

# Hardware Modeling [VU] (191.011)

## – WS25 –

### Composite Types

Florian Huemer & Sebastian Wiedemann & Dylan Baumann

WS 2025/26

- VHDL type classes (repetition)
  - scalar
  - ***composite***
  - file
  - access
  - protected
- Composite types
  - Array types (homogeneous collections)
  - Record types (heterogeneous collections)
- Value types

# Array Types

HWMod  
WS25

Comp. Types

Arrays

Predef. Arrays

Element Access

Multidim. Arrays

Attributes

Records

Aggregates

## ■ Declaration syntax

```
type TYPE_NAME is
  array(range_constraints) of element_type;
```

## ■ Can be constrained or unconstrained

## ■ Examples

```
1 -- declaration
2 type u_t is array(integer range <>) of boolean; -- unconstrained
3 type c_t is array(0 to 13) of boolean; -- constrained
4 -- usage
5 variable u : u_t(3 downto -7);
6 variable c : c_t;
7 constant const_u : u_t := 1 & 2 & 3; -- range is inferred
```

## ■ Synthesizeable, if elements are synthesizable

## ■ Needed to describe memory

# Predefined Array Types

HWMod  
WS25

Comp. Types

Arrays

Predef. Arrays

Element Access

Multidim. Arrays

Attributes

Records

Aggregates

## ■ `string` type

### ■ Declaration

```
type string is  
    array (positive range <>) of character; IEEE SA  
OPEN
```

### ■ Unconstrained with **positive** range

### ■ Usage example

```
constant CONST_STRING : string := "Hello World";
```

## ■ Other predefined array types

### ■ type `integer_vector` is

```
array (natural range <>) of integer; IEEE SA  
OPEN
```

### ■ type `boolean_vector` is

```
array (natural range <>) of boolean; IEEE SA  
OPEN
```

### ■ type `time_vector` is

```
array (natural range <>) of time; IEEE SA  
OPEN
```

# Element Access

HWMod  
WS25

Comp. Types

Arrays

Predef. Arrays

Element Access

Multidim. Arrays

Attributes

Records

Aggregates

- Element access syntax
  - **Not** via square brackets, i.e., [ ]
  - Parentheses with integer index as parameter
  - Similar to function calls
- Element access example

```
1 process
2   variable a : integer_vector(0 to 3);
3 begin
4   a(0) := 1; -- write access
5   report to_string(a(0)); -- read access
6   wait;
7 end process;
```

- Runtime range checks

# Element Access (cont'd)

HWMod  
WS25

Comp. Types

Arrays

Predef. Arrays

Element Access

Multidim. Arrays

Attributes

Records

Aggregates

- Range access syntax: parentheses with integer range expression
- Range access example

```
1 process
2   variable a : integer_vector(0 to 5);
3 begin
4   a(0) := 4;
5   a(1) := 2;
6   a(2 to 5) := a(0 to 1) & a(0 to 1);
7   for i in 0 to 5 loop
8     report to_string(a(i));
9   end loop;
10  wait;
11 end process;
```

# Multidimensional Array Examples

HWMod  
WS25

Comp. Types

Arrays

Predef. Arrays

Element Access

Multidim. Arrays

Attributes

Records

Aggregates

## ■ 2D Array

```
1 process
2   type a2d_t is array(integer range <>, integer range <>) of boolean;
3   variable a2d : a2d_t(3 downto -4, 0 to 7);
4 begin
5   a2d(-4, 0) := true;
6   report to_string(a2d(-4, 0)); -- output: true
7   wait;
8 end process;
```

## ■ Array of an Array

```
1 process
2   type aoa_t is array(integer range <>) of boolean_vector(0 to 7);
3   variable aoa : aoa_t(3 downto -4);
4 begin
5   aoa(-4)(0) := true;
6   report to_string(boa(-4)(0)); -- output: true
7   wait;
8 end process;
```

# Array Attributes

HWMod  
WS25

Comp. Types

Arrays

Predef. Arrays

Element Access

Multidim. Arrays

Attributes

Records

Aggregates

- Defined for array-type objects (variables, constants, etc.)  
Important: **not** for the type itself
- Example

```
1 process
2   type a_t is array(1 downto -1) of boolean;
3   variable a : a_t;
4 begin
5   report "length=" & to_string(a'length); -- output: length=3
6   wait;
7 end process;
```

- Other attributes: low, high, left, right, ascending, etc. (see VHDL standard)

# Array Attributes – Range

HWMod  
WS25

Comp. Types

Arrays

Predef. Arrays

Element Access

Multidim. Arrays

Attributes

Records

Aggregates

- Access an array's declared range

- Example

```
1 process
2   variable a : integer_vector(2 downto 0) := 1 & 2 & 3;
3 begin
4   report "range";
5   for i in a'range loop
6     report "index=" & to_string(i) & ", value=" & to_string(a(i));
7   end loop;
8   wait;
9 end process;
```

- Output

```
[...]: index=2, value=1
[...]: index=1, value=2
[...]: index=0, value=3
```

# Array Attributes – Multidimensional Arrays

HWMod  
WS25

Comp. Types

Arrays

Predef. Arrays

Element Access

Multidim. Arrays

Attributes

Records

Aggregates

- Array dimension specified by additional integer ( $\geq 1$ )

- Example

```
1 process
2   type a_t is array(3 downto 0, 1 to 2) of boolean;
3   variable a : a_t;
4 begin
5   report "length(1)=" & to_string(a'length(1));
6   report "length(2)=" & to_string(a'length(2));
7   wait;
8 end process;
```

- Output

```
[..]:(report note): length(1)=4
[..]:(report note): length(2)=2
```

# Record Types

HWMod  
WS25

Comp. Types

Arrays

Records

Element Access

Unconstrained

Elements

Aggregates

- Composed of elements of (potentially) different types (comparable to C structs)
- Declaration syntax

```
type TYPE_NAME is record
    {element_declaration}
end record;
```

- Example

```
1 type my_record_t is record
2   a : integer_vector(7 downto 0);
3   b : boolean;
4   c, d : integer;
5 end record;
```

- Synthesizable (except if they comprise non-synthesizable data types)

# Element Access

HWMod  
WS25

Comp. Types

Arrays

Records

Element Access

Unconstrained

Elements

Aggregates

- Element access syntax: dot operator
- Example

```
1 process
2   type vec2_t is record
3     x, y : real;
4   end record;
5   variable v : vec2_t;
6 begin
7   v.x := 1.0;
8   v.y := 2.0;
9   report "x=" & to_string(v.x) & ", " &
10        "y=" & to_string(v.y); -- output: x=1.0, y=2.0
11   wait;
12 end process;
```

# Unconstrained Elements

HWMod  
WS25

Comp. Types

Arrays

Records

Element Access

Unconstrained  
Elements

Aggregates

- Records can have elements of unconstrained types
- Constrain types when record is used
- Example

```
1 process
2   type ur_t is record
3     a : integer_vector; -- unconstrained
4     b : boolean_vector; -- unconstrained
5     c : real_vector(1 downto 0); --constrained
6   end record;
7   variable v : ur_t(a(1 downto 0), b(0 to 7));
8 begin
9   wait;
10 end process;
```

# Aggregate Expressions

HWMod  
WS25

Comp. Types

Arrays

Records

Aggregates

Positional  
Association

Named Association

Others Clause

- Used to initialize and manipulate values of composite data types

- Syntax

- Aggregates are enclosed by parentheses, i.e., ( )
- Individual elements separated by commas
- Example: (1, 2, 3) (equivalent to 1 & 2 & 3)

- Array initialization example

```
constant A : integer_vector(0 to 2) := (1, 2, 3);
```

- Two categories

- Positional association
- Named association

# Positional Association

HWMod  
WS25

Comp. Types

Arrays

Records

Aggregates

Positional  
Association

Named Association

Others Clause

## ■ Array examples

```
1 constant A : boolean_vector(1 downto 0) := (true, false) -- a(1)=true
2 constant B : integer_vector(2 to 4) := (1, 2, 3); -- b(2)=1
3 constant C : string(1 to 3) := ('a', 'b', 'c'); -- equivalent to "abc"
4 constant D : bit_vector(2 downto 0) :=
5     ('1', '0', '1'); -- equivalent to "101"
```

## ■ Record examples

```
1 type vec3d_t is record
2     x, y, z : real;
3 end record;
4 constant V : vec3d_t := (1.0, 0.0, 0.0);
5
6 type demo_t is record
7     a : integer_vector(1 downto 0);
8     v : vec3d_t;
9     b : boolean;
10 end record;
11 constant DEMO : demo_t := ((1, 2), V, true);
```

# Named Association

HWMod  
WS25

Comp. Types

Arrays

Records

Aggregates

Positional

Association

Named Association

Others Clause

## ■ Array examples

```
1 constant A : boolean_vector(1 downto 0) :=
2     (1 => true, 0 => false); -- equivalent to (true, false)
3 constant B : boolean_vector(7 downto 0) :=
4     (7 => true, 0 => true, 6 downto 1 => false);
```

## ■ Record example

```
1 constant DEMO : demo_t := (
2     a => (1, 2),
3     v => (x => 1.0, y => 0.0, z => 0.0),
4     b => true
5 );
```

# Others Clause

HWMod  
WS25

Comp. Types

Arrays

Records

Aggregates

Positional  
Association

Named Association

Others Clause

## ■ Assign multiple elements at once

## ■ Array examples

```
1 constant A : boolean_vector(63 downto 0) := (others=>false);
2 constant B : integer_vector(63 downto 0) := (1, others=>0); -- b(63)=1
3 constant C : integer_vector(63 downto 0) := (0=>1, others=>0); -- c(0)=1
```

## ■ Record examples

```
1 constant V0 : vec3d_t := (x=>1.0, others=>0.0); -- v0.x=1.0
2 constant V1 : vec3d_t := (1.0, others=>0.0); -- v1.x=1.0
3 constant V2 : vec3d_t := (z=>1.0, others=>0.0); -- v2.z=1.0
4 constant D : demo_t := (v=>V2, others=>true); -- error
```

# Lecture Complete!