1. VHDL语言基础
   1. 标识符

标识符用来定义常量、变量、信号、端口、子程序或参数的名字，由字母(A~Z, a~z)、数字(0~9)和下划线(\_)字符组成。

要求：

1. 首字符必须是字母；
2. 末字符不能为下划线；
3. 不允许出现两个连续的下划线；
4. 不区分大小写；
5. VHDL定义的保留字（关键字），不能用作标识符；
6. 标识符字符最长可以是32个字符。

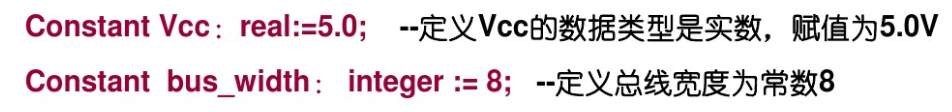
关键字：关键字是VHDL中具有特别含义的单词，只能作为固定用途，用户不能用其作为标识符。例如：  
ABS, ACCESS, AFTER, ALL, AND, ARCHITECTURE, ARRAY, ATTRIBUTE, BEGIN, BODY, BUFFER, BUS, CASE, COMPONENT, CONSTANT, DISCONNECT, DOWNTO, ELSE, ELSIF, END, ENTITY, EXIT, FILE, FOR, FUNCTION, GENERIC, GROUP, IF, INPURE, IN, INOUT, IS, LABEL, LIBRARY, LINKAGE, LOOP, MAP, MOD, NAND, NEW, NEXT, NOR, NOT, NULL, OF, ON, OPEN, OR, OTHERS, OUT

* 1. 数据对象

数据对象包括常量、变量、信号和文件四种类型。

* + 1. 常量Constant

常量是对某一常量名赋予一个固定的值，而且只能赋值一次。通常赋值在程序开始前进行，该值的数据类型则在说明语句中指明。

Constant 常量名: 数据类型: = 表达式

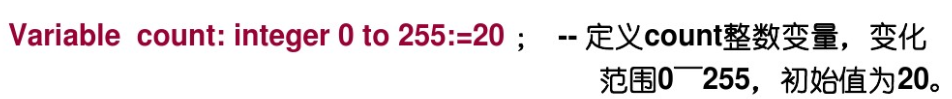
常量所赋的值应和定义的数据类型一致；

常量在程序包、实体、构造体或进程的说明性区域内必须加以说明。定义在程序包内的常量可供所含的任何实体、构造体所引用，定义在实体说明内的常量只能在该实体内可见，定义在进程说明性区域中的常量只能在该进程内可见。

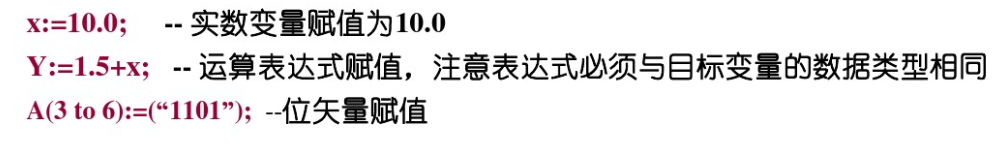
* + 1. 变量Variable

变量只能在进程语句、函数语句和过程语句结构中使用。变量的赋值是直接的，非预设的，分配给变量的值立即成为当前值，变量不能表达“连线”或存储元件，不能设置传输延迟量。

Variable 变量名: 数据类型: = 初始值



目标变量名: = 表达式



* + 1. 信号Signal

信号表示逻辑门的输入或输出，类似于连接线，也可以表达存储器元件的状态。信号通常在构造体、程序包和实体中说明。

 Signal 信号名: 数据类型: = 初始值

信号赋值语句：  
目标信号名 <= 表达式；

* 1. 数据类型
     1. VHDL的预定义数据类型

在VHDL标准程序包STANDARD中定义好，实际使用过程中，已自动包含进VHDL源文件中，不需要通过USE语句显示调用。

1. 布尔(Boolean)

TYPE BOOLEAN IS (FALSE, TRUE); --取值为FALSE或TRUE，不是数值，不能运算，一般用于关系运算符

1. 位(Bit)

TYPE BIT IS (‘0’, ‘1’); --取值为0或1，用于逻辑运算

1. 位矢量(Bit\_Vector)

TYPE BIT\_VECTOR IS ARRAY(Natural range<>)OF BIT; --基于Bit类型的数组，用于逻辑运算

1. 字符(Character)

TYPE CHARACTER IS (NUL, SOH, STX, ..., ‘ ’, ‘!’, ...); --通常由’’引起来，区分大小写；

1. 字符串(String)

VARIABLE string\_var: STRING(1 TO 7);

String\_var:= “A B C D”; --通常由””引起来，区分大小写；

1. 整数(Integer)

取值范围 –(2^31 – 1) —(2^31 – 1)，可用32位有符号二进制数表示

在实际应用中，VHDL仿真器将Integer作为有符号数处理，而VHDL综合器将Integer作为无符号数处理；

要求用RANGE子句为所定义的数限制范围，以便根据范围来决定表示此信号或变量的二进制数的位数。

1. 实数(Real)

取值范围 -1.0E^38— +1.0E^38，仅用于仿真不可综合

1. 时间(Time)

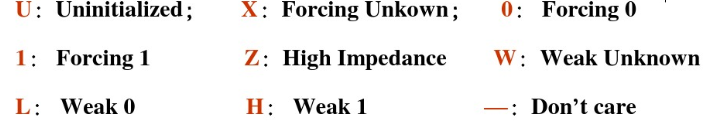
物理量数据，完整地包括整数和单位两个部分，用至少一个空格隔开，仅用于仿真不可综合

1. 错误等级(Severity Level)

表示系统状态，仅用于仿真不可综合；

TYPE severity\_level IS(NOTE, WARNING, ERROR, FAILURE);

IEEE预定义的标准逻辑位与矢量：

* 1. 标准逻辑位（Std\_Logic）
  2. 标准逻辑位矢量（Std\_Logic\_vector）

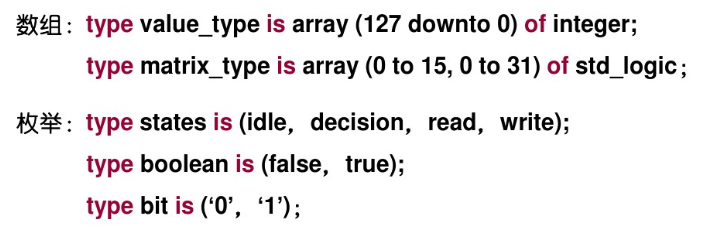
基于Std\_Logic类型的数组；

使用Std\_Logic和Std\_Logic\_Vector要调用IEEE库中的Std\_Logic\_1164程序包；就综合而言，能够在数字器件中实现的是“-、 0、 1、 Z”四种状态。

在条件语句中，必须要全面考虑Std\_Logic的所有可能取值情况，否则综合器可能会插入不希望的锁存器。

用户自定义：  
TYPE 数据类型名 IS 数据类型定义 OF 基本数据类型

或 TYPE 数据类型名 IS 数据类型定义



SUBTYPE 子类型名 IS 基本数据类型定义 RANGE 约束范围

subtype digit is integer range 0 to 9;

* 1. 数据类型转换

VHDL为强类型语言，不同类型的数据不能进行运算或直接赋值。

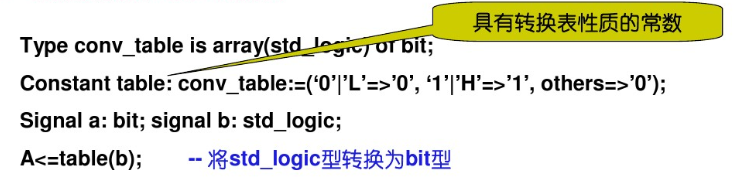
1. 类型标记法

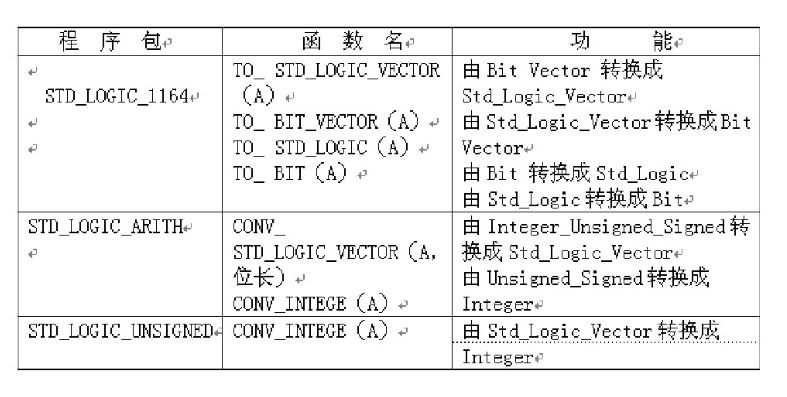
Variable A:integer; Variable B:real;

A = integer(B); B = real(A);

1. 函数法

Conv\_integer(A); --由std\_logic转换为integer型，在std\_logic\_unsigned包。

1. 常数转换法/常量转换法

在”STD\_LOGIC\_1164”、”STD\_LOGIC\_ARITH”、”STD\_LOGIC\_UNSIGNED”的程序包中提供的数据类型变换函数。

属性

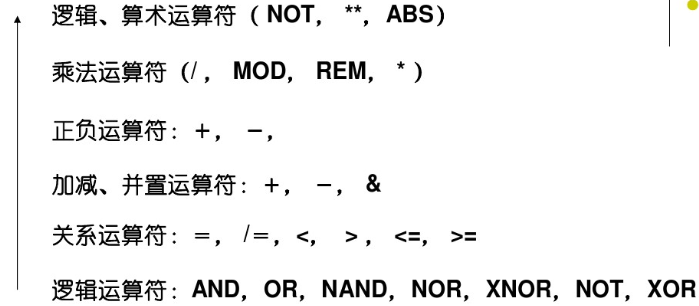
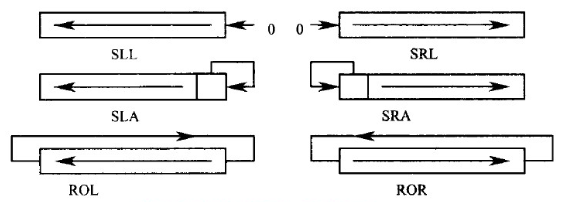
属性提供的是关于信号、类型等的指定特性。

event: 若属性对象有事件发生，则生成布尔值”true”，常用来检查时钟边沿是否有效。

上升沿: Clock’EVENT AND Clock = ‘1’

range: 生成一个限制性数组对象的范围

left: 生成数据类型或者数据子类型的左边界值

 运算符优先级：  
移位运算符左边为一维数组，其类型必须是BIT或BOOLEAN，右边必须是整数移位次数为整 数的绝对值。

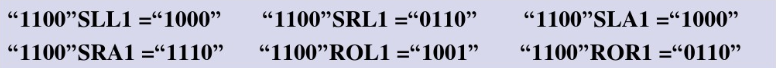
SLL: 将位向量左移，右边移空位补零；

SRL: 将位向量右移，左边移空位补零；

SLA: 将位向量左移，右边第一位的数值保持原值不变；

SRA: 将位向量右移，左边第一位的数值保持原值不变；

ROL和ROR: 自循环左右移位。

例：

取余运算（a REM b）的符号与a相同，其绝对值小于b的绝对值。

例如：（-5）REM 2 = （-1） 5 REM 2 = （1）

取模运算（a mod b）的符号与b相同，其绝对值小于b的绝对值。

例如：（-5）mod 2 = 1 5 mod （-2） = （-1）