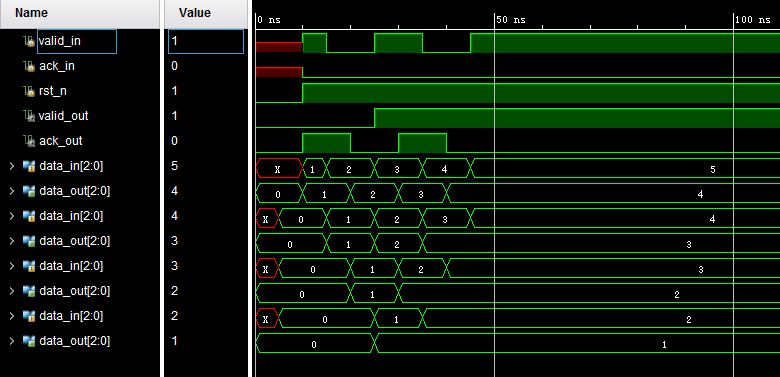


图6

3.综合版图（如图7）

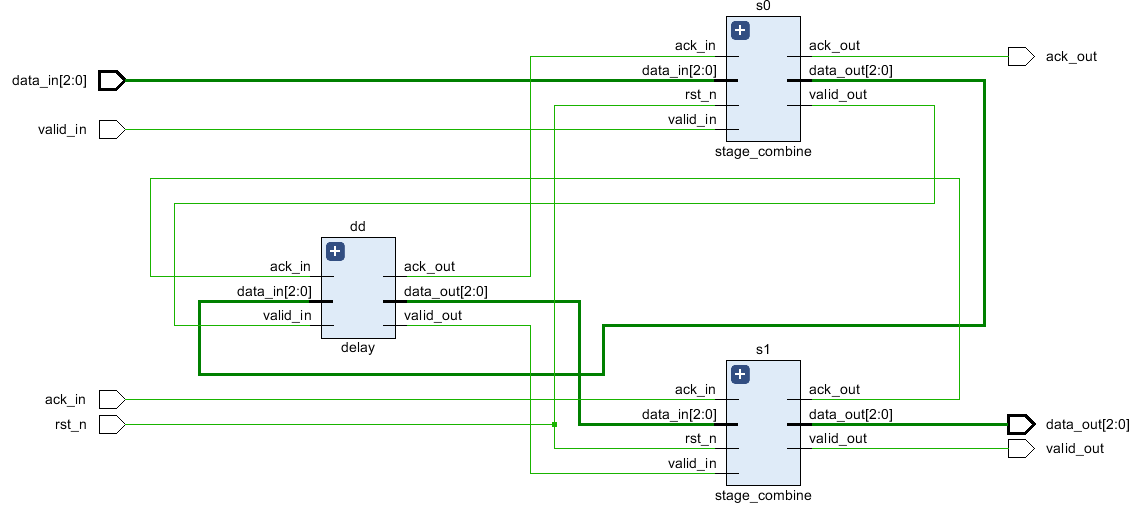


图7

四、实验分析

1.流水线的吞吐率（图8所示）

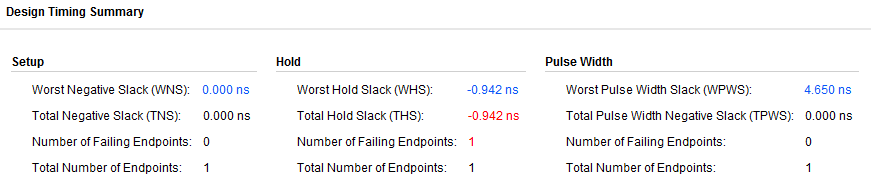


图8

使用vivado进行综合，使用的开发板是v7 690t，时钟选择的是输入信号valid，初始周期设置为10ns，观察得到该4级流水线的延迟是10.942ns，加上流水段之间的延迟15ns（4级流水线中间3个延迟，每个5ns）。因此吞吐率为38.55M results/s。

2.流水线的延迟

上图已经显示了，延迟是10.942+15=25.942ns。

3.流水线的利用率

利用率无法用波形来测算，因为没有时钟，因此整个流水线相当于行波流水一样的，尽管设置了中间的寄存器但是还是会在信号变化的“一拍”内完成所有的操作，类似于一个组合逻辑。

因此只能手工测算。手工测算得到的结果是，当流水线充满时，如果流水线级数是偶数2n，那么有n个stage处于pass状态，有n个stage处于capture状态，利用率为50%；当流水线充满时，如果流水线级数是奇数2n+1，那么有n+1个stage处于pass状态，有n个stage处于capture状态，利用率为n+1/2n+1,大于50%。

以上是理论计算，实际观察波形，在未对muller\_c门增加延迟的情况下，单单增加流水段之间的延迟，由于中间两段的c,p输入始终相反（即时刻处于capture状态，因为变化是瞬时完成所以无法观察到其处于pass状态的时候），所有观察到的利用率始终为25%。

在对muller\_c门增加延迟后，即考虑它存在延迟，此时可以观察到利用率为50%的情况，但是也不是时刻都是50%。例如下面两种情况：第一种利用率为50%但是流水线当中的情况不是我们设想的capture和pass是互相间隔的，而是第一段和最后一段处于pass中间两段处于capture。（如图9）

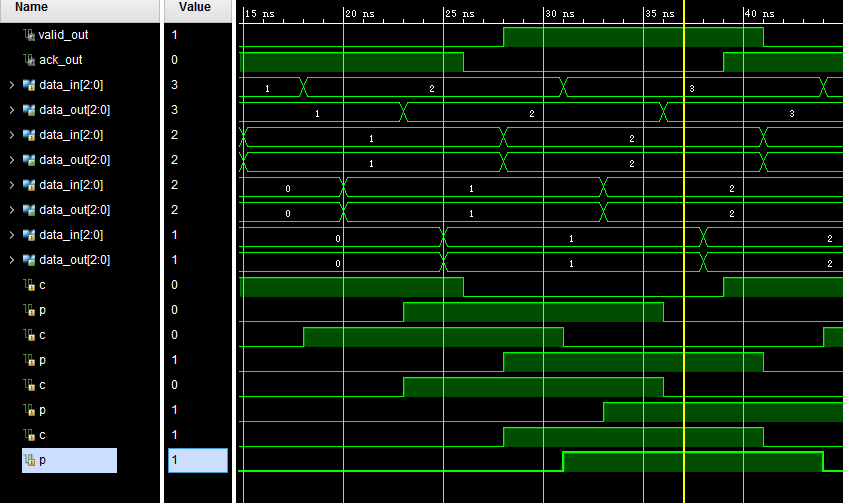


图9

第二种情况还是利用率为25%，因此貌似这个利用率是取决于流水线当中的各种延迟的，并且充满以后利用率也不是恒为50%。所以其平均利用率绝对要低于50%。（图10）

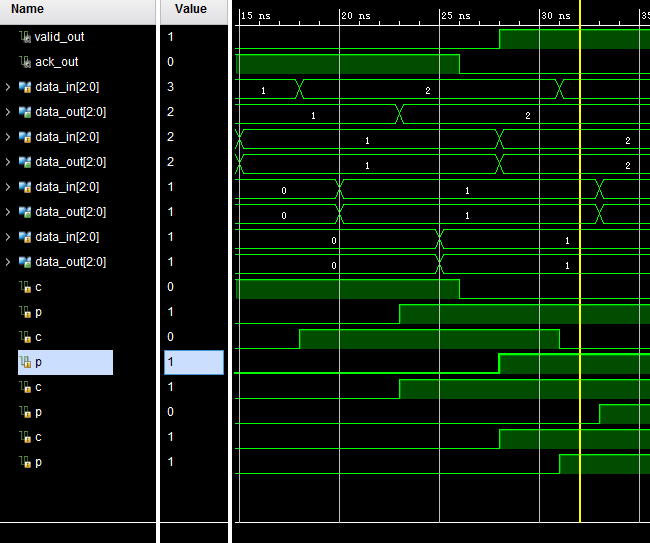


图10