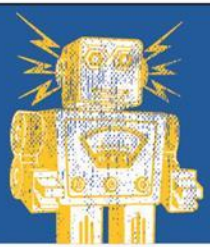


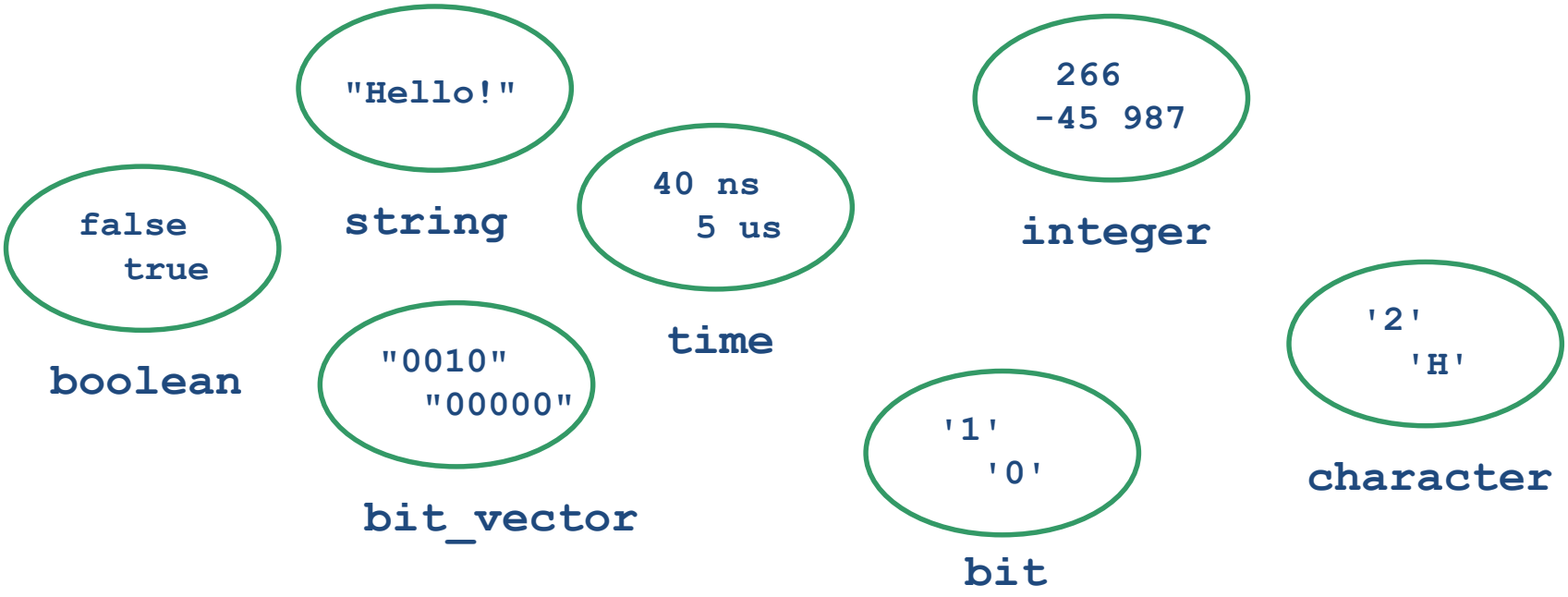
Introduction to VHDL IV

Packages - Libraries - Examples

Roland Höller
Email: hoeller@technikum-wien.at

VHDL Data Types (Standard Package)

- Every signal or variable posesses a type
- The type is defined in the declaration
- For all assignments the types have to match



Other VHDL Data Types

```
library IEEE;
use IEEE.std_logic_1164.all;
```

- Multivalued logic (simulation, bus modeling)
  ('U', 'X', '0', '1', 'Z', 'W', 'L', 'H', '-')
  - std_ulogic, std_ulogic_vector (unresolved)
  - std_logic, std_logic_vector (resolved)

```
library IEEE;
use IEEE.std_logic_arith.all;
```

- For arithmetic operation ('+', '-', '/', '*', ...)
  - signed
  - unsigned

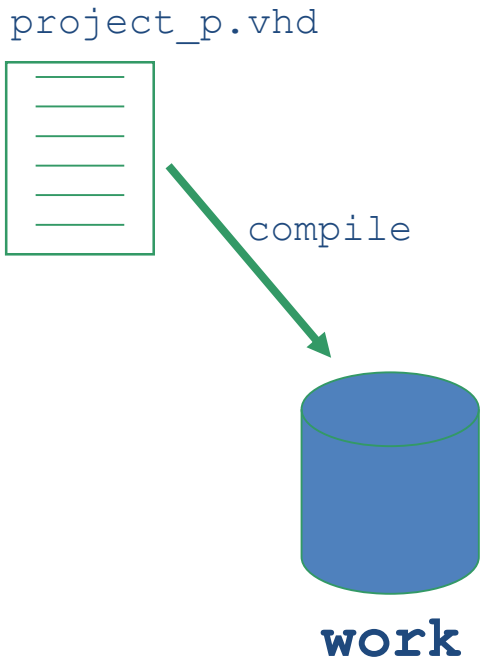
The Package

- Collection of definitions, data types, subprograms
- Referenced via the „use“ statement

```
package project_p is

  constant c_pi : real := 3.14;

  type t_state is (idle_s, read_s);

  component orgate
    port (a_i : in std_logic;
          b_i : in std_logic;
          or_o : out std_logic);
  end component;

end project_p;
```

The Library

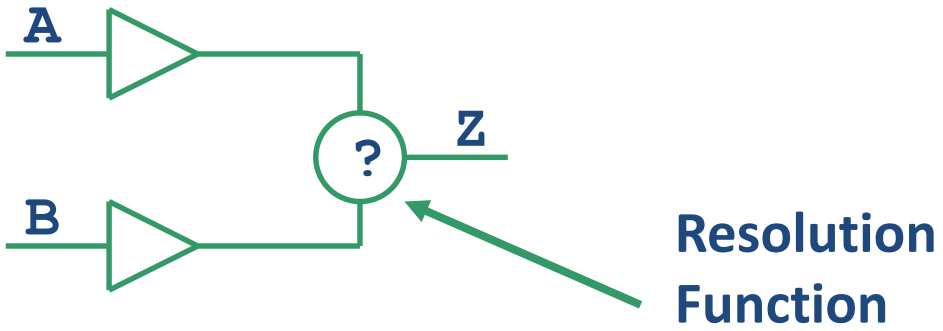
- Collection of compiled design units




```
library work;
use work.project_p.all;

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;
```

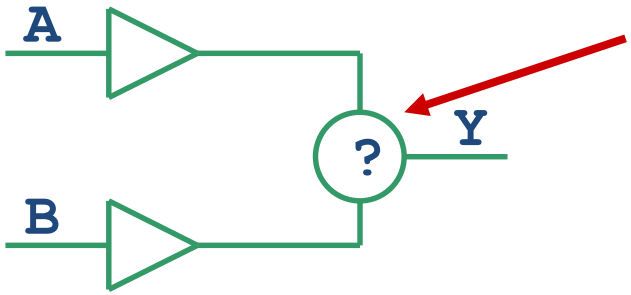
Resolved - Unresolved

- An assignment is represented by a driver:
  - Unresolved data type
    - one signal may only have one driver (std_ulogic, bit)
  - Resolved data type
    - one signal can have several drivers (std_logic, unsigned, signed)




```
-- signal z has more
-- than one driver
z <= a;
z <= b;
```

Resolved - Unresolved



Resolution Function

- **U** – Uninitialized
- **X** – Forcing Unknown
- **0** – Forcing 0
- **1** – Forcing 1
- **Z** – High Impedance
- **W** – Weak Unknown
- **L** – Weak 0
- **H** – Weak 1
- **–** – Don‘t Care

Resolution Table

U	X	0	1	Z	W	L	H	-	
U	U	U	U	U	U	U	U	U	U
U	X	X	X	X	X	X	X	X	X
U	X	0	X	0	0	0	0	X	0
U	X	X	1	1	1	1	1	X	1
U	X	0	1	Z	W	L	H	X	Z
U	X	0	1	W	W	W	W	X	W
U	X	0	1	L	W	L	W	X	L
U	X	0	1	H	W	W	H	X	H
U	X	X	X	X	X	X	X	X	-

User defined VHDL Data Types

- Enumeration types
  - Often used for state machines.

```
type t_state is (IDLE, STATE1, STATE2);
signal s_state : t_state;
```

- Subtypes are used to define a subset of a type

```
subtype t_eightvalues is integer range 0 to 7;
signal s_select : t_eightvalues;
```

Procedures

```
procedure MUX21 (signal s_sel_i   :  in bit;
                 signal s_data0_i :  in bit;
                 signal s_data1_i :  in bit;
                 signal s_data_o  :  out bit) is
begin
  case s_sel_i is
    when '0' => s_data_o <= s_data0_i;
    when others => s_data_o <= s_data1_i;
  end case;
end MUX21;
-------------------------------------------------
procedure MUX21 (signal s_sel_i   :  in std_logic;
                 signal s_data0_i :  in std_logic_vector;
                 signal s_data1_i :  in std_logic_vector;
                 signal s_data_o  :  out std_logic_vector) is
begin
  case s_sel_i is
    when '0' => s_data_o <= s_data0_i;
    when others => s_data_o <= s_data1_i;
  end case;
end MUX21;
```

Functions

```
-- This function calculates the odd or even parity of a bus
-- with variable length and variable array boundaries.
function calc_parity (
         ev_odd : std_ulogic;  -- High = odd, Low = even parity
         data :   unsigned)    -- Data to calculate parity
  return std_ulogic is         -- Odd or even parity

  variable result : std_ulogic;

begin
  if ev_odd = '0' then         -- This is even parity
    result := data(data'right);
    for i in data'right+1 to data'left loop
      result := result xor data(i);
    end loop;  -- i
  else                         -- This is odd parity
```

Functions

```
    result := data(data'right);
    for i in data'right+1 to data'left loop
      result := result xor data(i);
    end loop;  -- i
    result := not(result);
  end if;
  return result;
end;
```

Subprogram Overloading

- Overloaded subprograms are distinguished by
- the number of formal parameters
- the base type of the formal parameter
- the return type of the function

```
signal s_index : integer := 0;
signal s_sum : unsigned(7 downto 0);

begin

s_sum <= unsigned(data1_i) + unsigned(data0_i);

s_index <= s_index + 1;
```

Advanced VHDL I

```
library IEEE;
use IEEE.std_logic_1164.all;

entity mux_chain is
  generic (N : natural := 5);
  port (sel :      in std_logic_vector(N-1 downto 0);
        data_in :  in std_logic_vector(N downto 0);
        data_out : out std_logic);
end mux_chain;

architecture one of mux_chain is
  function mux_chain_func(sel, data: std_logic_vector)
           return std_logic is
  variable i_sel : std_logic_vector(sel'length-1 downto 0);
  variable i_data : std_logic_vector(data'length-1 downto 0);
  variable result : std_logic;
begin
```

Advanced VHDL II

```
  i_sel := sel;
  i_data := data;
  result := i_data(i_data'left);
  for i in i_sel'length-1 downto 0 loop
    if i_sel(i) = '1' then
      result := i_data(i);
    end if;
  end loop;
  return result;
end;

begin
  data_out <= mux_chain_func(sel, data_in);
end one;
```

Adder With Variable Data Width

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;

entity adder is
  generic (N : natural := 4);
  port (a_i :   in  std_logic_vector(N-1 downto 0);
        b_i :   in  std_logic_vector(N-1 downto 0);
        sum_o : out std_logic_vector(N-1 downto 0);
        cy_o :  out std_logic);
end adder;

architecture rtl of adder is
  signal s_sum : unsigned(N downto 0);
begin
  s_sum <= unsigned(a_i) + conv_unsigned(unsigned(b_i),N+1);
  cy_o <= s_sum(N);
  sum_o <= s_sum(N-1 downto 0);
end rtl;
```

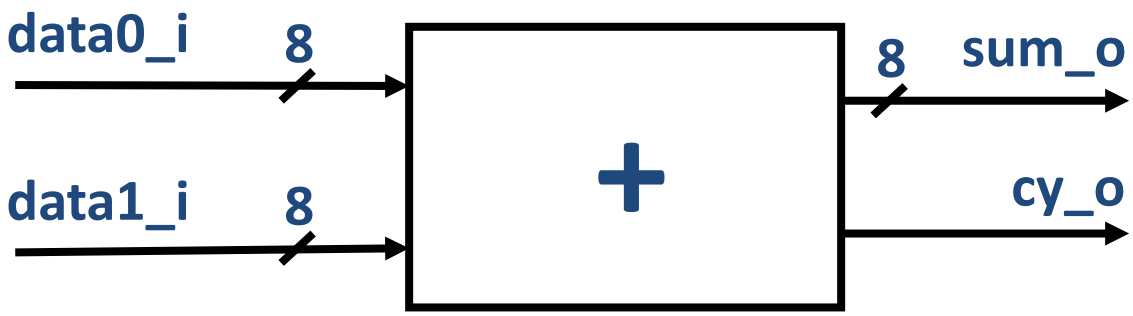
Summary

- VHDL Data Types
- Resolution Function
- Procedures, Functions
- Advanced Examples

Questions

- 1) Which data types does VHDL know natively?
  - A integer
  - B real
  - C bit
  - D char
- 2) What does operator overloading in VHDL mean?
  - A same function name with different parameters is automatically resolved
  - B compile error because of two functions having the same name
  - C compile error because of two functions having the same parameters
  - D not existing in VHDL
- 3) Which VHDL construct allows to place commonly used declarations?
  - A package
  - B architecture
  - C entity
  - D configuration

Class Example: 8-bit Adder



Be careful with the different VHDL data types!

Which VHDL data types can be used with the „+“ function?