办公室的故事（Incremental Compilation)

我们总是在想，有没有那么一种可能，让我可以少花力气多赚钱，我们这里就开始讨论这样一个问题。

好比你现在有一个很大的公司，你有一个办公室，里面空空的，你可以有很多方法把你的员工放在里面。当然你可以一股脑儿的把所有人全部很随意的放在里面。但是导致的结果会是什么样的呢？你会发现两个工作关系很精密的人，每天需要从这头走到那头的交流。这样的工作效率不可能好的。所以我们会比较希望把他们整合在一起。至少可以让一个部门坐在一起。然后我们再在这个相对比较小的区域里面做调整来提高工作效率，这样会容易很多了。每个部门当然就是公司的一部分(design patition)，我们需要把这个部门的人都放在一个限定的空间里面，而给每个部门提供的空间，我们定义为 Logic Lock. 空间是一个物理的概念，而部门是一个人为理念造成的一个概念。我们并不是一定要把他们俩结合起来。但是，如果结合起来会有这样一个好处。

好比，我有五个部门 A,B,C,D,E。而不给他们限制空间，他们可能随意的去做。也许相对来说部门内的人士坐在一起了，但是这中间留下的空间变得很没规律。两种情况下会有问题，一个是，需要加一个新的部门，那么这个部门就需要拆开放，另外就是某些部门扩张了，新人就没地方放了。所以，宏观调控是非常重要的。所以我们需要把设计理念的分割和物理空间的分割紧密的结合在一起。

增量编译的思想和这个故事是完全一致的。我们需要把设计分成一些相对比较大的部分，然后给他们安排相应的位置去放置。这样有非常多的好处：

1. 关系相对精密的电路的位置比较接近，减少了连接线上的延迟。
2. 对一些已经表现很好的模块可以保留他们的编译结果。这样可以获得最大的好处。每一次编译都获得一个比较好的结果加以保留，这样整个设计就一点点按照增量的方式获得了最好的那个结果。
3. 由于一些模块已经保留了，所以工具不会在那些模块上面耗费时间，所以编译时间大大缩短。

我们来看看有这种思想可以实现的两种编译过程：

自上而下：
我是公司的总经理，我们去招了一堆人，然后把它们放在各个部门里面。根据每个部门人数的多少，给他们划分相应的资源。

自下而上：
我是公司的总经理，我手下有一些部门的负责人，我让这些人去招人。我给他们一些人数的预算，然后给他们提供空间。那些部门经理自己去招人来填充自己的一亩三分地。最后把这个办公室填满。

和其他所有工具一样，我们需要有一些行为准则来获得最好的效果：

1. 尽量把工作关系精密的人放在同一个部门里面。部门和部门之间是不能进行优化的。好比两个人需要每小时都交流工作，但是你把他们放在两个部门，即使是相互隔壁的部门，也不能保证他们正好墙对墙的做。工具只对部门内部进行调整优化，出了部门，他们就死人不管了。
2. 关门原则。在部门和部门之间都按上门。就是说对输出和输入的数据都用 register 打一下。这其实是和第一条是相关的。因为部门之间不进行任何优化，为怕避免麻烦，建议大家都按上门来提高效率
3. 部门和部门之间尽量少串门。尽可能的减少互动。这个其实是可以在设计的过程中进行调整挑选的。
4. 保证一个部门的人数，不是太少（大于 2,000LE）。这很好理解，就那么几个人，还优化什么呀。
5. 不要放不用的接口。没有结果的连接在编译中是会被优化掉的。但是在增量编译中，却会被保留。所以如果已经知道一个接口是没有用的，不如把它干掉算了。
6. 大家知道，FPGA 内部是没有双向信号的，所以一定避免双向信号。除非你这个型号是要直接连到芯片管脚上的。
7. 避免同一个信号重复输出。
8. 不要把直接从输入到输出的信号放到部门内部。这是一种资源浪费
9. 公司有一些资源，好比打印机啊，传真机啊之类的，在分配办公区域的时候，尽量考虑资源的合理使用。由于资源位置是固定的，所以当你把区域分配给部门的时候，其实也把资源分配给他们使用了。而如果他们用不上，也就浪费了。
10. 多给部门与部门之间的交流留一些时间预算。

这么看起来，如果你学会了增量编译，好像顺便把 HR 也个学了。