VHDL Types

- Basic Types

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

- 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

* 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

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

VHDL Types

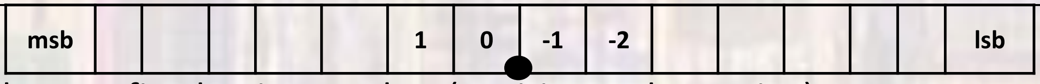
- Numeric Types – numeric_std
  - 2 types
    - signed array of std_logic (analogous to a std_logic_vector)
    - unsigned array of std_logic (analogous to a std_logic_vector)
  - Values
    - signed is interpreted as 2's complement (positive and negative)
    - unsigned is interpreted as unsigned magnitude (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

VHDL Types

- Fixed Point Types – fixed_pkg
  - 2 types
    - sfixed array of std_logic
    - ufixed array of std_logic
  - Values
    - sfixed is interpreted as 2's complement fixed point number (positive and negative)
    - unsigned is interpreted as unsigned magnitude fixed point number (always positive)
  - Operators
    - Comparison: =, /=, <, <=, >, >=
    - Boolean: not, and, or, nand, nor, xor, xnor
    - Arithmetic: †: sign -††††, abs††††, +, -, *†††, /†††, mod, rem
  - Functions
    - resize resize unsigned using zero extension
      resize signed using sign extension
    - is_negative determines if the value is less than 0
    - add_carry create a carry out during addition
    - scalb scales the value by a factor of two (shift)
    - is_negative determines if the value is less than 0
    - maximum, minimum provides the largest or smallest of two values
    - saturate provides the largest possible value for the given range

Positive indices represent the integer portion of the number
Negative indices represent the fractional portion of the number
signal foo: sfixed(7 downto -8);



†††* and / will create large logical solutions
†††† signed only

VHDL Types

- Floating Point Types – float_pkg
  - 1 type
    - float   array of std_logic
  - 3 sub types
    - float32   float(8 downto -23)
    - float64   float(11 downto -52)
    - float128   float(15 downto -112)
  - Overflow/Underflow/Rounding/Errors
    - Complicated set of rules to deal with these
    - Details are in the spec
  - Operators
    - Comparison: =, /=, <, <=, >, >=
    - Boolean: not, and, or, nand, nor, xor, xnor
    - Arithmetic: : sign -, abs, +, -, *, /, mod, rem
  - Functions
    - operator functions - Comparison and Arithmetic operations have corresponding functions with special modes
    - resize functions
    - utility functions
    - type conversions
    - constants

Float 32 format is:
sign   8 bit exponent   23 bit mantissa
mantissa is normalized to 1. xxx and the 1. removed
exponent is biased by 127
mantissa and exponent are unsigned
Value = (-1*sign)*1.mantissa x $2^{(exponent-127)}$