VHDL – TD1
Syntaxe VHDL

1 Légal ou illégal ?

(Exemples empruntés à V. Pedroni, “Circuit Design with VHDL ”, MIT Press)

Déterminer les instructions VHDL légales et illégales.

1.1 Legal and illegal operations between data of different types.

```
SIGNAL a: BIT;
SIGNAL b: BIT_VECTOR(7 DOWNTO 0);
SIGNAL c: STD_LOGIC;
SIGNAL d: STD_LOGIC_VECTOR(7 DOWNTO 0);
SIGNAL e: INTEGER RANGE 0 TO 255;
...
a <= b(5);
b(0) <= a;
c <= d(5);
d(0) <= c;
a <= c;
b <= d;
e <= b;
e <= d;
```

1.2 Example: Legal and illegal operations between types and subtypes.

```
SUBTYPE my_logic IS STD_LOGIC RANGE '0' TO '1';
SIGNAL a: BIT;
SIGNAL b: STD_LOGIC;
SIGNAL c: my_logic;
...
b <= a;
b <= c;
```

1.3 Legal and illegal array assignments.

The assignments in this example are based on the following type definitions and signal declarations:

```
TYPE row IS ARRAY (7 DOWNTO 0) OF STD_LOGIC; -- 1D array
TYPE array1 IS ARRAY (0 TO 3) OF row; -- 1Dx1D array
TYPE array2 IS ARRAY (0 TO 3) OF STD_LOGIC_VECTOR(7 DOWNTO 0); -- 1Dx1D
TYPE array3 IS ARRAY (0 TO 3, 7 DOWNTO 0) OF STD_LOGIC; -- 2D array
SIGNAL x: row;
SIGNAL y: array1;
SIGNAL v: array2;
SIGNAL w: array3;

--------- Scalar assignments: ---------------
```

```
x(0) <= y(1)(2);
x(1) <= v(2)(3);
x(2) <= w(2,1);
y(1)(1) <= x(6);
y(2)(0) <= v(0)(0);
y(0)(0) <= w(3,3);
w(1,1) <= x(7);
w(3,0) <= v(0)(3);
--------- Vector assignments: ---------------------
x <= y(0);
x <= v(1);
x <= w(2);
x <= w(2, 2 DOWNTO 0);
v(0) <= w(2, 2 DOWNTO 0);
v(0) <= w(2);
y(1) <= v(3);
y(1)(7 DOWNTO 3) <= x(4 DOWNTO 0);
v(1)(7 DOWNTO 3) <= v(2)(4 DOWNTO 0);
w(1, 5 DOWNTO 1) <= v(2)(4 DOWNTO 0);
```

1.4 Legal and illegal operations with signed/unsigned data types.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all; -- extra package necessary
...
SIGNAL a: IN SIGNED (7 DOWNTO 0);
SIGNAL b: IN SIGNED (7 DOWNTO 0);
SIGNAL x: OUT SIGNED (7 DOWNTO 0);
...
v <= a + b;
w <= a AND b;)
```

1.5 Legal and illegal operations with std_logic_vector.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all; -- no extra package required
...
SIGNAL a: IN STD_LOGIC_VECTOR (7 DOWNTO 0);
SIGNAL b: IN STD_LOGIC_VECTOR (7 DOWNTO 0);
SIGNAL x: OUT STD_LOGIC_VECTOR (7 DOWNTO 0);
...
v <= a + b;
w <= a AND b;
```

1.6 Legal and illegal operations with subsets.

```
TYPE long IS INTEGER RANGE -100 TO 100;
TYPE short IS INTEGER RANGE -10 TO 10;
SIGNAL x : short;
SIGNAL y : long;
...
y <= 2*x + 5;
y <= long(2*x + 5);
```

The legal and illegal assignments presented next are based on the following type definitions and signal declarations:

```
TYPE byte IS ARRAY (7 DOWNTO 0) OF STD_LOGIC; -- 1D array
TYPE mem1 IS ARRAY (0 TO 3, 7 DOWNTO 0) OF STD_LOGIC; -- 2D array
TYPE mem2 IS ARRAY (0 TO 3) OF byte; -- 1Dx1D array
TYPE mem3 IS ARRAY (0 TO 3) OF STD_LOGIC_VECTOR(0 TO 7); -- 1Dx1D array
SIGNAL a: STD_LOGIC; -- scalar signal
SIGNAL b: BIT; -- scalar signal
SIGNAL x: byte; -- 1D signal
SIGNAL y: STD_LOGIC_VECTOR (7 DOWNTO 0); -- 1D signal
SIGNAL v: BIT_VECTOR (3 DOWNTO 0); -- 1D signal
SIGNAL z: STD_LOGIC_VECTOR (x'HIGH DOWNTO 0); -- 1D signal
SIGNAL w1: mem1; -- 2D signal
SIGNAL w2: mem2; -- 1Dx1D signal
SIGNAL w3: mem3; -- 1Dx1D signal

x(2) <= a;
y(0) <= x(0);
z(7) <= x(5);
b <= v(3);
w1(0,0) <= x(3);
w1(2,5) <= y(7);
w2(0)(0) <= x(2);
w2(2)(5) <= y(7);
w1(2,5) <= w2(3)(7);

b <= a;
w1(0)(2) <= x(2);
w2(2,0) <= a;

x <= "11111110";
y <= ('1','1','1','1','1','1','0','Z');
z <= "11111" & "000";
x <= (OTHERS => '1');
y <= (7 =>'0', 1 =>'0', OTHERS => '1');
z <= y;
y(2 DOWNTO 0) <= z(6 DOWNTO 4);
w2(0)(7 DOWNTO 0) <= "11110000";
w3(2) <= y;
z <= w3(1);
z(5 DOWNTO 0) <= w3(1)(2 TO 7);
w3(1) <= "00000000";
w3(1) <= (OTHERS => '0');
w2 <= ((OTHERS=>'0'),(OTHERS=>'0'),(OTHERS=>'0'),(OTHERS=>'0'));
w3 <= ("11111100", ('0','0','0','0','Z','Z','Z','Z',),
      (OTHERS=>'0'), (OTHERS=>'0'));
w1 <= ((OTHERS=>'Z'), "11110000" ,"11110000", (OTHERS=>'0'));

x <= y;
y(5 TO 7) <= z(6 DOWNTO 0);
w1 <= (OTHERS => '1');
w1(0, 7 DOWNTO 0) <="11111111";
w2 <= (OTHERS => 'Z');
w2(0, 7 DOWNTO 0) <= "11110000";
```

2 Attributes

(Exemples empruntés à P. Ashenden, “Systems on Silicon”, Morgan Kaufmann Publishers)

2.1 Attributes of Scalar Types

```
TYPE Resistance IS RANGE 0 TO 1E9
      UNITS
            ohm;
            kohm = 1000 ohm;
            Mohm = 1000 kohm;
      END UNITS Resistance;

TYPE Set_index_range IS RANGE 21 DOWNTO 11;
TYPE Logic_level is (unknown,low,undriven,high);

Resistance'LEFT =
Resistance'RIGHT =
Resistance'LOW =
Resistance'HIGH =
Resistance'ASCENDING =
Resistance'IMAGE(2 kohm) =
Resistance'VALUE("5 Mohm") =

Set_index_range'LEFT =
Set_index_range'RIGHT =
Set_index_range'LOW =
Set_index_range'HIGH =
Set_index_range'ASCENDING =
Set_index_range'IMAGE(14) =
Set_index_range'VALUE("20") =

Logic_Level'LEFT =
Logic_Level'RIGHT =
Logic_Level'LOW =
Logic_Level'HIGH =
Logic_Level'ASCENDING =
Logic_Level'IMAGE(undriven) =
Logic_Level'VALUE("low") =
```

2.2 Attributes of arrays.

```
TYPE A IS ARRAY (1 TO 4, 31 DOWNTO 0) OF BOOLEAN;

A'LEFT(1) =
A'LOW(1) =
A'RIGHT(2) =
A'HIGHT(2) =
A'LENGTH(1) =
A'LENGTH(2) =
A'RANGE(1) =
A'REVERSE_RANGE(2) =
A'ASCENDING(1) =
A'ASCENDING(2) =
```