VHDL Review

Last updated 2/1/24

VHDL Review

- VHDL
- VHSIC Hardware Description Language
- Very High Speed Integrated Circuit

VHDL Review

- Synthesizable constructs
  - Code that can be converted to hardware
  - Concurrent logic
  - Sequential logic (processes with edge detection)
  - Structural logic
  - with-select, when-else, …
  - if-else, case, …

- Un-synthesizable constructs
  - Code that cannot be converted to hardware
    - Allowed for simulation, descriptive or compilation purposes
  - Time – wait, delay
  - Loops – while, for, …
  - Initial conditions
  - Variables
    - Strictly speaking variables can be used in synthesis – but it is very dangerous

VHDL Review

- Concurrent Logic in HDL
  - Concurrent activities are happening all the time (in parallel)
    - Assignment: <=
    - with-select
    - when-else
    - …

```
z <= (b or c) when (d = '0') else (e and f);
```

z can change if any value changes, immediately

VHDL Review

- Sequential Logic in HDL
  - Sequential activities only happen in certain situations
    - E.g. Rising edge of clock
  - Sequential activities are identified by placing them in a "process" block
  - Process blocks are only executed when a signal in the blocks "sensitivity list" changes
  - Sequential Signals are ONLY updated at the end of the process block
  - Process blocks themselves are concurrent activities

VHDL Review

- Process block format

```
process (sensitivity list)
  begin
    actions
  end process;
```

```
label process (sensitivity list)
  begin
    actions
  end process;
```

VHDL Review

- VHDL file components
  - Header
    - Description of who created the file
    - Description of the purpose of the block
    - Description of inputs and outputs
  - Inclusions
    - Any libraries that will be referenced in the design
  - Entity
    - Formal definition of the inputs and outputs
    - Any generic parameters are defined here also
  - Architecture
    - Internal signal declarations
    - Component declarations for hierarchical blocks used in the design
    - Instantiation (hook-up) of any hierarchical blocks used in the design
    - HDL description of the desired functionality
      - Concurrent, sequential, structural logic

VHDL Review

- VHDL file components - example

```
----------------------------
--
-- basic_vhdl.vhdl
-- Created: 7/16/18
-- By: johnsontimoj
-- For: EE3921
--

-- File Overview ---
--
-- This file demonstrates basic VHDL file structure

-- File Details ---
-----------------------------
```

Header Information

Document
who, when, what and how

Should be sufficient to allow a co-worker to use/modify the design

VHDL Review

- VHDL file components - example

```
----------------------------
--
-- basic_vhdl.vhdl
-- Created: 7/16/18
-- By: johnsontimoj
-- For: EE3921
--

-- File Overview ---
--
-- This file demonstrates basic VHDL file structure

-- File Details ---
-----------------------------
```

Header Information

```
-- Library inclusions
library IEEE;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
```

Library Inclusions

Companies may have their own libraries
- we will use the IEEE standard library

Only include the portions of the library you need
- ieee.std_logic_1164.all provides:
  - std_logic
  - std_logic_vector
  - 9 logic levels
- ieee.numeric_std.all provides
  - Signed vectors – signed, unsigned

VHDL Review

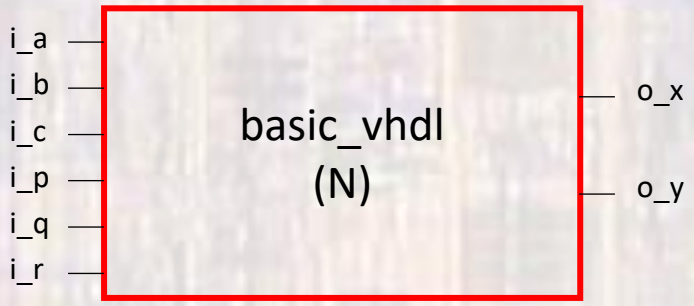
- VHDL file components - example

```
----------------------------
--
-- basic_vhdl.vhdl
-- Created: 7/16/18
-- By: johnsontimoj
-- For: EE3921
--

-- File Overview ---
--
-- This file demonstrates basic VHDL file structure

-- File Details ---
-----------------------------
```

Header Information

```
-- Library inclusions
library IEEE;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
```

Library Inclusions



Generic

Ports

```
-- Entity definition
entity basic_vhdl is
    generic( N:     positive := 8);
    port(    i_a:   in       std_logic;
             i_b:   in       std_logic;
             i_c :  in       std_logic;
             i_p:   in       std_logic_vector(N-1 downto 0);
             i_q:   in       std_logic_vector(N-1 downto 0);
             i_r :  in       std_logic_vector(N-1 downto 0);
             o_x:   out      std_logic;
             o_y:   out      std_logic_vector(N-1 downto 0)
    );
end entity;
```

Entity

The Entity is the external view of the block

Generics indicate 'adjustable' aspects of the block

Ports are the i/o connections of the block

Ports have a direction but no interpretation

Wires

Limited to std_logic or std_logic_vector for us

VHDL Review

- VHDL file components - example

```
----------------------------
--
-- basic_vhdl.vhdl
-- Created: 7/16/18
-- By: johnsontimoj
-- For: EE3921
--
```

Header Information

The Architecture is the operational description of the block
- Behavioral
- Structural
- Testbench
- …

Before the begin
- Internal signals
- Constants used for readability

After the begin
- HDL for the specific application

```
    i_q:    in      std_logic_vector(N-1 downto 0);
    i_r :   in      std_logic_vector(N-1 downto 0);
    o_x:    out     std_logic;
    o_y:    out     std_logic_vector(N-1 downto 0)
  );
end entity;
```

```
-- Behavioral Architecture Definition
architecture behavioral of basic_vhdl is
    signal e:   std_logic;
    signal f:   std_logic;
    signal g:   std_logic;
    signal t:   unsigned(N-1 downto 0);
    signal u:   signed(N-1 downto 0);

    Begin

        e <= i_a and i_b;
        f <= i_c nor i_a;
        g <= e xor f;
        o_x <= g and not i_p(3);

        t <= unsigned(i_p xor i_q);
        u <= signed("11" & i_a & t(4) & i_r(3 downto 2) & i_a & g);
        o_y <= (7 => '1' , 4 => u(5), 1 => i_a , 5 => t(4) , others => '1' );    -- array assignment

end architecture;
```

Architecture

Internal signals

Architectural Definition

VHDL Review

- VHDL file components - example

```
-----------------------------
--
-- basic_vhdl.vhdl
-- Created: 7/16/18
-- By: johnsontimoj
-- For: EE3921
--

-- File Overview ---
--
-- This file demonstrates basic VHDL file structure

-- File Details ---
-----------------------------
```

Header Information

```
-- Library inclusions
library IEEE;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
```

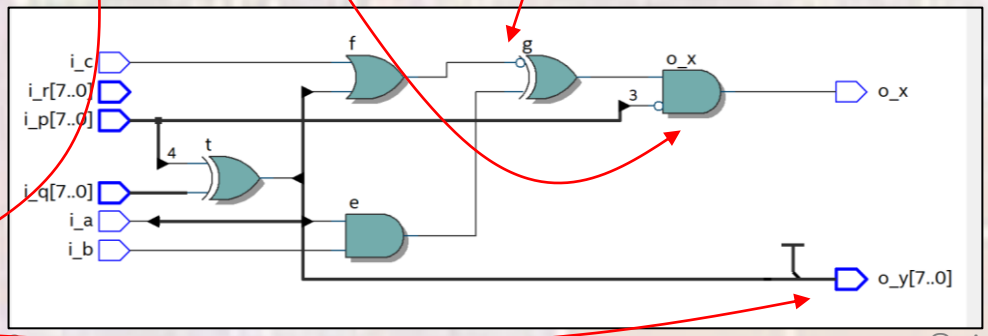
Library Inclusions

```
-- Entity definition
entity basic_vhdl is
    generic( N:       positive := 8);
    port(     i_a:    in      std_logic;
              i_b:    in      std_logic;
              i_c :   in      std_logic;
              i_p:    in      std_logic_vector(N-1 downto 0);
              i_q:    in      std_logic_vector(N-1 downto 0);
              i_r :   in      std_logic_vector(N-1 downto 0);
              o_x:    out     std_logic;
              o_y:    out     std_logic_vector(N-1 downto 0)
    );
end entity;
```

Entity

```
-- Behavioral Architecture Definition
architecture behavioral of basic_vhdl is
    signal e:   std_logic;
    signal f:   std_logic;
    signal g:   std_logic;
    signal t:   unsigned(N-1 downto 0);
    signal u:   signed(N-1 downto 0);


    Begin

        e <= i_a and i_b;
        f <= i_c nor i_a;
        g <= e xor f;
        o_x <= g and not i_p(3);

        t <= unsigned(i_p xor i_q);
        u <= signed("11" & i_a & t(4) & i_r(3 downto 2) & i_a & g);
        o_y <= (7 => '1' , 4 => u(5), 1 => i_a , 5 => t(4) , others => '1' );    -- array assignment

end architecture;
```

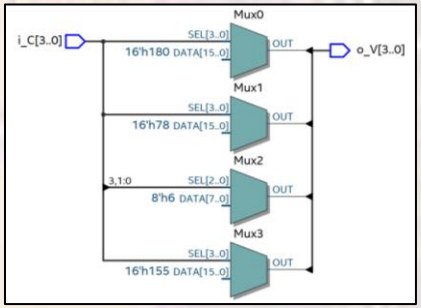
Architecture



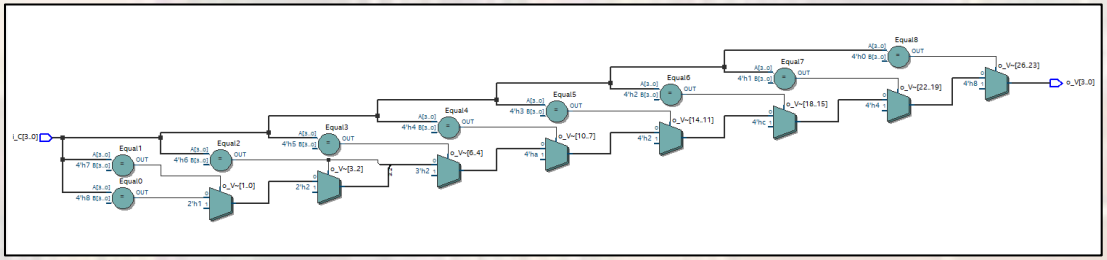
VHDL Review

- Concurrent / Sequential constructs
  - Concurrent logic can be created with either the concurrent logic constructs with-select and when-else or the process constructs if-else, case
  - The process construct versions of concurrent logic are more flexible

with-select
and/or
case

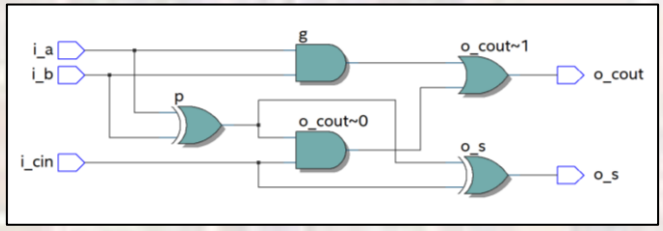


when-else
and/or
if-else



VHDL Review

- Structural VHDL
  - Explicitly build the logic from hierarchical blocks
  - Example
    - Full Adder block




```
---------------------------------
--
-- full_adder.vhdl
--
-- by: johnsontimoj
--
-- created: 7/5/2017
--
-- version: 0.0
--
---------------------------------
---------------------------------
--
-- 1 bit full adder
--
-- inputs: a, b, cin
--
-- outputs: s, cout
--
---------------------------------

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

```
entity full_adder is
    port( i_a:      in  std_logic;
          i_b:      in  std_logic;
          i_cin:    in std_logic;
          o_s:      out std_logic;
          o_cout:   out std_logic
    );
end entity;

architecture behavioral of full_adder is

    signal p: std_logic;
    signal g: std_logic;

begin

    p <= i_a xor i_b;
    g <= i_a and i_b;

    o_s <= p xor i_cin;
    o_cout <= g or (p and i_cin);

end;
```

VHDL Review

- Structural VHDL
  - 4 bit adder example

```
----------------------------------
--
--  adder_4bit.vhdl
--
--  by: tj
--
--  created: 7/5/2017
--
--  version: 0.0
--
----------------------------------
--
--  4 bit adder to show cell instantiation
--
--  inputs: - a, b, cin
--
--  outputs: - sum, cout
--
----------------------------------
library IEEE;
use ieee.std_logic_1164.all;

entity adder_4bit is
    port(   i_A:    in  std_logic_vector(3 downto 0);
            i_B:    in std_logic_vector(3 downto 0);
            i_CIN:      in std_logic;
            o_SUM:      out std_logic_vector(3 downto 0);
            o_COUT:     out std_logic
    );
end entity;

architecture structural of adder_4bit is
```

Top Level Entity

```
----------------------------------
--  1 bit full adder prototype
----------------------------------
component full_adder is
    port(   i_a:        in  std_logic;
            i_b:        in  std_logic;
            i_cin:      in std_logic;
            o_s:        out std_logic;
            o_cout:     out std_logic
    );
end component;
-------------------------------------
-- intermediate carrys mapped to co
-- with 1st stage Cout mapped to co(0) and 4th stage cout mapped to co(3)
-------------------------------------
signal co: STD_LOGIC_VECTOR(3 downto 0); -- intermediate carrys
                                         -- no vector interpretation
```

Component Prototype

```
begin

    add_0:  full_adder port map(  i_a => i_A(0),
                                  i_b => i_B(0),
                                  i_cin => i_CIN,
                                  o_s => o_SUM(0),
                                  o_cout => co(0)
                                  );
    add_1:  full_adder port map(  i_a => i_A(1),
                                  i_b => i_B(1),
                                  i_cin => co(0),
                                  o_s => o_SUM(1),
                                  o_cout => co(1)
                                  );
    add_2:  full_adder port map(  i_a => i_A(2),
                                  i_b => i_B(2),
                                  i_cin => co(1),
                                  o_s => o_SUM(2),
                                  o_cout => co(2)
                                  );
    add_3:  full_adder port map(  i_a => i_A(3),
                                  i_b => i_B(3),
                                  i_cin => co(2),
                                  o_s => o_SUM(3),
                                  o_cout => co(3)
                                  );


    o_COUT <= co(3);

end architecture;
```

Instantiations

Explicit Port Mapping

VHDL Review

- Structural VHDL
  - 4 bit adder example