**前言**:阻塞与非阻塞赋值是Verilog语言中最基本的部分，也是让大部分Verilog新手最困惑的地方。关于阻塞与非阻塞的著作文章可谓汗牛充栋，这些文章对阻塞与非阻塞赋值的原理进行了非常详细的讲解，但新手读了之后依然有种似懂非懂的感觉，编码过程中一如既往的犯错。所以，本文的目的立足于提供一种实用化的解决方案，用最简单的语言和形象的类比让新手能够一目了然的明白正确的编码方式以及相应的电路行为逻辑，关于仿真细节的讲解不是本文重点，需要了解更多细节的朋友可以参考文后列举的参考文献。

本文共分为三部分，第一部分是正确使用阻塞与非阻塞赋值的基本原则。第二部分是阻塞与非阻塞赋值对应电路的行为逻辑。第三部分是阻塞与非阻塞赋值的原理简介。

一．Golden Rule

编码原则很多，就阻塞非阻塞赋值而言，新手最需要牢记的是其中三条：

1)      时序逻辑一定用非阻塞赋值”<=”,一旦看到敏感列表有posedge就用”<=”。

2)      组合逻辑一定用”=”，一旦敏感列表没有posedge就用”=”，一旦看到assign就用”=”。

3)      时序逻辑和组合逻辑分成不同的模块，即一个always模块里面只能出现非阻塞赋值”<=”或者”=”。如果发现两种赋值并存，一个字”改”，心存侥幸可能会给后续工作带来更多麻烦。

以上三条，对新手而言不必追求为什么，需要的就是条件反射的照章办事。最后说一句，新手可能记不住哪个符号是阻塞赋值，哪个是非阻塞赋值，大家可以数数，”非阻塞赋值”一共5个字，“阻塞赋值“4个字，所以非阻塞用的符号”<=”比阻塞赋值用的符号”=”长。

二．电路行为逻辑。

第一节给出了三条最基本的编码原则，有个朋友可能会想，按照这三条编码原则写出来的代码会按怎样的逻辑工作呢？这一节就是回答这个问题。

首先解释一下阻塞赋值与非阻塞赋值的含义。所谓的阻塞赋值”=”就是说，在这个语句没有执行完之前，后面的语句是不执行的。这里执行的含义是指完成变量值的更新。非阻塞赋值”<=”是指，所有的语句可以并发执行，而前面的值是否执行完毕不会影响后面的语句，换句话说，语句的顺序是无关紧要的。

举个例子，假设一个模块，有2个寄存器，b和c，初值都是1。a为输入信号线。在某个时刻，因为某种原因，模块被触发执行。对于组合逻辑而言，一般是输入信号值变了，对于时序逻辑而言，一般是时钟沿到了。

首先看组合逻辑：假设输入a = 2;

always@(a)

begin

       b = a;

    c = b;

end

由于是阻塞赋值，所以首先执行完第一句b=a，执行完成之后b=2。接着执行c=b，执行完成后c=2，一次仿真结束后 b=c=2；

对于时序逻辑而言，依然假设a =2;

always@(posedge clk)

begin

       b <= a;

       c <= b;

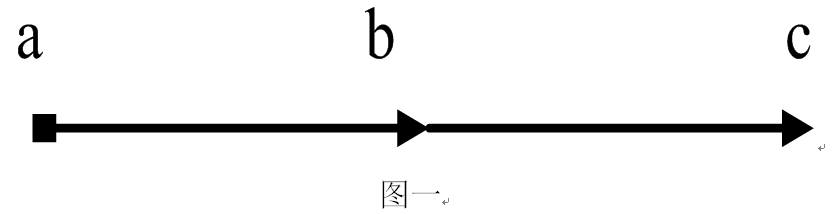
end

由于是非阻塞赋值，首先执行第一句b<=a，这时候a = 2,但是还没有执行完第一句的时候，第二句c<=b也执行了，由于第一句没有执行完，b的值还是1，这时候赋值给c的值也是1。执行完毕的结果就是c=1,b=2.等到模块再次被触发的时候c的值更新为2。有个朋友可能就会问了，凭啥第一句执行到一半就该第二句执行呢？到底是第一句先完成赋值呢还是第二句先完成赋值？答案是，谁先完成赋值都没关系，结果是一样的。

为什么说结果一样呢？因为两种赋值方式分别是按照下面的顺序执行的。阻塞赋值，就跟C语言一样，严格按照代码书写的先后顺序执行，所有值都是立即更新，并且在下面的语句中按照新值执行。而时序电路就不一样了，大家可以这么理解时序逻辑的代码行为，一次执行分为两轮：第一轮是所有的左值都先赋给临时变量，第二轮用输入值以及和右值同名的临时变量值去更新左值。比如上面的例子，第一轮，赋给临时变量：tempc=c;tempb=b。第二轮，临时变量更新左值，b = a;c = tempb;output = tempc。从上面的分析也可以看出，组合逻辑的结果与代码顺序直接相关，而时序逻辑与代码顺序没有关系。这就是所谓的顺序执行（组合逻辑）与并发执行（时序逻辑）。

为了进一步理解这两种赋值方式的行为，下面用对应的电路进行说明，以前面的代码为例。大家首先闭上眼睛想想，对应的电路是什么样子的呢？

其实答案很简单，对于阻塞赋值来说，如图一所示，综合的结果就是一根导线，当然，可能有反相器，buf什么的，反正还是可以看作一根线。



到这里，应该就很容易理解顺序执行的行为方式了。

有细心的朋友可能会问，如果换种写法呢？

always@(a)

begin

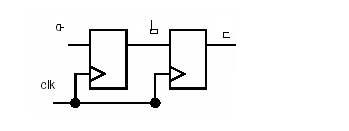
       c = b;

   b = a;

end

很显然，这种电路的行为跟之前是不一样的，从逻辑来看会产生类似于非阻塞赋值的结果，但很显然不满足非阻塞赋值并发执行的特点。如果把输入电平a触发改成时钟边沿触发posedge clk，出来的就是寄存器，但这违反了时序逻辑不用阻塞赋值的原则，所以严重不推荐。至于这种组合逻辑描述方式出来的电路是啥我也不知道，大家可以自己综合看看，或者哪位高人补上~~应该注意的是，如果想象不出这种怪异的coding方式会产生何种电路，就不要这么写，因为实现这种逻辑最好的办法是采用非阻塞方式描述。

而对于非阻塞赋值而言，如图二所示，综合出来的结果就是2个寄存器。对b,c赋值的过程就是寄存器输入采样的过程，很显然两个采样是同时进行的，而且一次时钟沿只会采样一次，所以输入值a会首先被采样到b，再在下一个时钟被采样到c。



                图二

总结一下，关于两种赋值方式，首先讲述了代码执行的过程，然后用直连线和寄存器分别对应了两种描述方式。应该指出的是，非阻塞赋值用寄存器的类比是完全准确的，而阻塞赋值用直连线的类比却未必准确，只不过因为一般认为直连线是从输入到输出依次更新的，而且没有传输以外的延迟，所以这种类比有助于新人理解，虽然不够严密。大家熟悉了之后就应该按照更严谨的方式去理解。