1. **标识符：**

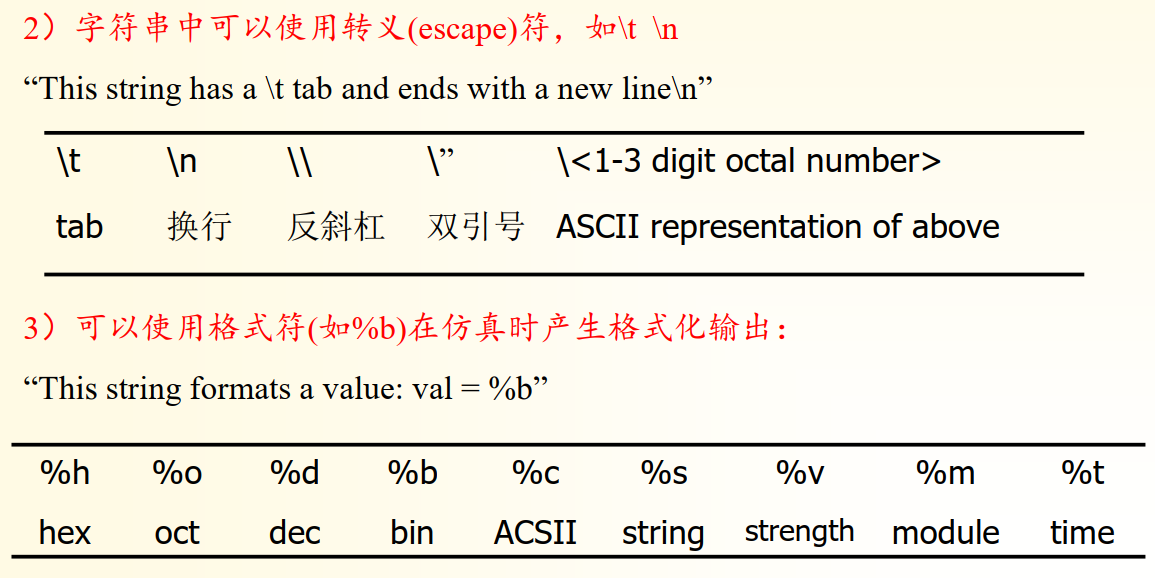
区分大小写，必须以字母（a-z，A-Z）或（\_）开头，后面可以是字母、数字、或\_。

1. **科学表示法：**

<尾数><e或E><指数>，表示：尾数X。

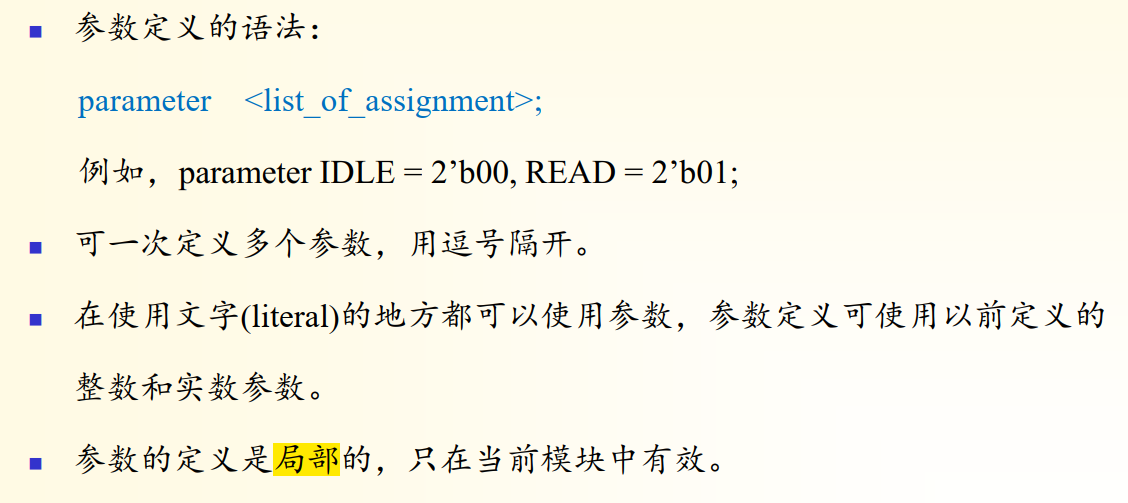
1. **字符串：**

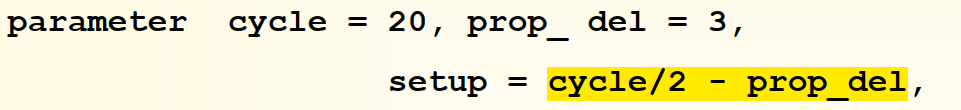
在一行中用双引号括起来，不能跨行。可以使用转义符。



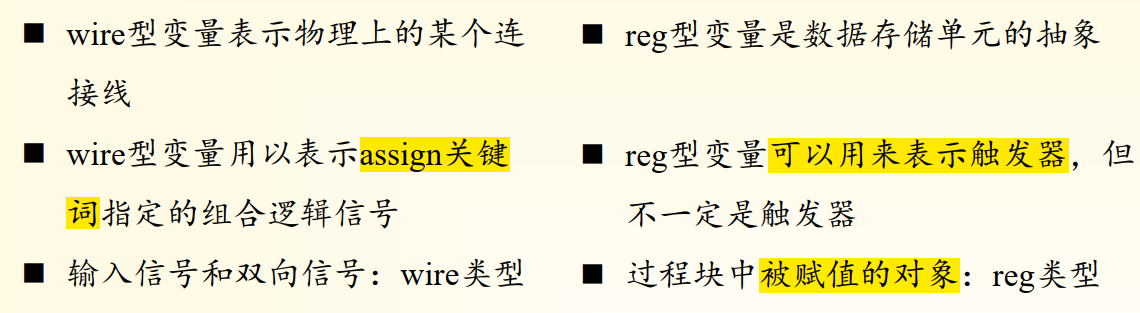
1. **Parameter：**

·声明一个可变常量，常用于定义延时及宽度变量。

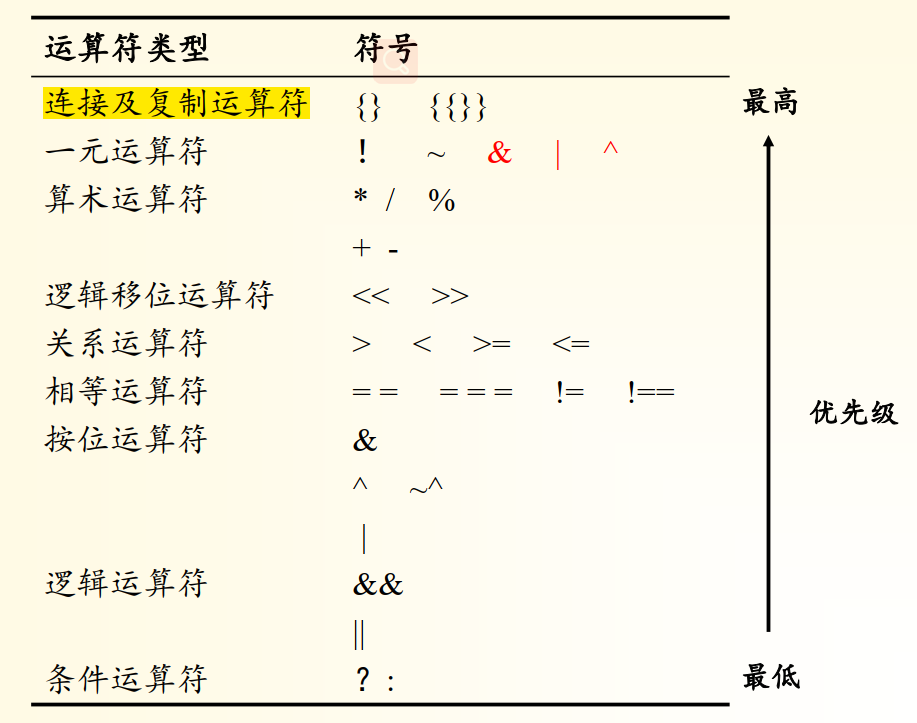




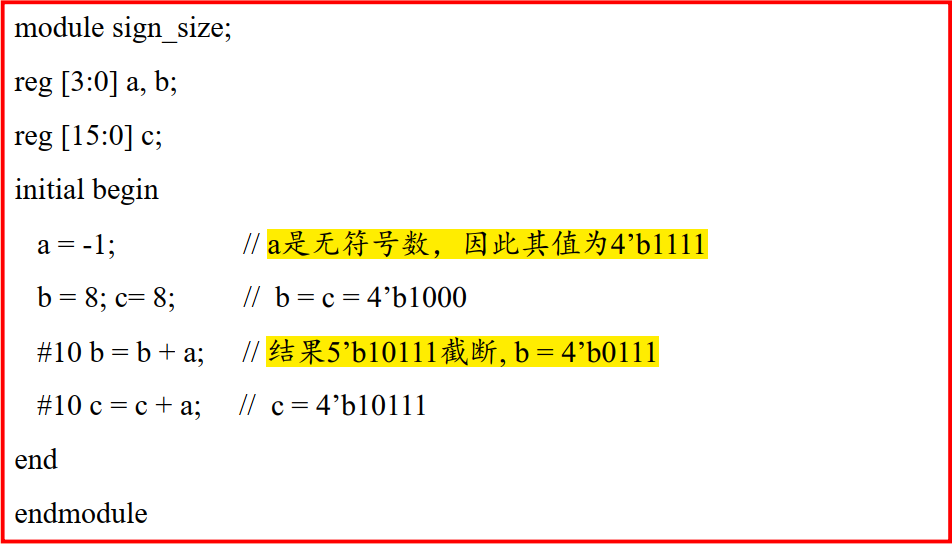
1. **Wire & reg：**



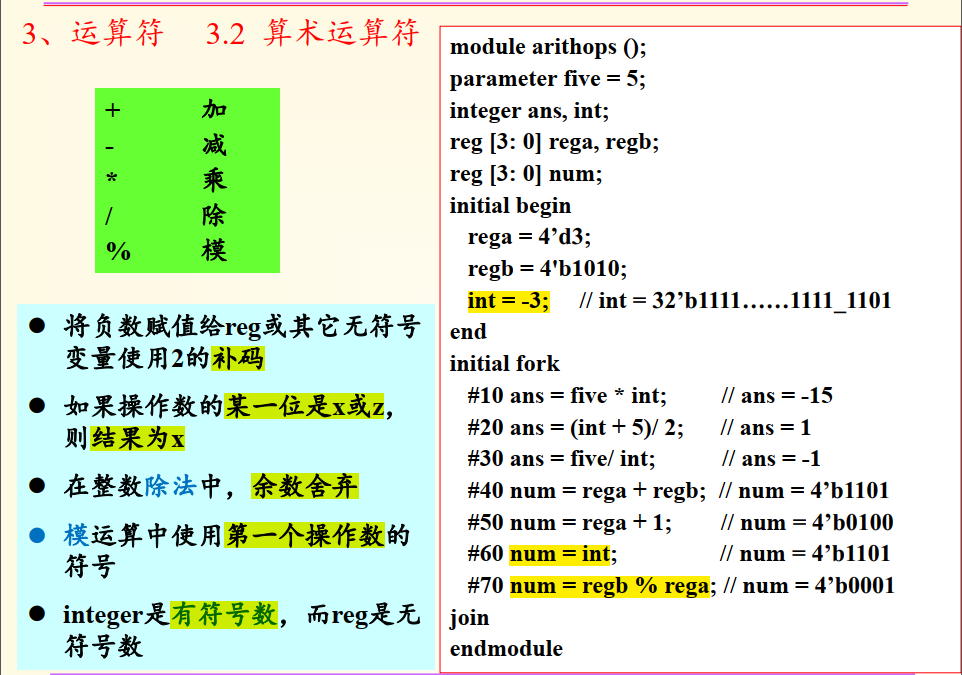
1. **运算符优先级：**







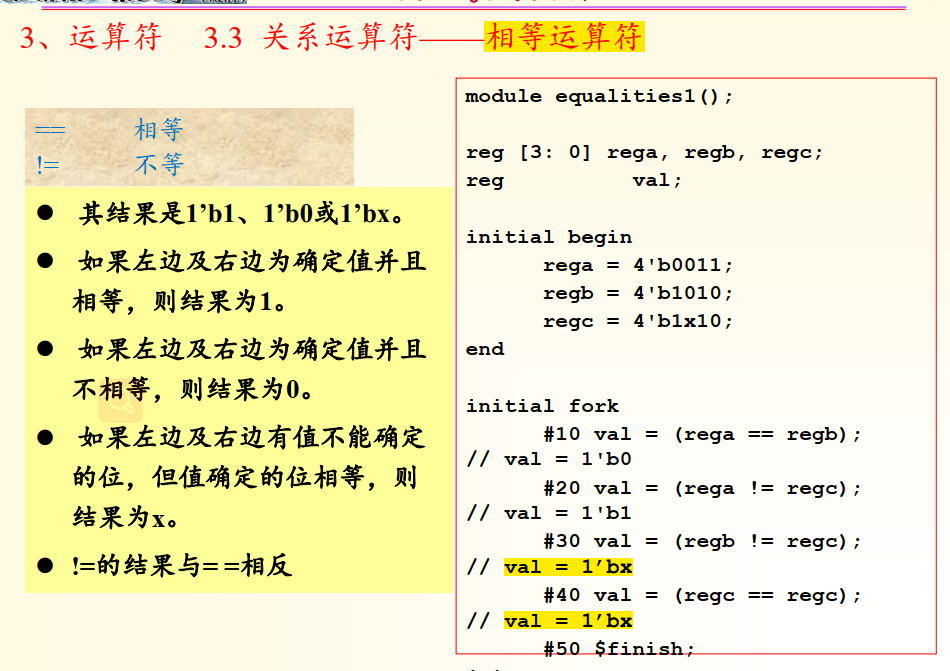
1. **算数运算符：**



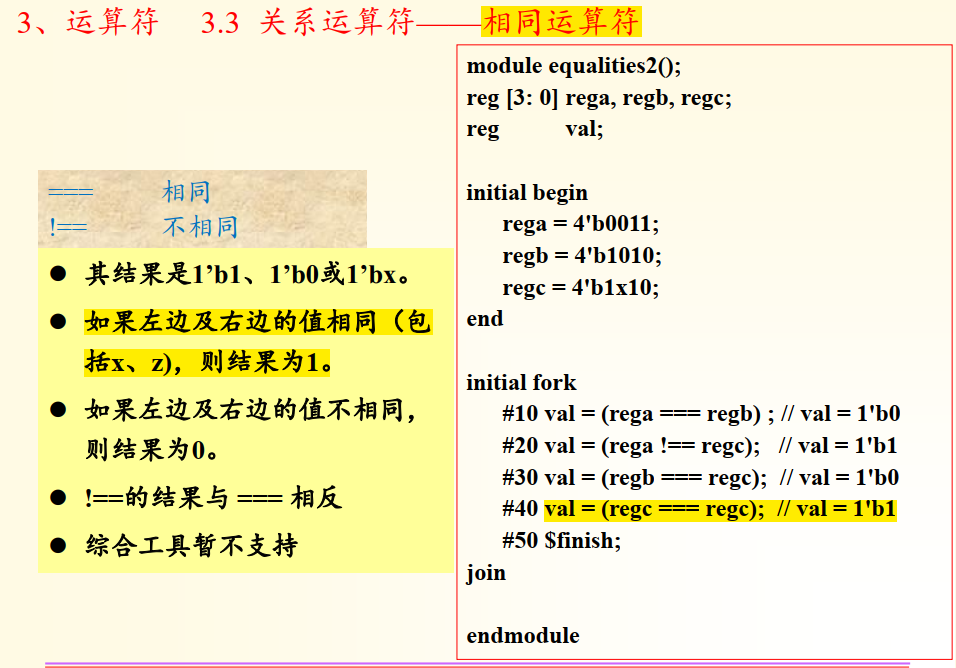
1. **关系运算符：**

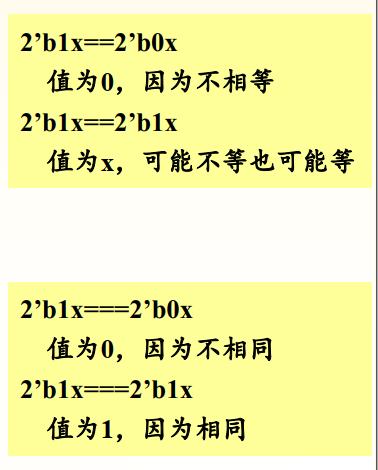
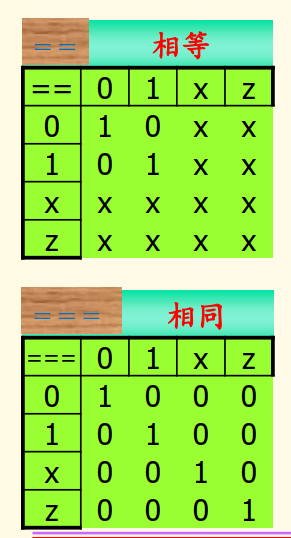


1. **相等运算符：**

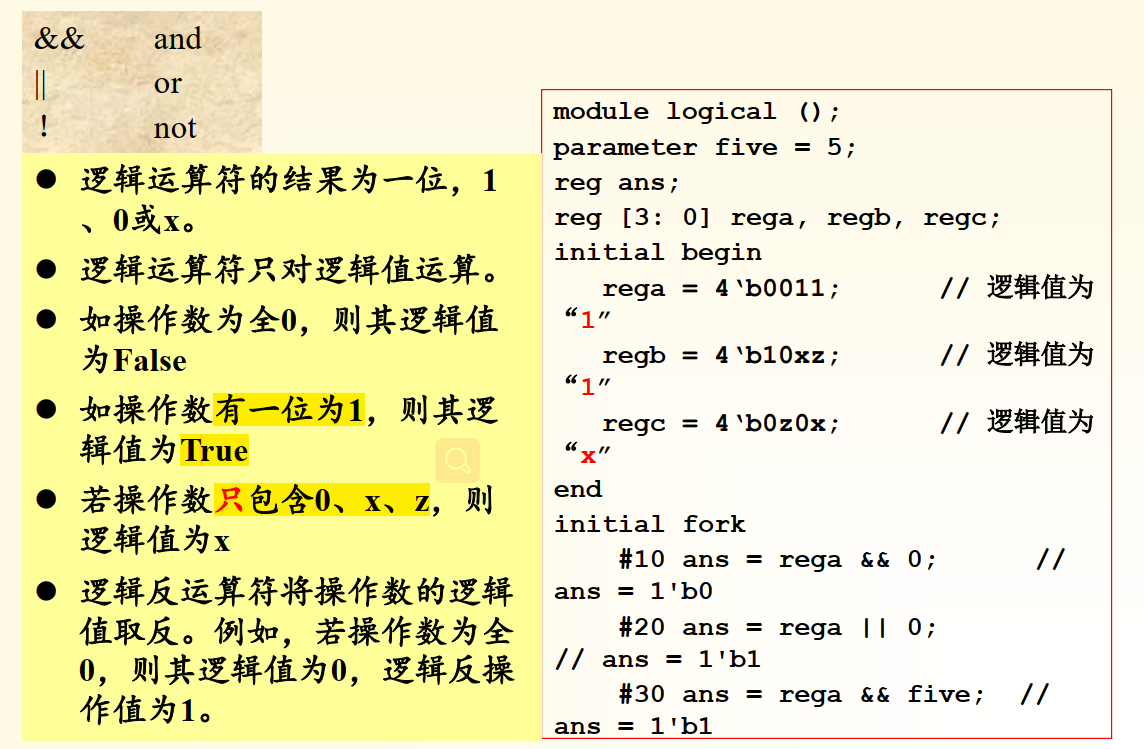


1. **相同运算符：**





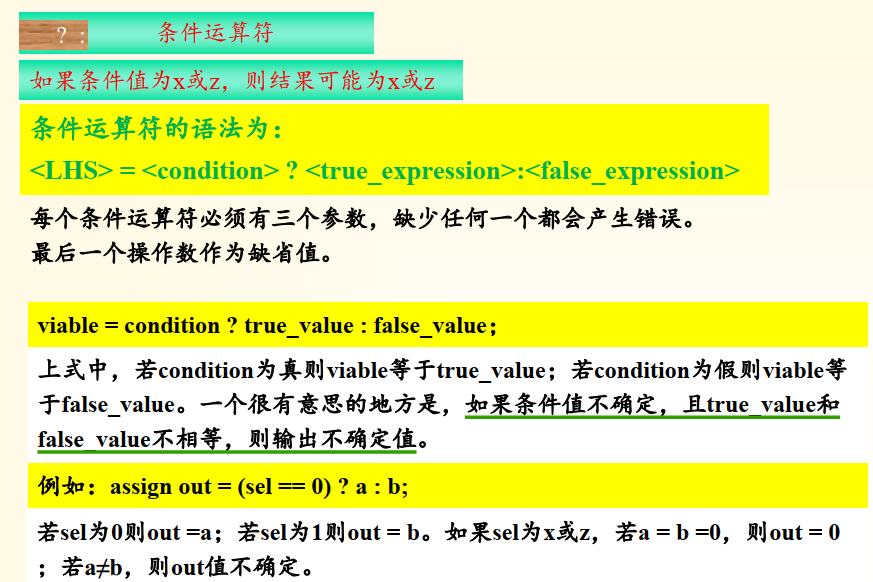
1. **逻辑操作符（注意和位运算符区分）**



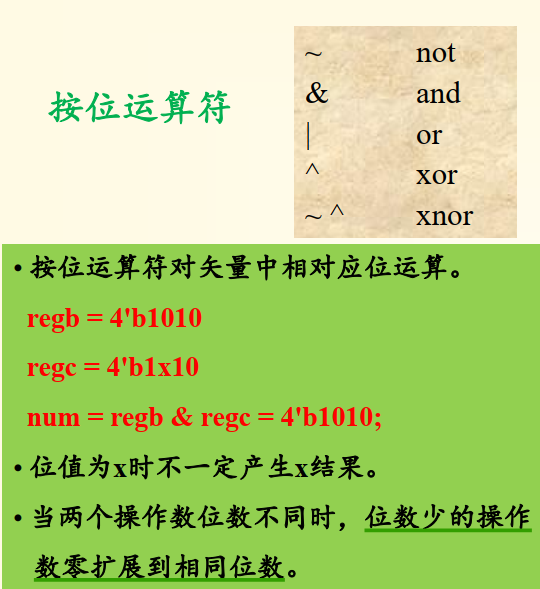
1. **移位运算符**



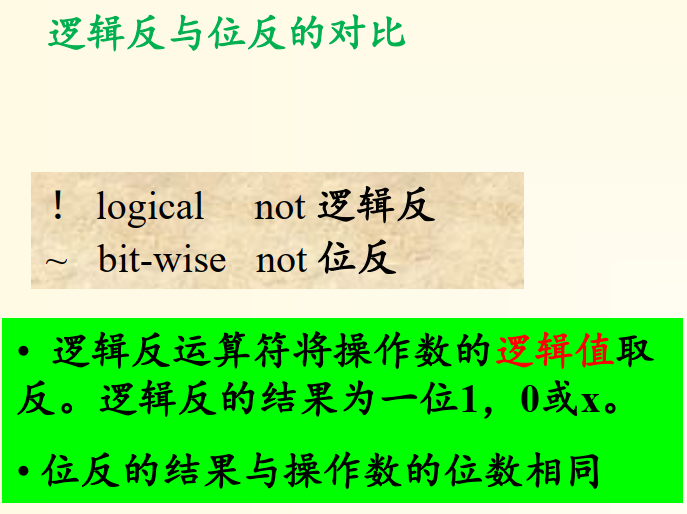
1. **条件运算符**



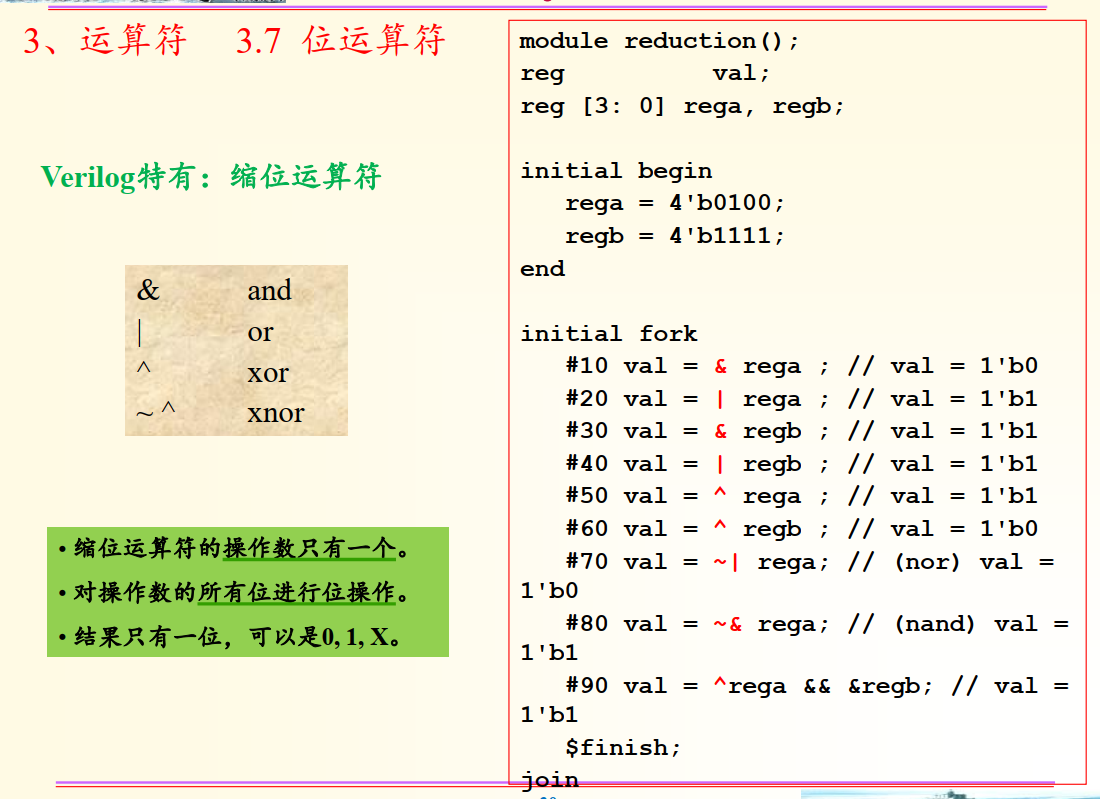
1. **位运算符**



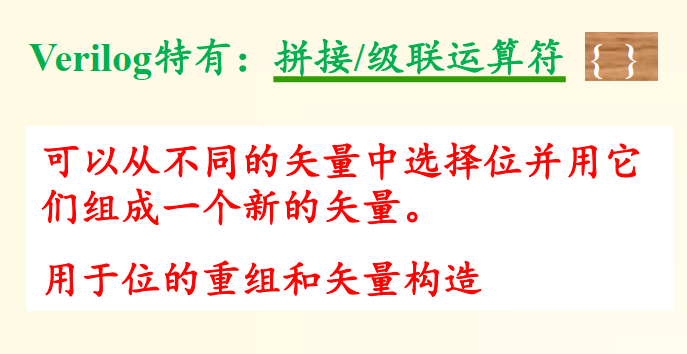
1. **逻辑反与位反**

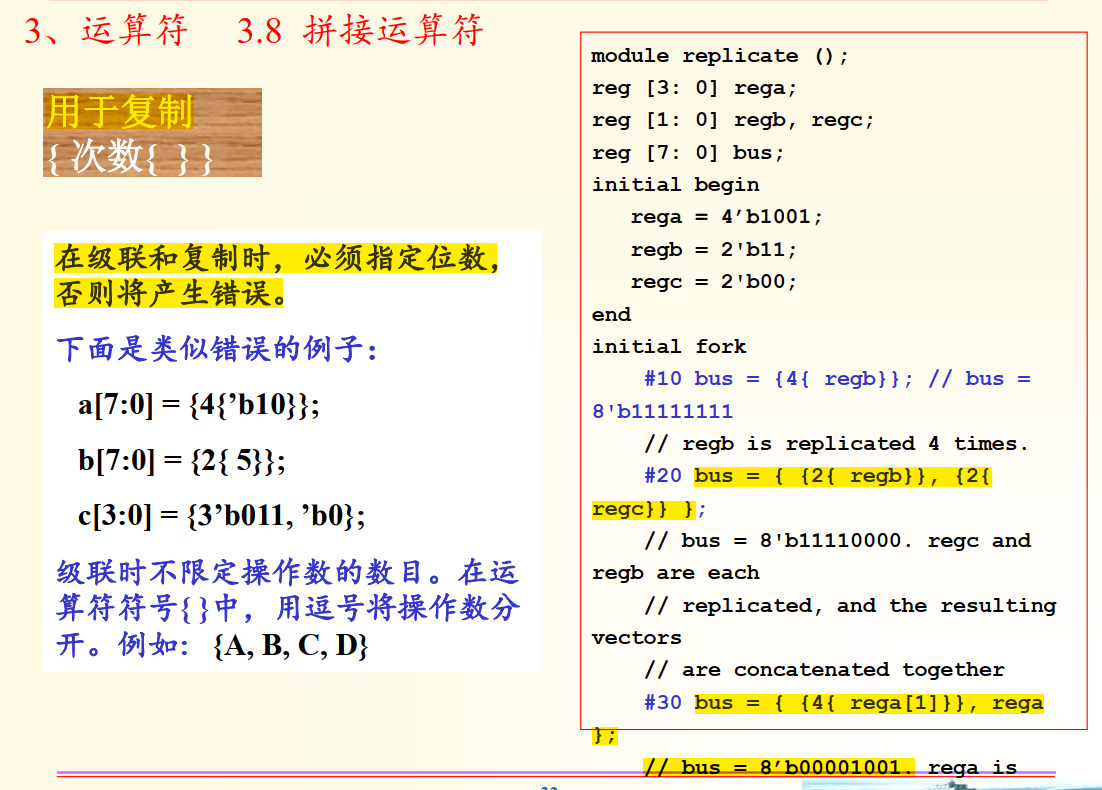


1. **缩位运算符**

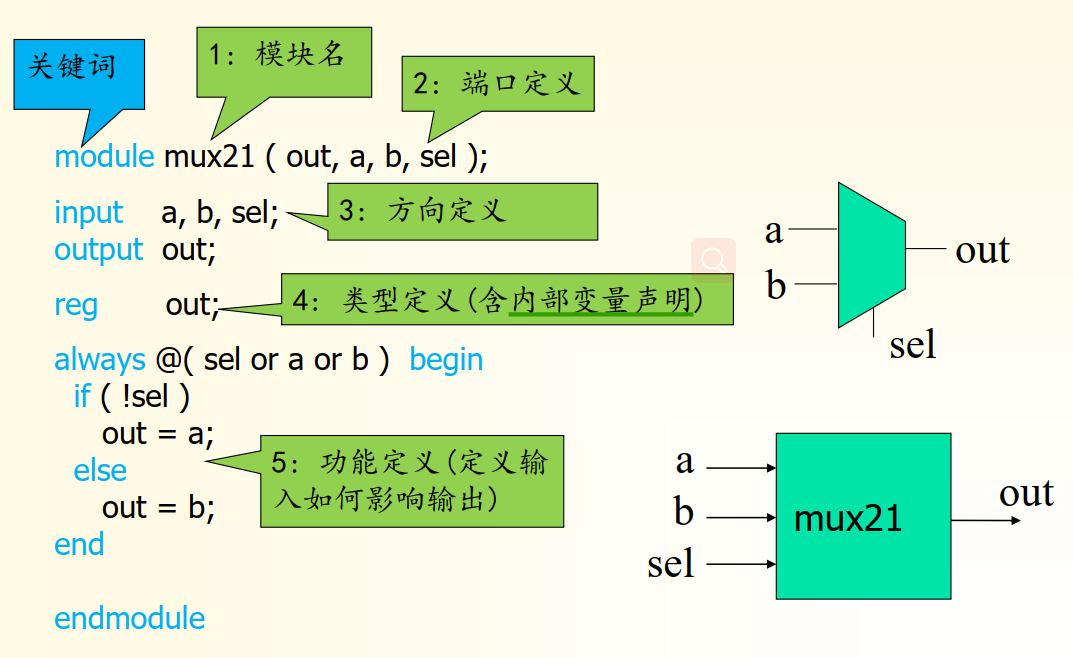


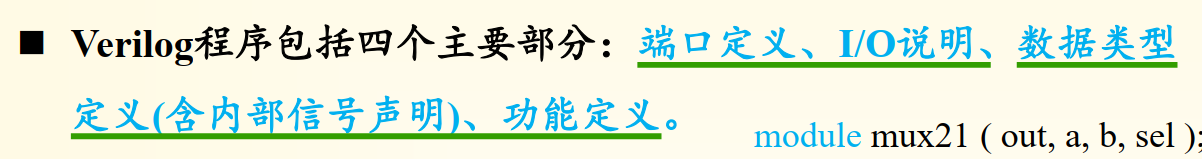
1. **拼接运算符**

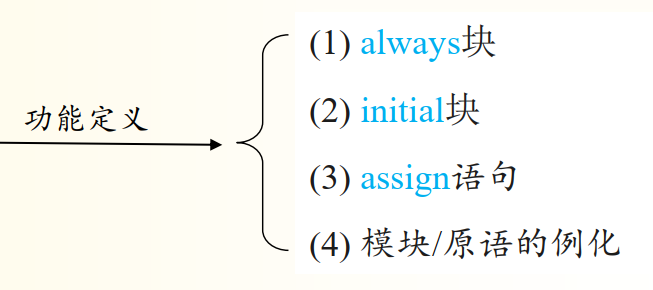




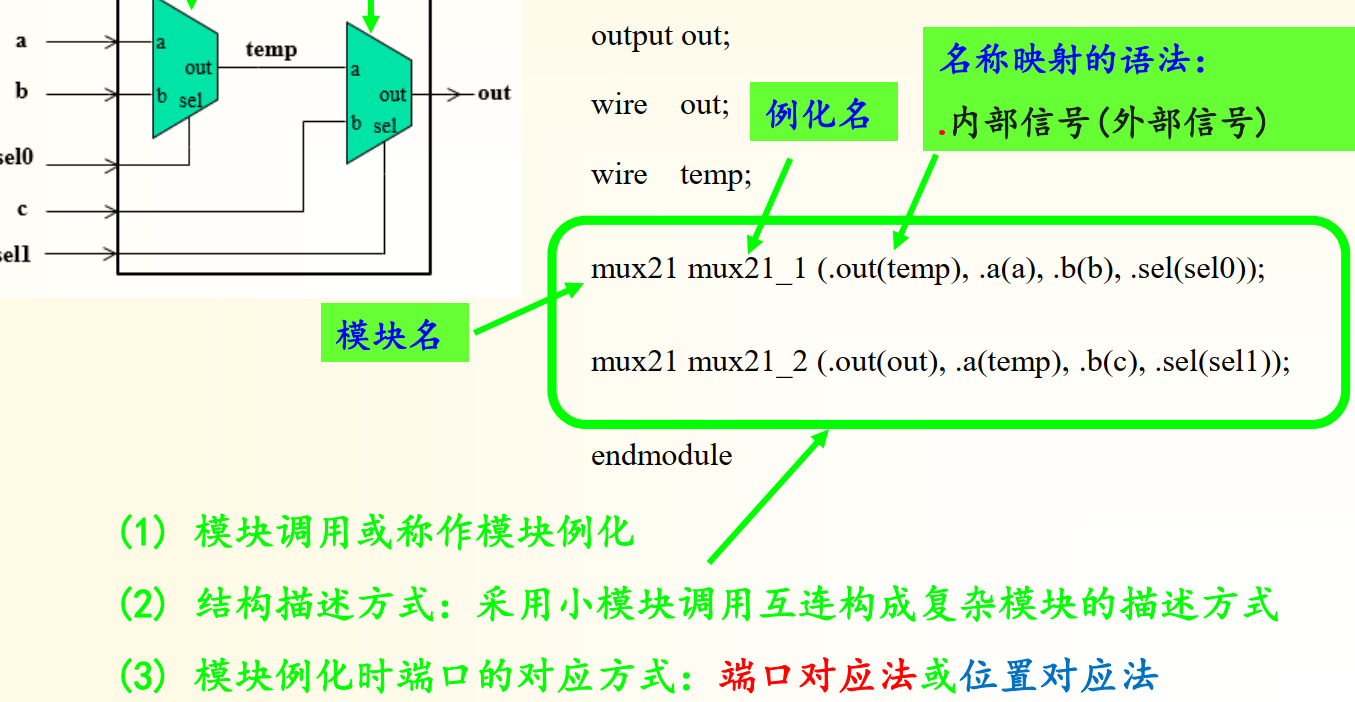
1. **模块化**

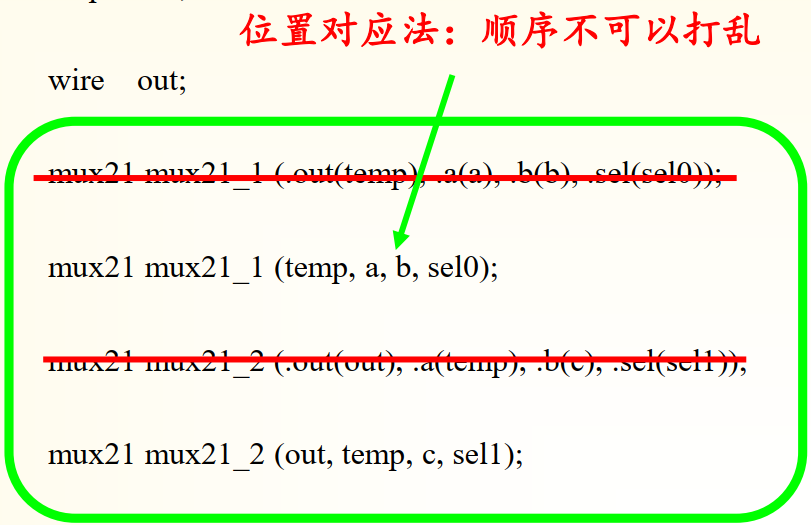




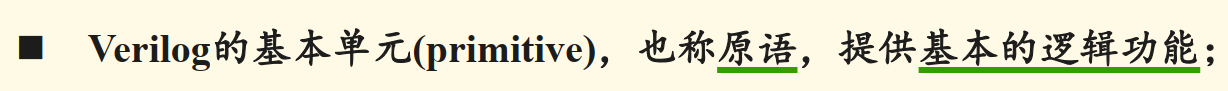


1. **模块例化**





1. **原语**

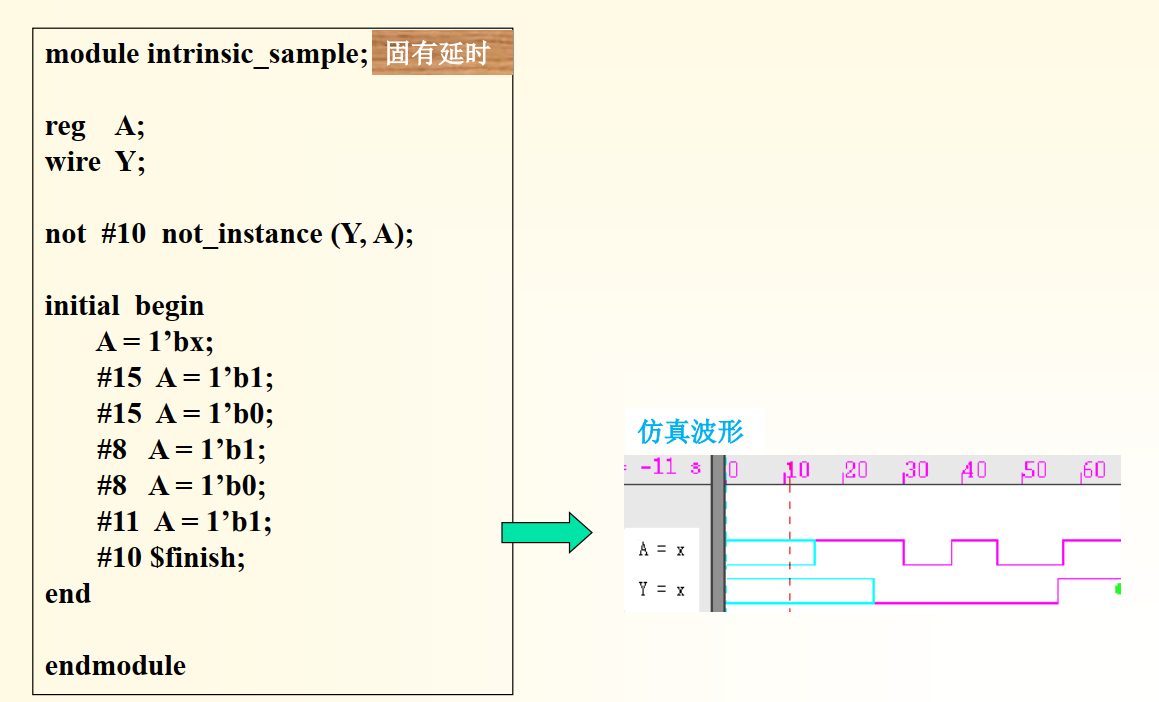


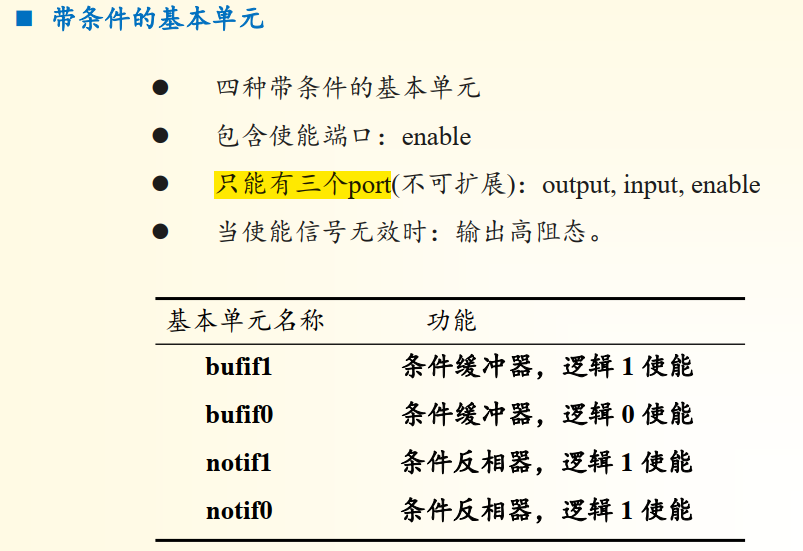


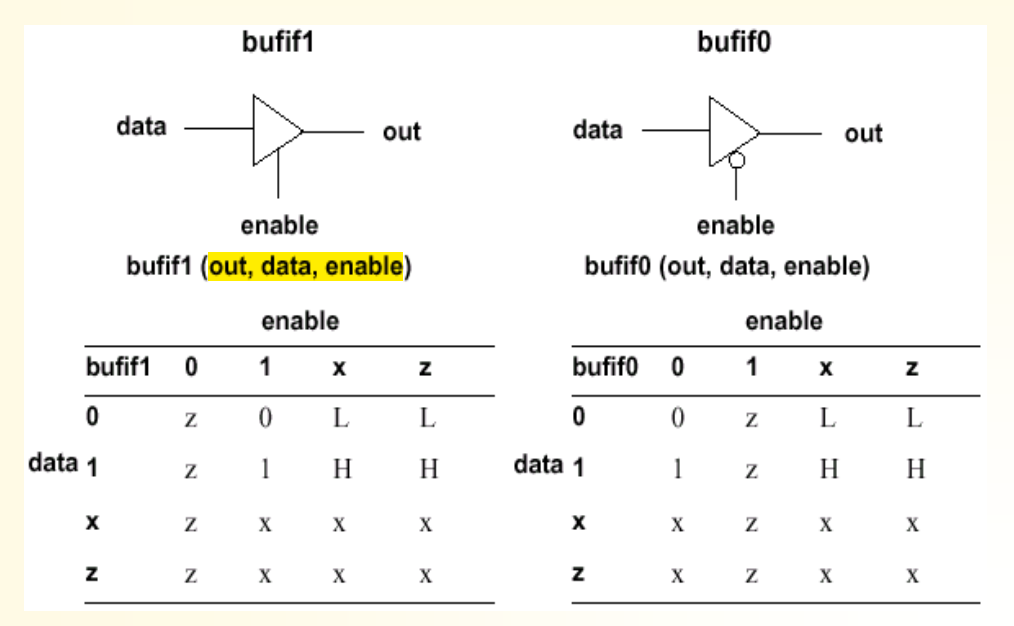


1. **基本单元的调用**

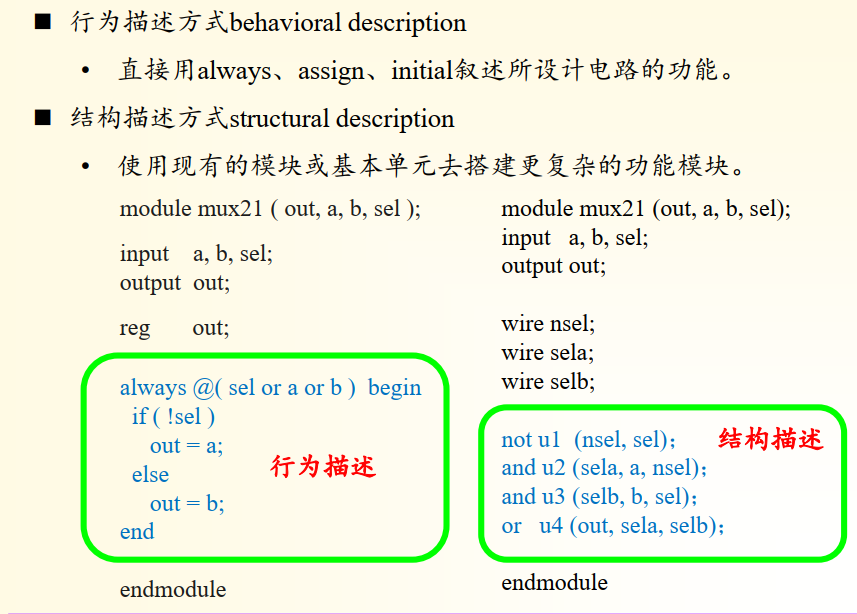




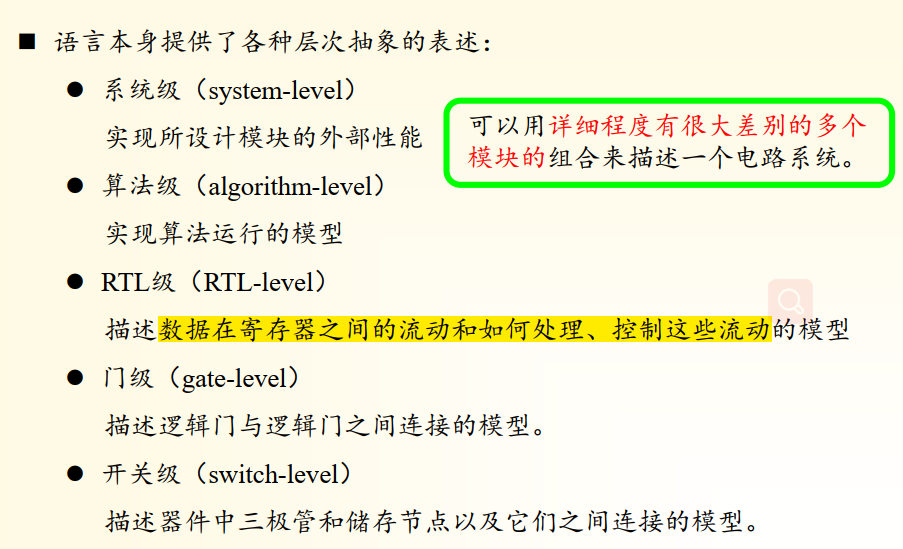




1. **建模的两种描述方式**

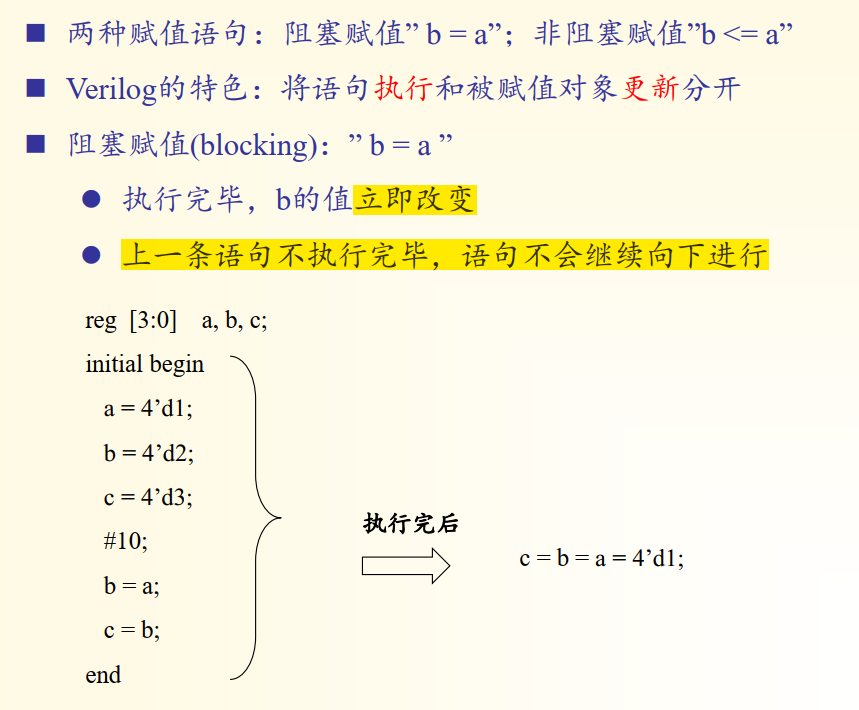


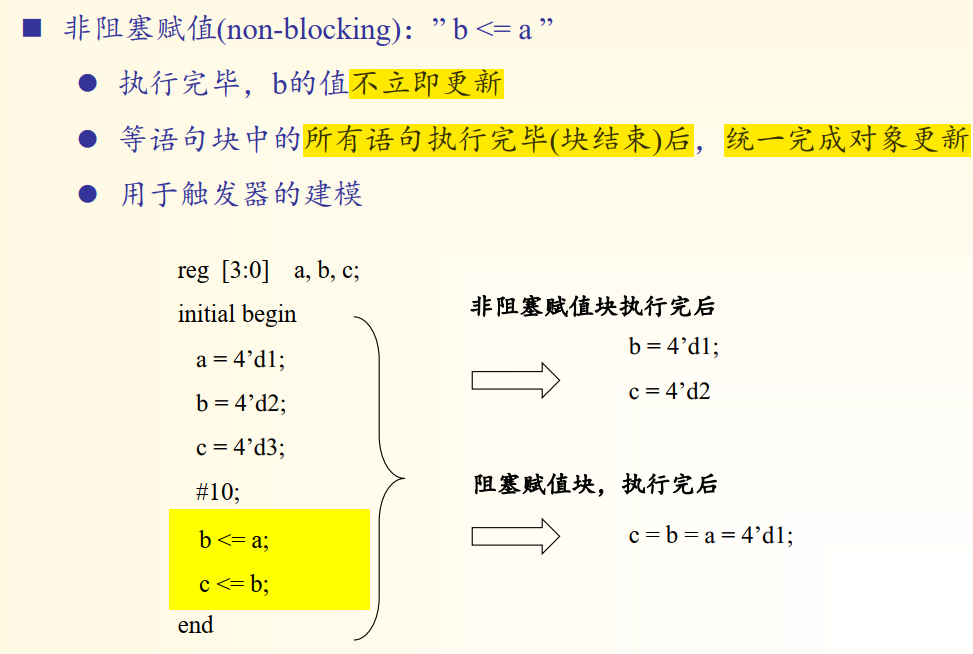
1. **Verilog的抽象级别：**

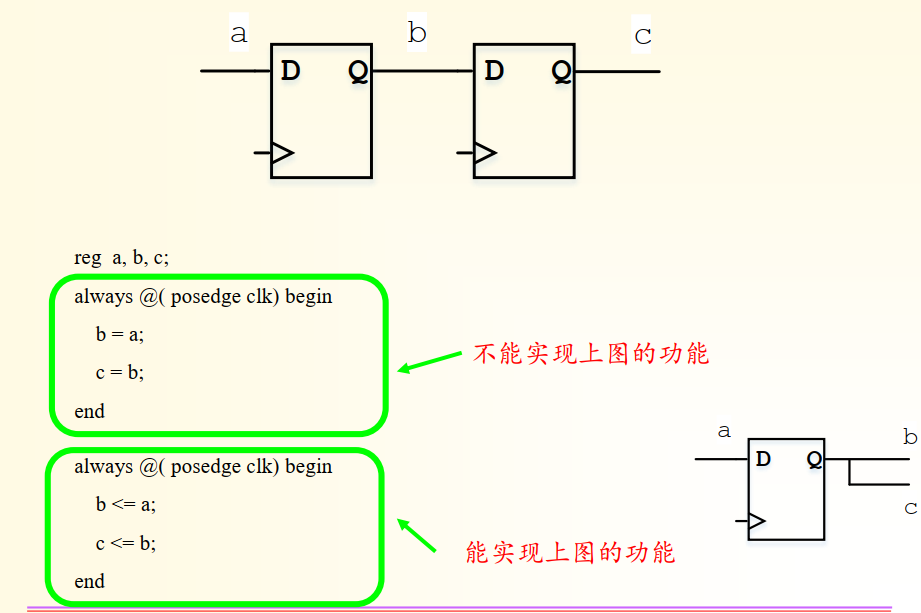


可以用/\*.....\*/和//...对Verilog HDL程序的任何部分作注释

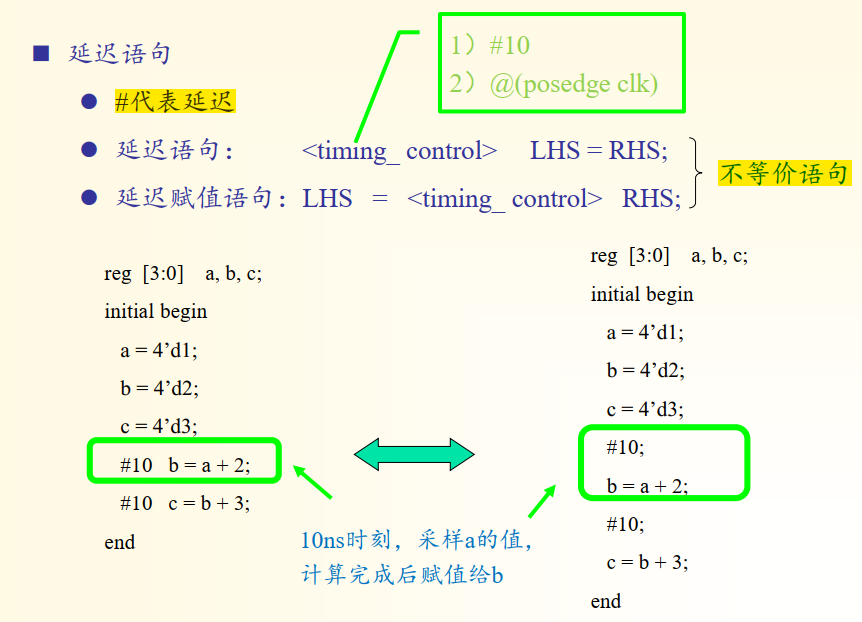
1. **赋值语句**

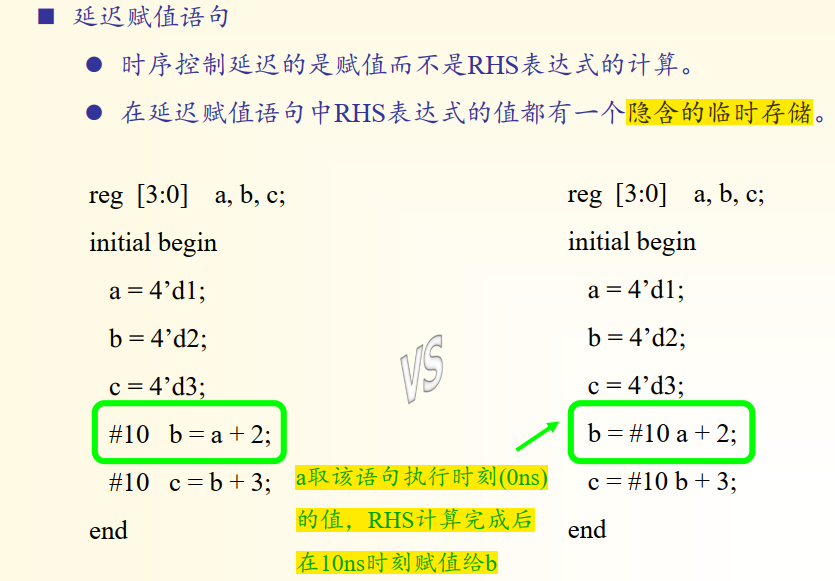


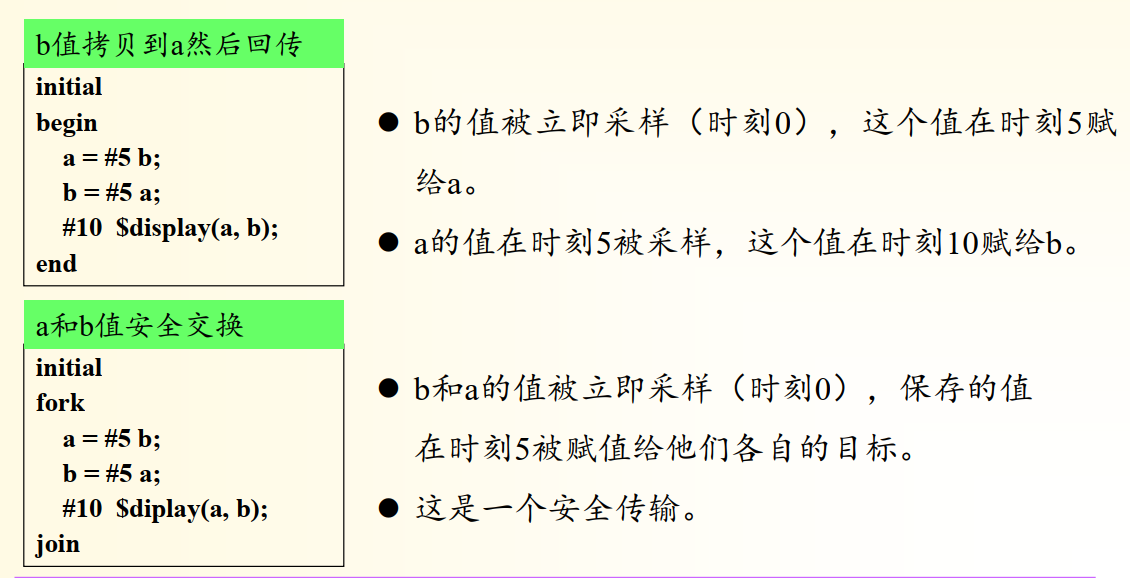




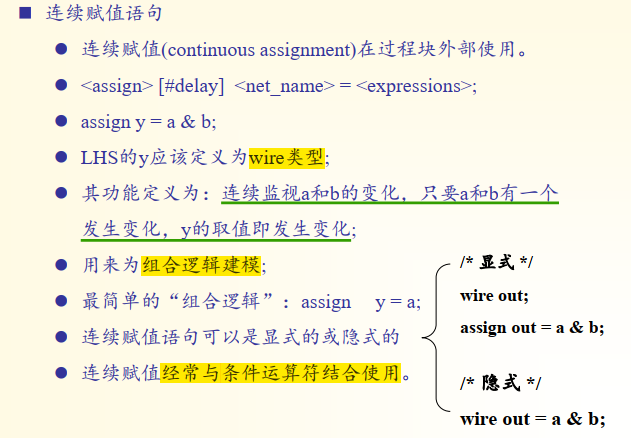
1. **延迟语句**



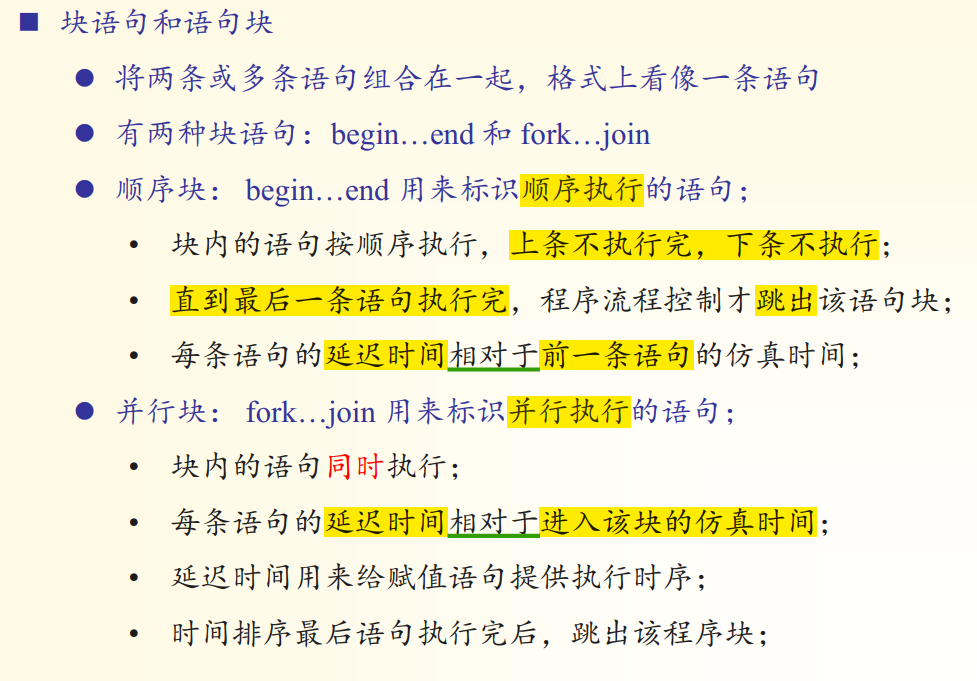


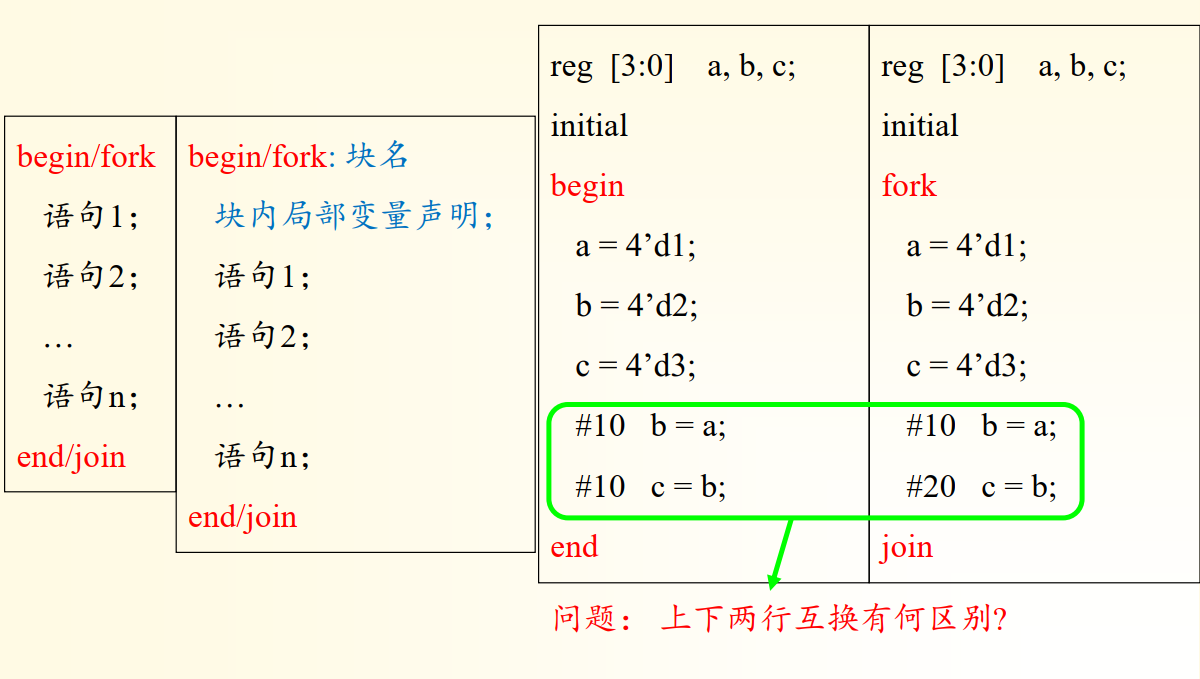


1. **连续赋值语句**

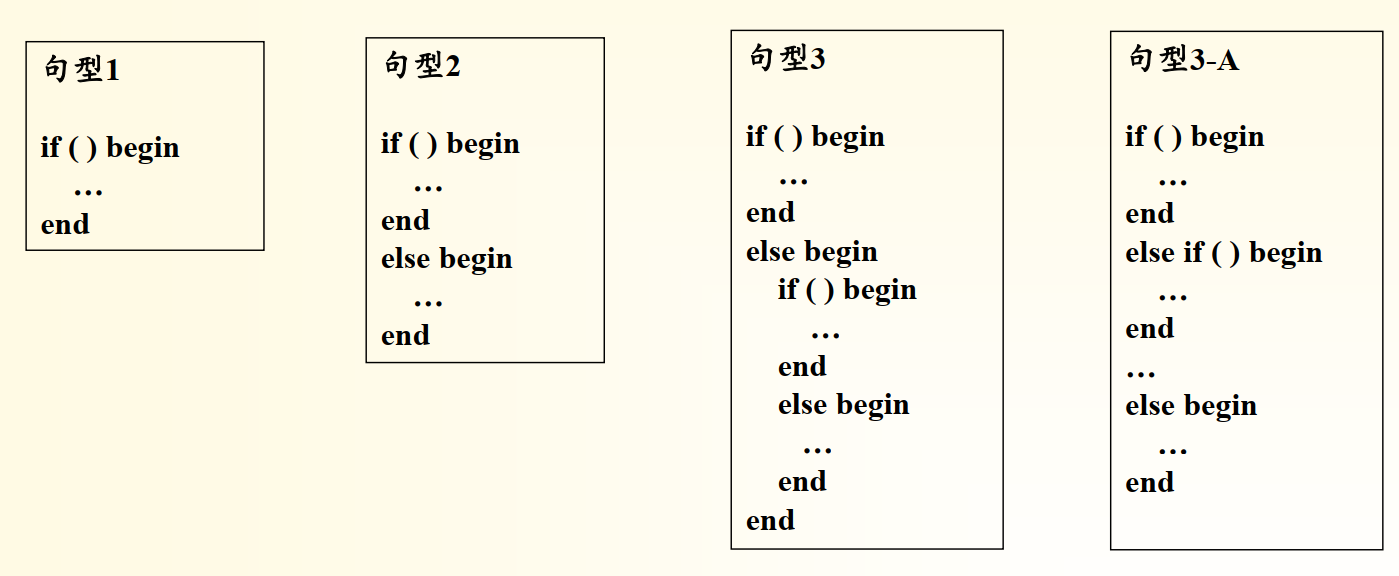


1. **块语句**

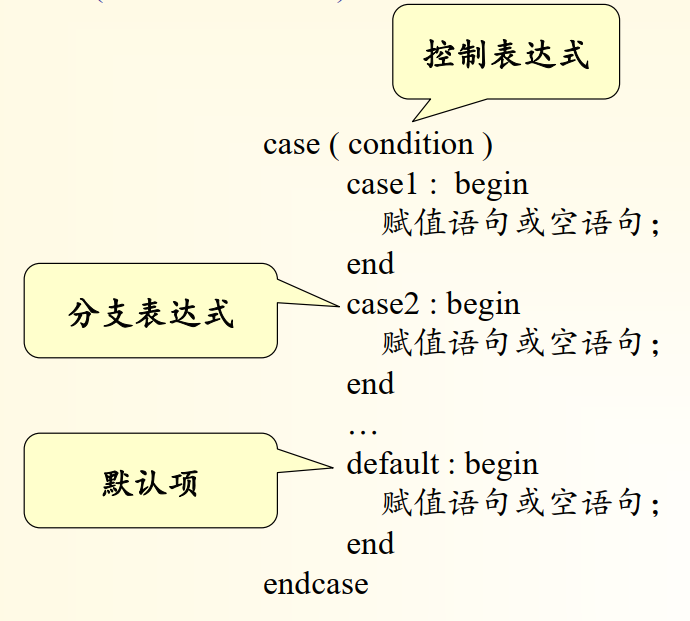


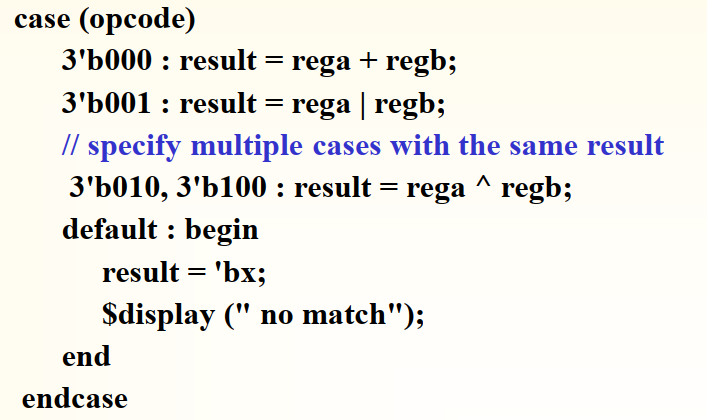


1. **条件语句**

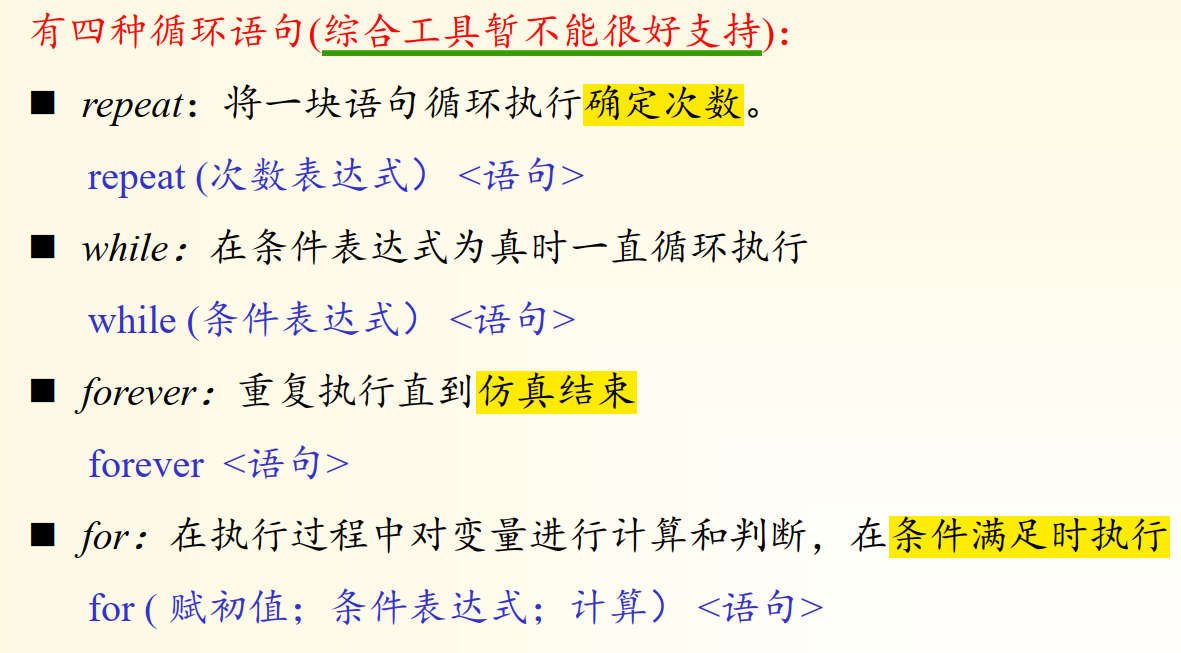


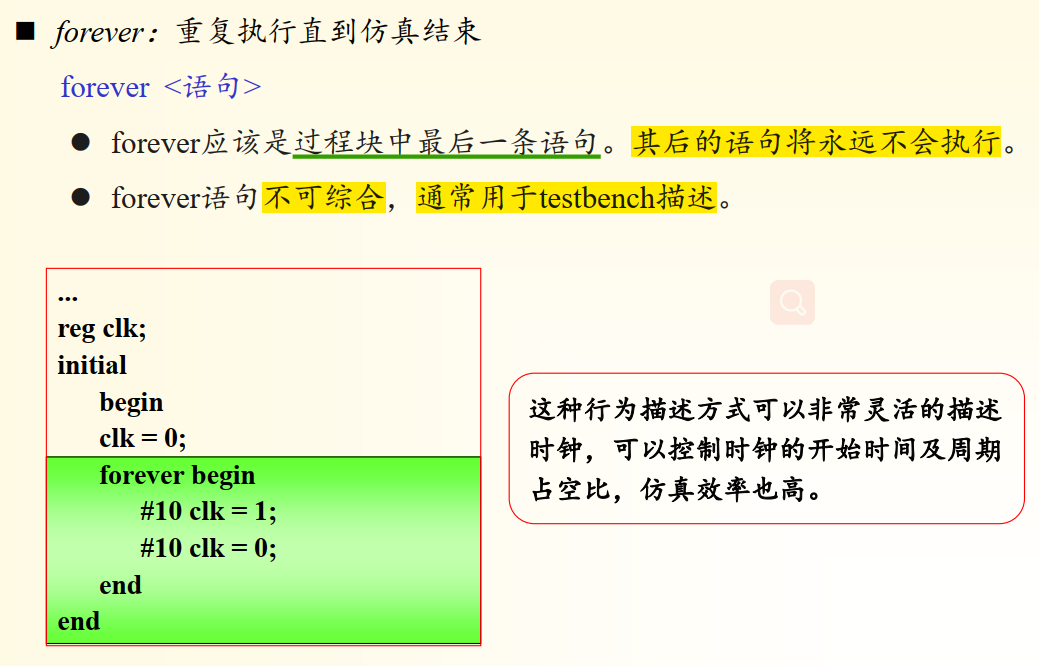
1. **Case语句**

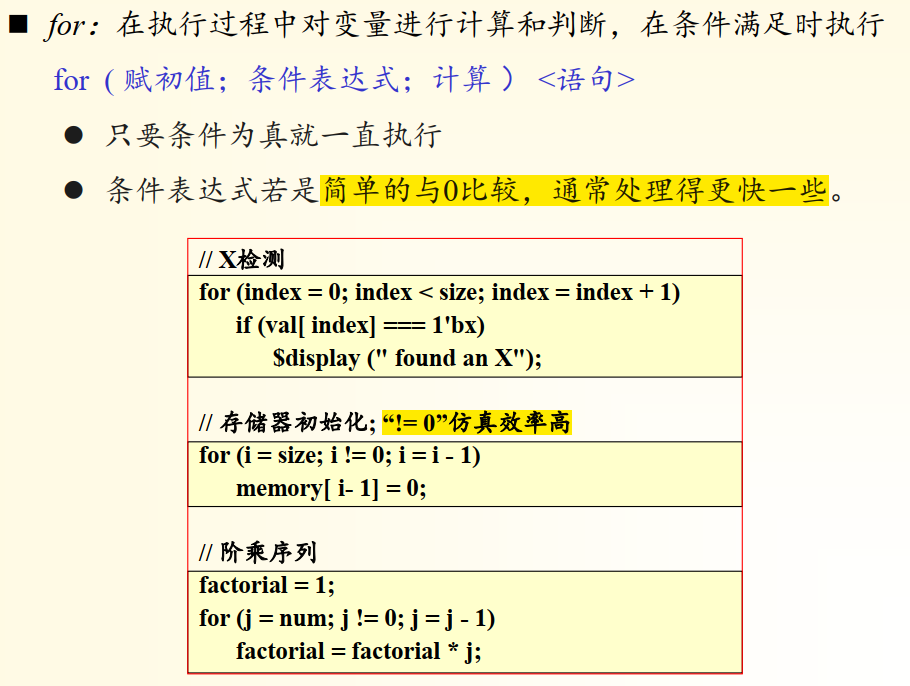




1. **循环语句**







1. **系统任务$dumpfile和$dumpvar**

**$dumpfile和$dumpvar是verilog语言中的两个系统任务，可以调用这两个系统任务来创建和将指定信息导入VCD文件. (什么是VCD文件? 答：VCD文件是在对设计进行的仿真过程中，记录各种信号取值变化情况的信息记录文件。EDA工具通过读取VCD格式的文件，显示图形化的仿真波形，所以，可以把VCD文件简单地视为波形记录文件.)下面分别描述它们的用法并举例说明之。**

**$dumpfile系统任务：为所要创建的VCD文件指定文件名。**

**举例（"//"符号后的内容为注释文字）：**

**initial**

**$dumpfile ("myfile.dump"); //指定VCD文件的名字为myfile.dump，仿真信息将记录到此文件**

**$dumpvar系统任务：指定需要记录到VCD文件中的信号，可以指定某一模块层次上的所有信号，也可以单独指定某一个信号。**

**典型语法为$dumpvar(level, module\_name); 参数level为一个整数，用于指定层次数，参数module则指定要记录的模块。整句的意思就是，对于指定的模块，包括其下各个层次(层次数由level指定)的信号，都需要记录到VCD文件中去。**

**举例：**

**initial**

**$dumpvar (0, top); //指定层次数为0，则top模块及其下面各层次的所有信号将被记录**

**initial**

**$dumpvar (1, top); //记录模块实例top以下一层的信号**

**//层次数为1，即记录top模块这一层次的信号**

**//对于top模块中调用的更深层次的模块实例，则不记录其信号变化**

**initial**

**$dumpvar (2, top); //记录模块实例top以下两层的信号**

**//即top模块及其下一层的信号将被记录**

**假设模块top中包含有子模块module1，而我们希望记录top.module1模块以下两层的信号，则语法举例如下：**

**initial**

**$dumpvar (2, top.module1); //模块实例top.module1及其下一层的信号将被记录**

**假设模块top包含信号signal1和signal2(注意是变量而不是子模块), 如我们希望只记录这两个信号，则语法举例如下：**

**initial**

**$dumpvar (0, top.signal1, top.signal2); //虽然指定了层次数，但层次数是不影响单独指定的信号的**

**//即指定层次数和单独指定的信号无关**

**我们甚至可以在同一个$dumpvar的调用中，同时指定某些层次上的所有信号和某个单独的信号，假设模块top包含信号signal1，同时包含有子模块module1，如果我们不但希望记录signal1这个独立的信号，而且还希望记录子模块module1以下三层的所有信号，则语法举例如下：**

**initial**

**$dumpvar (3, top.signal1, top.module1); //指定层次数和单独指定的信号无关**

**//所以层次数3只作用于模块top.module1, 而与信号**

**top.signal1无关**

**上面这个例子和下面的语句是等效的：**

**initial**

**begin**

**$dumpvar (0, top.signal1);**

**$dumpvar (3, top.module1);**

**end**

**$dumpvar的特别用法(不带任何参数)：**

**initial**

**$dumpvar; //无参数，表示设计中的所有信号都将被记录**

**最后，我们将$dumpfile和$dumpvar这两个系统任务的使用方法在下面的例子中综合说明，假设我们有一个设计实例，名为 i\_design，此设计中包含模块module1，模块module1下面还有很多层次，我们希望对这个设计进行仿真，并将仿真过程中模块module1及其以下所有层次中所有信号的变化情况，记录存储到名为mydesign.dump的VCD文件中去，则例示如下：**

**initial**

**begin**

**$dumpfile ("mydesign.dump"); //指定VCD文件名为mydesign.dump**

**$dumpvar (0, i\_design.module1); //记录i\_design.module1模块及其下面层次中所有模块的所有信号**

**end**