




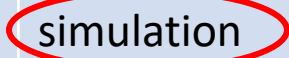




VHDL Types (CPE 1500)



- Basic Types – used in CPE 1500

Type	Class	Synthesizable	Usage
boolean	 enumerated	Y	indirect - comparisons
bit	 enumerated	Y	not recommended
character	 enumerated	Y	not recommended
severity_level	 enumerated	N	simulation
integer	integer	Y	array indices, compile time calculation, simulation
natural	integer (subtype)	Y	
positive	integer (subtype)	Y	
real	 floating point	N	compile time calculation, simulation
time	physical	N	 simulation
bit_vector	 array of bit	Y	not recommended
string	 array of character	Y	simulation (file read/write)

VHDL Types (CPE 1500)

- Synthesis Types – `std_logic_1164`

- 4 types

- `std_ulogic` 
- `std_ulogic_vector` 
- `std_logic` – resolved* version of `std_ulogic`
- `std_logic_vector` – resolved* version of `std_ulogic_vector`

- 9 metalogical values

Value	Definition	Synthesizable
'U'	un-initialized	N
'X'	Forcing Unknown	N
'0'	Forcing 0	Y
'1'	Forcing 1	Y
'Z'	High Impedance	Y
'W'	Weak Unknown	N
'L'	Weak 0	N
'H'	Weak 1	N
'-'	Don't Care	N / Y ***

* Resolved : allows high impedance signals to “resolve” to a 1 or 0, ie. have multiple drivers

** Comparison of `std_logic_vectors` can return unexpected results

*** Don't Care can be used for synthesis in special situations

- Operators

- Comparison** : `=`, `/=`, `<`, `<=`, `>`, `>=`
- Boolean: `not`, `and`, `or`, `nand`, `nor`, `xor`, `xnor`
- Shifting: `srl`, `sll`, `rol`, `ror`
- Concatenation: `&`

VHDL Types (CPE 1500)

- Numeric Types – `numeric_std`
 - 2 types
 - `signed` array of `std_logic` (analogous to a `std_logic_vector`)
 - `unsigned` array of `std_logic` (analogous to a `std_logic_vector`)
 - Values
 - `signed` is interpreted as 2's complement (positive and negative)
 - `unsigned` is interpreted as unsigned magnitude (always positive)
 - Operators
 - Comparison: `=`, `/=`, `<`, `<=`, `>`, `>=`
 - Boolean: `not`⁺⁺, `and`, `or`, `nand`, `nor`, `xor`, `xnor`
 - Arithmetic[†]: `sign`⁺⁺⁺⁺, `abs`⁺⁺⁺⁺, `+`, `-`, `*`⁺⁺⁺, `/`⁺⁺⁺, `mod`, `rem`, `**`⁺⁺⁺⁺⁺
 - Shifting: `srl`, `sll`, `rol`, `ror`
 - Concatenation: `&`
 - Functions
 - `resize` resize `unsigned` using zero extension
resize `signed` using sign extension

[†] Arithmetic operators other than multiplication preserve the length of the result vector i.e. wrap

⁺⁺ negation of 2's complement most negative value will return the most negative value

⁺⁺⁺ `*` and `/` will create large logical solutions

⁺⁺⁺⁺ `signed` only

⁺⁺⁺⁺ `only use with a base of 2`