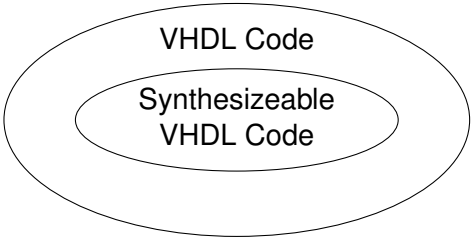
Hardware Modeling [VU] (191.011)
– WS25 –
Circuit Description with Entities and Architectures

Florian Huemer & Sebastian Wiedemann & Dylan Baumann

WS 2025/26

Modified: 2025-12-16, 16:08 (f8a58e9)

Introduction

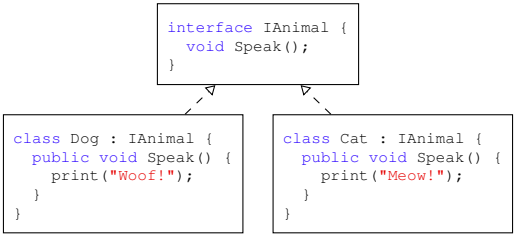


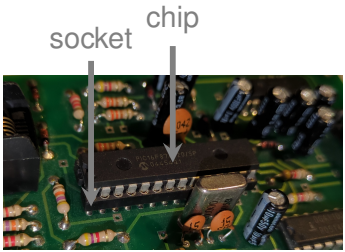
Recall

Not everything in VHDL is synthesizable.

Entities

- Interface specification of a module
  - Interface signals
  - Parameters
- No internal behavior specified
- Multiple different architectures possible
- Analogies





Entities (cont'd)

- Entity declaration syntax

```
entity NAME is
  [ generic ( {generic_element;} generic_element ); ]
  [ port ( {port_element;} port_element ); ]
  [ entity_declarative_part ]
  [ begin entity_statement_part ]
end entity;
```

- Generic clause: configuration/parameters
- Port clause: physical I/O signals
- Declarations: constants, types, subprograms, etc.
- Statements: parameter checks

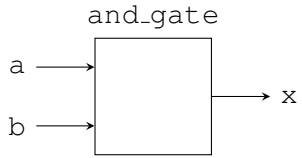
Ports

- Port element declaration syntax

```
port_element ::=
  PORT_NAME : [mode] DATA_TYPE [:= default_value]
```

- Mode
  - Possible Values: `in`, `out`, `inout`, `buffer`, `linkage`
  - Default (if mode is omitted): `in`
  - Defines the direction of a signal (can it be read, written or both)
- Example entity

```
1 entity and_gate is
2  port (
3   a : in boolean;
4   b : in boolean;
5   x : out boolean -- no semicolon here!
6  );
7 end entity;
```



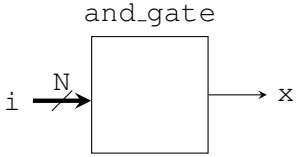
Generics

- Generic element declaration syntax

```
generic_element ::=
  GENERIC_NAME : DATA_TYPE [:= default_value]
```

- No mode required (all generics are “constant inputs”)
- Basically constants in the architecture
- Example entity

```
1 entity and_gate is
2   generic (
3     N : integer := 2
4   );
5   port (
6     i : in boolean_vector(N-1 downto 0);
7     x : out boolean
8   );
9 end entity;
```



Unconstrained Types

- Unconstrained generic example

```
1 entity cpu is
2   generic (
3     ENABLE_BRANCH_PREDICTION : string := "YES";  -- "NO"
4     [...]
5   );
6   [...]
7 end entity;
```

- Unconstrained port example

```
1 entity and_gate is
2   port (
3     i : in boolean_vector;
4     x : out boolean
5   );
6 end entity;
```

Architectures

- VHDL standard  :
  "An architecture body defines the body of a design entity."
- Contains the actual circuit description
- Architecture declaration syntax

```
architecture NAME of ENTITY_NAME is
  architecture_declarative_part
begin
  architecture_statement_part
end architecture;
```

- Declarative/statement part

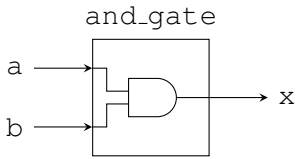
```
architecture_declarative_part ::=
                              { block_declarative_item }
architecture_statement_part ::=
                               { concurrent_statement }
```

Architecture (cont'd)

- Declarative part (block declarative items)
  - Signals
  - Constants
  - Types
  - Sub-programs
  - etc.
- Statement part (concurrent statements)
  - **Concurrent signal assignments**
  - Processes
  - Instantiation statements
  - Blocks statements
  - Generate statements
  - etc.

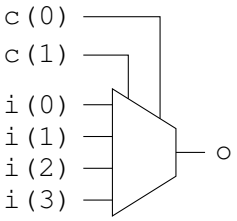
Concurrent Signal Assignments – Example: AND gate


```
1  entity and_gate is
2   port (
3    a : in boolean;
4    b : in boolean;
5    x : out boolean
6   );
7  end entity;
8
9  architecture arch of and_gate is
10 begin
11  x <= a and b;
12 end architecture;
```



Concurrent Signal Assignments – Example: Multiplexer


```
1  entity mux41 is
2   port (
3    c : in boolean_vector(1 downto 0);
4    i : in boolean_vector(3 downto 0);
5    o : out boolean
6   );
7  end entity;
8
9  architecture arch of mux41 is
10 begin
11  o <= i(0) when not c(1) and not c(0) else
12       i(1) when not c(1) and     c(0) else
13       i(2) when     c(1) and not c(0) else
14       i(3) when     c(1) and     c(0);
15 end architecture;
```



Signal Declarations

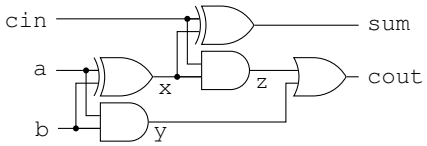
- Signal declaration syntax

```
signal NAME : DATA_TYPE [ := default_value ] ;
```

- Signals vs. variables

Signals	Variables
Declaration: (as) ports, entities, architectures, packages	Declaration: subprograms, processes
Assignment: deferred (event queue)	Assignment: immediate
Assignment operator: `<=`	Assignment operator: `:=`

Example - Full Adder



Important

The order of statements in an architecture carries **no** semantic significance!

```
1  entity fa is
2   port (
3    a, b, cin  : in boolean;
4    sum, cout : out boolean
5   );
6  end entity;
7
8  architecture arch of fa is
9   signal x, y, z : boolean;
10 begin
11  x <= a xor b;
12  y <= a and b;
13  sum <= cin xor x;
14  z <= cin and x;
15  cout <= y or z;
16 end architecture;
```

Lecture Complete!