# VHDL I

Tom Kelliher, CS 240

Mar. 1, 2006

## 1 Administrivia

**Announcements**

Study for the exam:

1. Boolean algebra and boolean identities.
2. Minterms.
3. Karnaugh maps, map minimization.
4. Circuit realization using AND, OR, and NOT gates.
5. Addition's lower bound.
6. Carry lookahead and radix 2 signed-digit addition.

**Assignment**

**From Last Time**

Carry-lookahead and signed-digit addition.

**Outline**

1. VHDL program structure.
2. Structural VHDL.
3. Class practice.

**Coming Up**

Exam I.

## 2 VHDL Program Structure

VHDL is **case insensitive**!!

1. Structure of a VHDL program:

   ```
   Library includes;
   Entity declaration;
   Architectural definition of entity;
   ```

2. Library includes:

   ```
   -- This is a comment.
   library ieee, lcdf_vhdl;
   use ieee.std_logic_1164.all, lcdf_vhdl_.func_prims.all;
   ```

   Reserved words: library, use, .all.

   Similar to import, include statements.

3. Entity declaration:

```
entity entity_name is
   port(i0, i1, i2 : in std_logic;
        o0         : out std_logic);
end entity_name;
```

Reserved words: entity, is, port, in, out, end.

Note that `entity_name` follows `end`.

4. Architectural definition of entity:

```
architecture arch_name of entity_name is

   component declarations;
   signal declarations;

   begin
   VHDL statements;
end arch_name;
```

Reserved words: architecture, of, begin.

`entity_name` must match. `arch_name` is just a "place holder" — possible to describe an entity with multiple architectures.

Again, note that `arch_name` follows `end`.

5. Component declaration:

```
component component_name
   port(i0, i1 : in std_logic;
        o0     : out std_logic);
end component;
```

Reserved words: component.

Like base class declarations in C++.

6. Signal declarations:

```
signal s0, s1, s2 : std_logic;
```

Similar to variable declarations.

## 3 Structural VHDL

1. Describes structure of a circuit — similar to netlist. Low-level description.

2. Example: Three input EXOR.

   Equation: $\overline{i_2}\,\overline{i_1}i_0 + \overline{i_2}i_1\overline{i_0} + i_2\overline{i_1}\,\overline{i_0} + i_2i_1i_0$

   VHDL:

```
library ieee, lcdf_vhdl;
use ieee.std_logic_1164.all, lcdf_vhdl_.func_prims.all;

entity EXOR2 is
   port(i2, i1, i0 : in std_logic;
        o          : out std_logic);
end EXOR2;

arch structural of EXOR2 is

   component NOT1
      port(in1 : in std_logic;
           out1 : out std_logic;);
   end component;

   component NAND3
      port(in1, in2, in3 : in std_logic;
           out1          : out std_logic);
   end component;

   component NAND4
      port(in1, in2, in3, in4 : in std_logic;
           out1               : out std_logic);
   end component;

   signal i2_n, i1_n, i0_n, t3, t2, t1, t0 : std_logic;

   begin

      g0: NOT1 port map(i2, i2_n);
      g1: NOT1 port map(i1, i1_n);
      g2: NOT1 port map(i0, i0_n);

      g3: NAND3 port map(i2_n, i1_n, i0, t3);
      g4: NAND3 port map(i2_n, i1, i0_n, t2);
      g5: NAND3 port map(i2, i1_n, i0_n, t1);
      g6: NAND3 port map(i2, i1, i0, t0);

      g7: NAND4 port map(t3, t2, t1, t0, o);

end structural;
```

## 4 Class Practice

Write structural VHDL for carry bit of full adder.